UNIVERSITY OF CALIFORNIA

Los Angeles

On the Performance and Linear Convergence of Decentralized Primal-Dual Methods

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Sulaiman A. Alghunaim

2020

ABSTRACT OF THE DISSERTATION

On the Performance and Linear Convergence of Decentralized Primal-Dual Methods

by

Sulaiman A. Alghunaim
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2020
Professor Ali H. Sayed, Chair

This dissertation studies the performance and linear convergence properties of primal-dual methods for the solution of decentralized multi-agent optimization problems. Decentralized multi-agent optimization is a powerful paradigm that finds applications in diverse fields in learning and engineering design. In these setups, a network of agents is connected through some topology and agents are allowed to share information only locally. Their overall goal is to seek the minimizer of a global optimization problem through localized interactions. In decentralized consensus problems, the agents are coupled through a common consensus variable that they need to agree upon. While in decentralized resource allocation problems, the agents are coupled through global affine constraints.

Various decentralized consensus optimization algorithms already exist in the literature. Some methods are derived from a primal-dual perspective, while other methods are derived as gradient tracking mechanisms meant to track the average of local gradients. Among the gradient tracking methods are the adapt-then-combine implementations motivated by diffusion strategies, which have been observed to perform better than other implementations. In this dissertation, we develop a novel *adapt-then-combine* primal-dual algorithmic framework that captures most state-of-the-art gradient based methods as special cases including all the variations of the gradient-tracking methods. We also develop a concise and novel analysis technique that establishes the linear convergence of this general framework under strongly-

convex objectives. Due to our unified framework, the analysis reveals important characteristics for these methods such as their convergence rates and step-size stability ranges. Moreover, the analysis reveals how the augmented Lagrangian penalty term, which is utilized in most of these methods, affects the performance of decentralized algorithms.

Another important question that we answer is whether decentralized proximal gradient methods can achieve global linear convergence for *non-smooth* composite optimization. For centralized algorithms, linear convergence has been established in the presence of a non-smooth composite term. In this dissertation, we close the gap between centralized and decentralized proximal gradient algorithms and show that decentralized proximal algorithms can also achieve linear convergence in the presence of a non-smooth term. Furthermore, we show that when each agent possesses a *different* local non-smooth term then global linear convergence cannot be established in the worst case.

Most works that study decentralized optimization problems assume that all agents are involved in computing all variables. However, in many applications the coupling across agents is sparse in the sense that only a few agents are involved in computing certain variables. We show how to design decentralized algorithms in sparsely coupled consensus and resource allocation problems. More importantly, we establish analytically the importance of exploiting the sparsity structure in coupled large-scale networks.

The dissertation of Sulaiman A. Alghunaim is approved.

Abeer Alwan

Robert M'Closkey

Lieven Vandenberghe

Ali H. Sayed, Committee Chair

University of California, Los Angeles

2020

To my family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, all praise be to Allah (God).

I would like to express my gratitude to my adviser professor Ali H. Sayed for giving me the opportunity to work under his supervision at the Adaptive System Laboratory at UCLA. Professor Sayed's patience with me in research has greatly improved the quality of my work. Through our interactions, I have learned many valuable lessons from him whether directly said or indirectly through his high standards and passion in work. I am sure that these lessons will stick with me forever.

I am grateful to all my committee members, Professor Abeer Alwan, Professor Robert M'Closkey, and Professor Lieven Vandenberghe. I thank them for their time spent serving on my committee.

I would like to thank Ryo Arreola and Deeona Columbia for their help during my time at UCLA. At UCLA, I have met many colleagues, visitors, and collaborators: Lucas Cassano, Kun Yuan, Bicheng Ying, Stefan Vlaski, Hawraa Salami, Chung-Kai Yu, Ernest K. Ryu, Chengcheng Wang, Edward Nguyen, and Professor João Y. Ishihara. I am sure that I have learned a lot from them through our discussions and chats. I also cannot forget the people I met during my visit to EPFL: Patricia Vonlanthen, Roula Nassif, Guillermo Ortiz Jimenez, Professor Ricardo Merched, Augusto Santos, Elsa Rizk, and Virginia Bordignon.

Without a doubt this experience would have been much harder if not for the support of my family especially my mother for her belief in me and my wife for being on my side. During my studies, I met with many friends, I thank all of them for all the enjoyable moments we spent in LA.

Finally, I would like to thank Kuwait University for supporting my studies and I am forever grateful to them.

VITA

2013	B.S. in Electrical Engineering, Kuwait University, Kuwait.
2013-2014	Data Core Engineer, Huawei Telecommunication, Kuwait
2016	M.S. in Electrical Engineering, University of California, Los Angeles.
2018	Visiting Student EPFL, Switzerland

PUBLICATIONS

- **S. A. Alghunaim** and A. H. Sayed, "Linear convergence of primal-dual gradient methods and their performance in distributed optimization," submitted for publication, 2019.
- **S. A. Alghunaim**, Ernest K. Ryu, K. Yuan, and A. H. Sayed, "Decentralized proximal gradient algorithms with linear convergence rates," conditionally accepted in IEEE Trans. Automatic Control.
- **S. A. Alghunaim**, K. Yuan, and A. H. Sayed, "A proximal diffusion strategy for multi-agent optimization with sparse affine constraints," in IEEE Trans. Automatic Control, 2020, to appear.
- **S. A. Alghunaim** and A. H. Sayed, "Distributed coupled multi-agent stochastic optimization," in IEEE Trans. Automatic Control, vol. 65, no. 1, pp. 175-190, 2020.
- **S. A. Alghunaim**, K. Yuan, and A. H. Sayed, "A linearly convergent proximal gradient algorithm for decentralized optimization," in Advances on Neural Information Processing Systems (NeurIPS), Vancouver, Canada, December 2019.

- **S. A. Alghunaim**, K. Yuan, and A. H. Sayed, "Dual coupled diffusion for distributed optimization with affine constraints," Proc. IEEE CDC, pp. 829-834, Miami Beach, FL, USA, Dec. 2018.

- **S. A. Alghunaim** and A. H. Sayed, "Distributed coupled learning over adaptive networks," Proc. IEEE ICASSP, pp. 6353-6357, Calgary, Canada, April 2018.

- **S. A. Alghunaim**, K. Yuan, and A. H. Sayed, "Decentralized exact coupled optimization, " Proc. Allerton Conference on Communication, Control, and Computing, pp. 338-345, Allerton, IL, October 2017.

CHAPTER 1

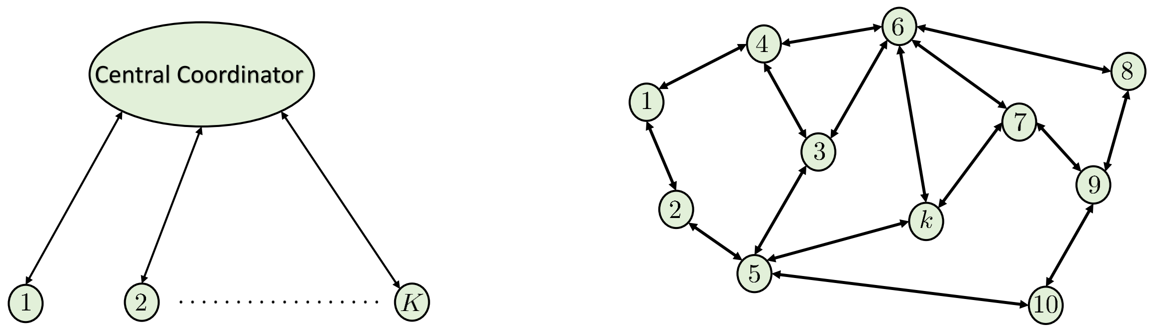
Introduction

In this chapter, we motivate decentralized multi-agent optimization problems and outline the main contributions of the dissertation. We also introduce our notation and review some key concepts.

1.1 Multi-Agent Optimization

Multi-agent optimization refers to optimization problems where more than one agent (e.g., entity, processor, robot, machine) are interested in solving a common optimization problem in a collaborative manner. For example, it is usually intractable or inefficient to solve large scale data problems by processing all the data in one single machine. To relieve the difficulty, one solution is to divide the data across multiple machines and solve the problem in a collaborative manner where each machine is connected to a central coordinator [1–4]. This *distributed* solution method is useful since it allows parallel processing and distributing of the computational load over different machines. However, this approach still requires a centralized network topology with a central node connected to all computing agents [5] – see Figure 1.1a. The potential bottleneck of the centralized network is the communication traffic jam on the central coordinator [6–8]. The performance of these methods can be significantly degraded when the bandwidth around the central node is low. Therefore, for this case it is critical to pursue solutions where there is no central coordinator and each machine only communicates locally with its immediate neighbors. These types of *fully* distributed solutions are often called *decentralized* methods and are the focus of this dissertation.

Decentralized optimization methods can be designed for any connected network topology



(a) Centralized **(b)** Decentralized

Figure 1.1: Centralized vs decentralized networks.

such as line, ring, grid, random geometric graph, or others – see Figure 1.1b. In these structures, there is no central node and each computing agent exchanges information with its immediate neighbors rather than with a remote central server. Decentralized methods have several advantages over non-fully distributed methods with a central coordinator [6, 7]. For example, when there is no central coordinator, the communication can be evenly distributed across the nodes so that decentralized algorithms converge faster than centralized solutions when the network has limited bandwidth or high latency [6, 7]. Apart from relieving communication bottlenecks, decentralized optimization problems naturally occur due to some physical settings such as in coordination and control of robotics systems [9, 10], estimation in wireless sensor networks [11–14], and smart-grids [15]. They are also more robust to failures since a failure of any node in the network is not detrimental compared to the failure of the central node in a centralized network.

1.2 Decentralized Consensus Optimization

In most multi-agent formulations of decentralized optimization problems, each agent generally has an individual cost function, $J_k(w) : \mathbb{R}^M \to \mathbb{R}$, and the goal is to minimize an aggregate sum of the costs, namely,

$$\min_{w \in \mathbb{R}^M} \quad \frac{1}{K} \sum_{k=1}^{K} J_k(w) \tag{1.1}$$

where K is the number of agents. The aggregate cost in (1.1) has one independent variable, $w \in \mathbb{R}^M$, which all agents need to agree upon. Each agent k wants to find the minimizer of (1.1) through local interactions with its direct neighbors. We provide here an overview of the solution methods that are available for solving such problems leading to primal-dual solutions, which are the focus of this work.

Early works on distributed computation and processing include [16–19]. In these early formulations, the agents share a common cost function $J_k(w) = J(w)$, and each agent computes a different block of the variable w to share the computational load among different processors. The *consensus* formulation (1.1) where different agents may share and compute common blocks of the optimization variable w was studied in the works [20, 21]. In these formulations, the agents also share the *same* cost function $J_k(w) = J(w)$, but different from [16–19], the formulation in [20, 21] allows the agents to compute common blocks of w, which require a consensus (agreement) step on the shared blocks. While these early contributions are useful for distributing the computations across different processors, the setups require the local cost to be identical. In general, the agents may have private *local* functions that they do not want to share. In this case, incremental gradient methods [22–25] have been used in [26–28]. However, these methods are not fully distributed (i.e., decentralized), since they either require a central coordinator [26] or require determining a cyclic trajectory that covers all agents in the network in succession, one after the other [27, 28].

Early notable works studying *decentralized* optimization methods include [29–37]. The works [33, 34, 37] focused on deterministic optimization problems where each agent knows exactly its local cost. In comparison, the works [29–32, 35, 36] proposed decentralized algorithms for adaptive learning over networks showing for the first time how adaptation in a stochastic setting can be performed over graphs; thus allowing agents to continually learn and track drifts in the data in the absence of information about the local costs (e.g., statistical distribution of the data) – see [38]. While these early works studied problem (1.1) under different settings, the algorithms studied there belong to the class of *primal* methods. In particular, these early algorithms can be derived from the *primal* domain by solving a penalized approximate problem and not the original problem – see, e.g., [39]. Among these algorithms are consensus (decentralized gradient descent) method and diffusion methods – see [8, 40]. Since these methods solve an approximate problem and not the original problem, they converge to a biased solution for constant step-sizes even under deterministic settings – see [41, 42]. For exact convergence, these early *primal* based methods require using decaying step-sizes to converge to the optimal solution. For highly accurate solutions, decaying step-sizes are undesirable since they slow down the convergence rate significantly.

To overcome the bias caused by these early methods, algorithms with multiple inner consensus steps per iteration were proposed in [43, 44], which require an increasing number of inner consensus steps as the iteration number grows, leading to an expensive solution. Another line of work recognizes that the consensus formulation (1.1) can be rewritten as an equality constrained problem by utilizing the properties of the network [45–49]. Here, unbiased methods are developed by utilizing primal-dual methods to solve the constrained problems [45–49]. Unlike primal methods, primal-dual methods have no bias and can converge to the exact minimizer of (1.1) without extra communication steps per iteration. Due to this attractive feature, many primal-dual algorithms have been proposed [39, 50–56] each with its own advantages. There also exist unbiased gradient-tracking methods [57–65]. These methods correct the bias of the primal methods [37, 66] directly by utilizing gradient tracking mechanisms [67, 68] to track the average of local gradients. Among the gradient tracking methods are the adapt-then-combine (ATC) implementations [57–60, 65], which are motivated on the structure first introduced by diffusion strategies [69] – see also [8, Ch. 7]. For primal methods, the ATC implementations have been shown to have better step-size stability range and performance than the other methods [70]. For the gradient-tracking methods, the ATC variants have only been observed to have better performance than non-ATC variants through numerical simulations [60].

Given the above brief history on decentralized consensus algorithms, it is unclear how all of these methods are related to each others. Moreover, we see that the gradient-tracking methods are not normally classified as primal-dual methods due to their non primal-dual derivations. In the first parts of this dissertation, we focus on the design and analysis of decentralized primal-dual methods. We will design a novel *adapt-then-combine* primal-dual algorithmic framework that captures most state-of-the-art decentralized methods as special cases when the objective is smooth including all the variations of the gradient-tracking methods. We also develop a concise and novel analysis technique that establishes the linear convergence of this general framework under smooth and strongly-convex objectives. Due to our unified framework, the analysis reveals important characteristics of these methods such as their step-size stability range and the influence of the augmented Lagrangian penalty term on the convergence rates of decentralized algorithms. For example, we analytically confirm the observation from [60] that the ATC gradient tracking implementations have better performance than non-ATC implementations.

A long standing open question regarding decentralized optimization problems is whether linear convergence can be established in the presence of non-smooth components. In this dissertation, we answer this question. Specifically, we show that global linear convergence cannot be achieved if each agent owns a *different* local non-smooth term in the worst case. We then show that global linear convergence is possible for a class of novel proximal decentralized algorithms designed for the case where all agents share a *common* non-smooth term. The reader may refer to Section (1.4) for an overview of this dissertation and its main contribution.

1.3 Decentralized Resource Sharing Optimization

While a large portion of this dissertation is focused on the consensus formulation (1.1), we will also study decentralized resource sharing problems. In these formulations, a collection of

K interconnected agents are coupled through an optimization problem of the following form:

$$\underset{w_1,w_2,\cdots,w_K}{\text{minimize}} \quad \sum_{k=1}^{K} J_k(w_k), \text{ subject to } \sum_{k=1}^{K} B_k w_k - b_k = 0, \tag{1.2}$$

where $J_k(w_k)$: $\mathbb{R}^{Q_k} \to \mathbb{R}$ is a cost function associated with agent k and $w_k \in \mathbb{R}^{Q_k}$ is the variable for the same agent. The matrix $B_k \in \mathbb{R}^{S \times Q_k}$ and the vector $b_k \in \mathbb{R}^S$ are known locally by agent k only. In this formulation, each agent wants to find its own minimizer, denoted by $w_k^\star$, while satisfying the global coupling constraint. Note that $\{w_k^\star\}$ are not necessarily consensual and are often different from each other. The resource sharing problem has been studied by different fields and dates back to studies in economics [71–77]. These early works require a central coordinator to solve such problems. The first center-free (i.e., decentralized) algorithm to solve these problems dates back to [78]. Problems of the type (1.2) appear in many other applications such as network utility maximization [79], smart grids [80], basis pursuit [81], and resource allocation in wireless networks [82].

This dissertation considers the case where the formulation involves multiple coupled affine constraints, and where each constraint may involve only a subset of the agents. The constraints are generally sparse, meaning that only a small subset of the agents are involved in them. This scenario arises in many applications including decentralized control formulations, resource allocation problems, and smart grids. Traditional decentralized solutions tend to ignore the structure of the constraints and lead to degraded performance. We instead develop a decentralized solution that exploits the sparsity structure. We examine how the performance of the algorithm is influenced by the sparsity of the constraints. We show analytically, and by means of simulations, the superior convergence properties of an algorithm that considers the sparsity structure in the constraints compared to others that ignore this structure.

1.4 Outline and Contributions

The outline and main contributions of this dissertation are summarized below:

- In the appendix of this chapter, we introduce our notation and review some key concepts that are useful throughout this work.

- In **Chapter** 2, we study a classical incremental primal-dual gradient algorithm for the solution of constrained optimization problems. Through an original proof we establish its linear convergence and give useful step-size and convergence rate upper bounds. We then relate the incremental implementation to the non-incremental Arrow-Hurwicz implementation. The analysis in this chapter will allow us to reveal (in Chapter 3) the influence of the augmented Lagrangian penalty term on the performance of decentralized algorithms. Part of these results are based on the work [83].

- In **Chapter** 3, we study problem (1.1) for smooth and strongly-convex aggregate costs. We propose a general *adapt-then-combine* (ATC) algorithmic framework that captures various state-of-the-art decentralized gradient based algorithms including all the variations of the gradient tracking methods. This result shows that ATC gradient tracking methods admit primal-dual interpretations. We also establish the linear convergence of the proposed framework, thus unifying the analysis of many existing algorithms. This result is important since it reveals the performance and behavior of these various algorithms. For example, we show that the ATC implementations have better performance than non-ATC implementations. Using the analysis from Chapter 2, we will reveal the benefits of the augmented Lagrangian penalty term on the convergence rate of decentralized algorithms. Part of these results are based on the works [83–85].

- In **Chapter** 4, we study problem (1.1) in the presence of a common general non-smooth term. For the first time, we establish the linear convergence of a proximal decentralized gradient based algorithm with a non-smooth term. This result closes the gap between centralized and decentralized proximal gradient algorithms. We then tailor an exiting result to the decentralized set-up where each agent owns a local non-smooth term, and show that global linear convergence cannot be established (in the worst case) for decentralized proximal gradient algorithms for that case. Part of these results are based on the works [84, 85].

- In **Chapters** 5 and 6, we study problems (1.1) and (1.2) under general multiple coupling across the agents. Specifically, we consider scenarios where there can exist multiple consensus variables or multiple coupling constraints with only a subset of agents involved in them. We then show how to design algorithms to exploit these structures. More importantly, we show theoretically that algorithms exploiting the structure can greatly improve the convergence rate compared to algorithms that do not exploit such structure. Part of these results are based on the works [86–90].

- In **Chapter** 7, we state some future directions that can build upon this dissertation.

Appendices

In this dissertation, we shall study several decentralized algorithms for the solution of optimization problems of the form (1.1) or (1.2) by networked agents. For these discussions, we introduce in this section the main notation, the network weights used to implement these algorithms, and review some important concepts, which are useful for our derivations and analysis.

1.A Notation

All vectors are column vectors unless otherwise stated. All norms are 2-norms unless otherwise stated. We use $\text{col}\{x_j\}_{j=1}^N$ to denote a column vector formed by stacking $x_1, ..., x_N$ on top of each other, $\text{diag}\{x_j\}_{j=1}^N$ to denote a diagonal matrix consisting of diagonal entries $x_1, ..., x_N$, and $\text{blkdiag}\{X_j\}_{j=1}^N$ to denote a block diagonal matrix consisting of diagonal blocks $X_1, ..., X_N$. We let $\text{blkrow}\{X_j\}_{j=1}^N = [X_1 \ \cdots \ X_N]$. For any integer set $\mathcal{M} = \{m_1, m_2, \cdots, m_N\}$, we let $U = [g_{mn}]_{m,n\in\mathcal{M}}$ denote the $N \times N$ matrix with $(i,j)-$th entry equal to g_{m_i,m_j}. For a vector $x \in \mathbb{R}^M$, the notation $\|x\|_D^2$ denotes $x^\mathsf{T} D x$ for a positive semi-definite matrix D. Similarly, for a positive constant $c > 0$, we let $\|x\|_c^2$ denote the scaled norm $c\|x\|^2$.

For a matrix $A \in \mathbb{R}^{M\times N}$, $\sigma_{\max}(A)$ denotes the maximum singular value of A, and $\underline{\sigma}(A)$ denotes the minimum *non-zero* singular value. The range space and null space of the matrix A are denoted by $\text{Range}(A)$ and $\text{Null}(A)$, respectively. For two symmetric matrices of similar dimensions, $A \geq B$ means that $A - B$ is positive semi-definite. Likewise, $A > B$ means that $A - B$ is positive definite. The $N \times N$ identity matrix is denoted by I_N. We let $\mathbb{1}_N$ be a vector of size N with all entries equal to one. The Kronecker product is denoted by $\otimes$. $\mathbb{R}$ and $\mathbb{R}^M$ denote the set of real valued numbers and vectors, respectively. The gradient of a differentiable function $f(.) : \mathbb{R}^M \to \mathbb{R}$ is $\nabla f(.) = \text{col}\{\frac{\partial f}{\partial x(1)}, \cdots, \frac{\partial f}{\partial x(M)}\}$ where $x(m)$ is the m-th entry in x and $\frac{\partial f}{\partial x(m)}$ is the derivative of f with respect to the entry $x(m)$. We use $O(\alpha)$ to indicate values on the order of the scalar α (e.g., $O(\alpha) = c\alpha$ for some constant c independent of α).

1.B Network Combination Weights

For any network topology (see, e.g., Fig. 1.1b) we let a_{sk} denote a scalar weight used by agent k to scale information arriving from agent s. We let $a_{sk} = 0$ if s is not a direct neighbor of agent k, i.e., there is no edge connecting them. We let $\mathcal{N}_k$ denote the set of agents the are directly connected to agent k, including agent k itself. We let A denote the combination matrix constructed from these weights:

$$A \triangleq [a_{sk}] \tag{1.3}$$

The matrix A is assumed to be symmetric and doubly stochastic. We also assume A is primitive, i.e., there exists an integer $j > 0$ such that all entries of A^j are positive. As long as the network is connected[1], there exist many rules to choose the weights $\{a_{sk}\}$ in a decentralized fashion – [8, 91]. For example, we can use the Metropolis rule to construct the combinations weights $\{a_{sk};\ s \in \mathcal{N}_k\}$ as follows [8, 92]:

$$a_{sk} = \begin{cases} \dfrac{1}{\max\{n_k, n_s\}}, & \text{if } s \in \mathcal{N}_k,\ s \neq k \\ 1 - \displaystyle\sum_{e \in \mathcal{N}_k \setminus \{k\}} a_{ek}, & s = k, \\ 0, & \text{otherwise.} \end{cases} \tag{1.4}$$

where $n_k = |\mathcal{N}_k|$ denote the number of agents directly connected to agent k. We will utilize the following well established result to derive many decentralized strategies [55, 93].

Lemma 1.1. (Consensus Matrix) *Let $A = [a_{sk}] \in \mathbb{R}^{K \times K}$ be a symmetric doubly stochastic matrix constructed from the network combination weights $\{a_{sk}\}$. Then, it holds that $I - A$ is symmetric and positive semi-definite. Moreover, if we introduce the eigen-decomposition*

$$V \triangleq c(I - A) = U\Sigma U^{\mathsf{T}}, \quad V^{\frac{1}{2}} \triangleq U\Sigma^{1/2}U^{\mathsf{T}}$$

[1] A network is connected if there exists a path with positive weights $\{a_{sk}\}$ connecting any two agents.

for any $c > 0$ and let:

$$\mathcal{A} = A \otimes I_M, \quad \mathcal{V} = V \otimes I_M, \quad \mathcal{V}^{\frac{1}{2}} = V^{\frac{1}{2}} \otimes I_M.$$

Then, for primitive A and any block vector $\mathcal{X} = \text{col}\{x^1, ..., x^K\}$ in the nullspace of $I_{MK} - \mathcal{A}$ with entries $x^k \in \mathbb{R}^M$ it holds that:

$$\mathcal{V}\mathcal{X} = 0 \iff \mathcal{V}^{\frac{1}{2}}\mathcal{X} = 0 \iff (I_{MK} - \mathcal{A})\mathcal{X} = 0 \iff x^1 = x^2 = ... = x^K \tag{1.5}$$

$\square$

1.C Optimization Background

In this section, we briefly review some basic optimization concepts. A set $\mathbb{C}$ is convex if for any two points $x_1, x_2 \in \mathbb{C}$ it holds that

$$\theta x_1 + (1 - \theta x_2) \in \mathbb{C} \text{ for } 0 \leq \theta \leq 1.$$

A function $f(.) : \mathbb{R}^M \to \mathbb{R}$ is convex if its domain $\textbf{dom}f$ is convex and [94]

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y), \quad \text{for all } x, y \in \ \textbf{dom}f$$

with $0 \leq \theta \leq 1$. The function $f(.)$ is strongly-convex with parameter $\nu_f > 0$ if $\textbf{dom}f$ is convex and [94]

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y) - \frac{\nu_f}{2}\theta(1 - \theta)\|x - y\|^2.$$

If $f(.)$ is differentiable then strong-convexity is equivalent to the gradient being strongly-monotone:

$$\big(\nabla f(x) - \nabla f(y)\big)^{\mathsf{T}}(x-y) \geq \nu_f \|x-y\|^2 \tag{1.6}$$

for all $x, y \in \ \mathbf{dom} f$. Equivalently, strong-convexity gives

$$f(y) \geq f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{\nu_f}{2}\|y-x\|^2.$$

The cost f is δ_f-smooth or equivalently the gradient of f is Lipschitz continuous with parameter $\delta_f > 0$ if

$$\|\nabla f(x) - \nabla f(y)\| \leq \delta_f \|x-y\|. \tag{1.7}$$

If f is convex with $\mathbf{dom} f = \mathbb{R}^M$ and ∇f is δ_f-Lipschitz, then it holds that [95, Theorem 2.1.5]

$$\big(\nabla f(x) - \nabla f(y)\big)^{\mathsf{T}}(x-y) \geq \frac{1}{\delta_f}\|\nabla f(x) - \nabla f(y)\|^2. \tag{1.8}$$

The subdifferential $\partial f(x)$ of a function $f(.) : \mathbb{R}^M \rightarrow \mathbb{R}$ at some $x \in \mathbb{R}^M$ is the set of all subgradients:

$$\partial f(x) = \big\{ g_x \mid g_x^{\mathsf{T}}(y-x) \leq f(y) - f(x), \forall \ y \in \mathbb{R}^M \big\}. \tag{1.9}$$

The proximal operator relative to a function $f(x)$ with step-size μ is defined by [96]:

$$\mathbf{prox}_{\mu f}(x) \ \stackrel{\Delta}{=} \ \arg\min_u \left(f(u) + \frac{1}{2\mu}\|x-u\|^2 \right). \tag{1.10}$$

The proximal mapping is firmly nonexpansive [96]:

$$\|\mathbf{prox}_{\mu f}(x) - \mathbf{prox}_{\mu f}(y)\|^2 \leq \big(\mathbf{prox}_{\mu f}(x) - \mathbf{prox}_{\mu f}(y)\big)^{\mathsf{T}}(x-y).$$

Using Cauchy-Schwarz in the previous inequality, it holds that:

$$\|\mathbf{prox}_{\mu f}(x) - \mathbf{prox}_{\mu f}(y)\| \leq \|x - y\|. \tag{1.11}$$

CHAPTER 2

Primal-Dual Gradient Methods

In this chapter, we revisit a discrete incremental implementation of the primal-descent dual-ascent gradient method applied to optimization problems with affine constraints. Through an original short proof, we establish linear convergence of the algorithm for strongly-convex cost functions with Lipschitz continuous gradients. We then relate the classical non-incremental implementation (Arrow-Hurwicz) to the incremental primal-dual implementation and establish its linear convergence as well. The proof technique in this chapter is important to the following chapters where we study primal-dual decentralized algorithms.

2.1 Problem Set-up

We consider the constrained problem:

$$\underset{w\in\mathbb{R}^M}{\text{minimize}} \quad J(w), \quad \text{subject to } Bw = b \tag{2.1}$$

where $w \in \mathbb{R}^M$, $B \in \mathbb{R}^{E\times M}$, and $b \in \mathbb{R}^E$. The cost $J(w) : \mathbb{R}^M \to \mathbb{R}$ is assumed to be δ-smooth and ν-strongly convex – see (1.6)–(1.7). Problem (2.1) can be reformulated into an equivalent saddle-point problem. Indeed, consider the following saddle-point formulation:

$$\min_w \max_\lambda \quad L(w,\lambda) = J(w) + \lambda^{\mathsf{T}}(Bw - b) \tag{2.2}$$

where $\lambda \in \mathbb{R}^E$ is the dual variable. Since $J(w)$ is convex and differentiable, then an optimal point $(w^\star, \lambda^\star)$ exists that solves (2.2); moreover, $w^\star$ is an optimal solution to the constrained problem (2.1) – see Section 2.2. In this chapter, we study the following classical algorithm.

Choose positive step-sizes μ_w and μ_λ and let w_{-1} and λ_{-1} be arbitrary initial conditions. Repeat for $i \geq 0$:

$$\begin{cases} w_i = w_{i-1} - \mu_w\big(\nabla J(w_{i-1}) + B^\mathsf{T}\lambda_{i-1}\big) & \text{(2.3a)} \\ \lambda_i = \lambda_{i-1} + \mu_\lambda(Bw_i - b) & \text{(2.3b)} \end{cases}$$

Note that the update (2.3) is *incremental* since the dual update (2.3b) uses the most recent primal variable w_i and not w_{i-1}. This chapter focuses on the primal-dual (PD) algorithm (2.3), which is aimed at solving (2.2), establishes its linear convergence properties and studies its relation to the non-incremental implementation:

$$\begin{cases} w_i = w_{i-1} - \mu_w\big(\nabla J(w_{i-1}) + B^\mathsf{T}\lambda'_{i-1}\big) & \text{(2.4a)} \\ \lambda'_i = \lambda'_{i-1} + \mu_\lambda(Bw_{i-1} - b) & \text{(2.4b)} \end{cases}$$

Note that algorithms (2.3) and (2.4) are applied to the Lagrangian, and not the augmented Lagrangian (AL) (which would an extra quadratic penalty term $\rho\|Bw - b\|^2$ added to the Lagrangian (2.2) where $\rho > 0$). Since we can absorb the quadratic term into a new cost $J_\rho(w) \triangleq J(w) + \rho\|Bw - b\|^2$, our convergence analysis is still applicable for that case.

Algorithms of the form (2.3) and (2.4) have been applied in various scenarios including but not limited to wireless systems [97,98], power systems [99], reinforcement learning [100,101], and network utility maximization [48,102].

2.1.1 Related Works

There exists a large body of literature on primal-dual saddle-point algorithms – see [48,102–109] and the references therein, including the seminal work [104], which proposed the classical recursion (2.4) and established its convergence. These works focus on proving convergence to an optimal solution without providing convergence rates, provide sub-linear convergence rates (e.g., $\frac{1}{i}$ where i is the iteration index), or show linear convergence from a starting point that is sufficiently close to a solution (local convergence). Some other works examined linear convergence under different settings.

The work [110] focuses on the *continuous* version of the primal-dual gradient dynamics and establishes linear convergence but using the *augmented* Lagrangian (AL). The work [111] proves linear convergence for a *continuous* primal-dual gradient dynamics on a smoothed AL called "proximal augmented Lagrangian", which can handle a non-smooth term. The work [112] focuses on the continuous version of the primal-dual dynamics for problem (2.1) with additional affine inequality constraints and establishes exponential convergence under the assumption that $J(w)$ is twice-differentiable with upper and lower bounded Hessian (i.e., strongly convex and smooth in addition to twice differentiability). It was shown in [112] that if the continuous dynamics is discretized using Euler discretization, then the discrete version converges linearly under small enough step sizes. However, no upper bound is given on the step-size. Moreover, the derived linear convergence bound depends on the continuous dynamics bounds. Note that Euler discretization uses identical step-sizes for the primal and dual updates (i.e., $\mu_w = \mu_\lambda$) and results in a *non-incremental* primal-dual dynamics, i.e., the dual update does not use the most recent primal update. Thus, the results in [110–112] are not directly applicable to incremental discrete implementation and/or do not provide useful bounds on the step-size and convergence rates.

The work [113] establishes the linear convergence of a primal-dual gradient algorithm for saddle-point problems with

$$L(w, \lambda) = J(w) + \lambda^{\mathsf{T}} B w - g(\lambda) \tag{2.5}$$

where $J(w)$ is convex and smooth, and $g(\lambda)$ is strongly-convex and smooth. Unlike the current chapter, the algorithm used in [113] is *non-incremental*; moreover, a particular fixed step-sizes are needed to establish linear convergence – [113, Theorem 3.1].

We remark that the works [114–117] established the linear convergence of operator-splitting based algorithms, which solve a general saddle point problem with

$$L(w, \lambda) = J(w) + \lambda^{\mathsf{T}} B w - g(\lambda) \tag{2.6}$$

where $g(.)$ is not necessarily differentiable. However, they require *both* the primal and dual functions, $J(.)$ and $g(.)$, to be strongly-convex functions. When either is missing, the matrix B plays a critical role for linear convergence, which was not the focus of these works. Note that in this chapter we focus on the linear convergence of the classical methods (2.3) and (2.4) that are different from the operator splitting methods [114–119].

2.1.2 Contribution

Given the above, our contribution in this chapter is twofold: **1)** Through an original self-contained short proof, we establish the linear convergence of the *incremental* implementation (2.3). Our analysis holds with or without the augmented Lagrangian penalty term; **2)** We relate the non-incremental implementation (2.4) to the incremental one (2.3) and show that its linear convergence follows from the analysis of the incremental one. We provide explicit upper bounds on the step-size parameters for stable behavior and on the resulting convergence rate

2.2 Convergence Results

This section gives the auxiliary results leading to the main convergence result. It is known that a pair $(w^\star, \lambda^\star)$ is an optimal solution to (2.2) if, and only if, it satisfies the optimality conditions [94]:

$$\begin{cases} 0 = \nabla J(w^\star) + B^{\mathsf{T}} \lambda^\star & \text{(2.7a)} \\ 0 = Bw^\star - b & \text{(2.7b)} \end{cases}$$

Note that $w^\star$ coincides with the minimizer of (2.1). To see that $w^\star$ must be an optimal solution to (2.1), we follow arguments similar to the ones in [94, Section 4.2.3]. Thus, consider the optimality criterion of (2.1) for differentiable $J(w)$ [94]:

$$\nabla J(w^\star)(z - w^\star) \geq 0 \quad \text{for all } z \text{ such that } Bz = b \tag{2.8}$$

Since $B(w^\star - z) = b - b = 0$, the above condition implies that:

$$\nabla J(w^\star)^\mathsf{T} x \geq 0 \tag{2.9}$$

for all x belonging to the null space of B (i.e., $Bx = 0$). If a linear function is nonnegative on a subspace, then it must be zero on the subspace [94]. Thus, the gradient $\nabla J(w^\star)$ is orthogonal to the null space of B, i.e., $\nabla J(w^\star)^\mathsf{T} x = 0$ for all x belonging to the null space of B. Since the range of B^T is orthogonal to the null space of B, it follows that $\nabla J(w^\star)$ belongs to the range of B^T [120]. This implies that condition (2.7a) holds for some $\lambda^\star$. Hence $w^\star$ solves (2.1) if, and only if, the optimality conditions (2.7) holds. This means that problems (2.1) and (2.2) are equivalent. Note that since $J(w)$ is strongly convex, $w^\star$ is unique –see [94, Example 5.4].

From (2.7a) and uniqueness of $w^\star$, $\lambda^\star$ will be unique if B has full row rank. In general $\lambda^\star$ is not necessarily unique. We will now characterize a particular dual solution that we later show convergence to. For that result and later analysis, we need the following result.

Lemma 2.1. *If λ_x is in the range space of $B \in \mathbb{R}^{E \times M}$, then it holds that:*

$$\|B^\mathsf{T} \lambda_x\|^2 \geq \underline{\sigma}^2(B) \|\lambda_x\|^2 \tag{2.10}$$

where $\underline{\sigma}(B)$ denotes the minimum non-zero singular value of B.

Proof. Introduce the truncated singular value decomposition [120] of the positive semi-definite matrix $B^\mathsf{T} B = U_r \Sigma_r U_r^\mathsf{T}$, where $U_r \in \mathbb{R}^{M \times r}$ (r denotes the rank of $B^\mathsf{T} B$) with $U_r^\mathsf{T} U_r = I_r$ and $\Sigma_r > 0$ is a diagonal matrix with entries equal to the non-zero eigenvalues of $B^\mathsf{T} B$ (i.e., the squared non-zero singular values of B). Since λ_x is in the range space of B, it holds that $\lambda_x = Bx$ for some x. Then,

$$\begin{aligned} \|B^\mathsf{T} \lambda_x\|^2 = \|B^\mathsf{T} B x\|^2 = x^\mathsf{T} U_r \Sigma_r^2 U_r^\mathsf{T} x \geq \underline{\sigma}^2(B) x^\mathsf{T} U_r \Sigma_r U_r^\mathsf{T} x \\ = \underline{\sigma}^2(B) \|\lambda_x\|^2. \end{aligned} \tag{2.11}$$

$\square$

Lemma 2.2. (Particular dual $\lambda_b^\star$) *There exists a unique optimal dual variable, denoted by $\lambda_b^\star$, lying in the range space of B.*

Proof. Any solution $\lambda^\star$ of the linear system of equations given in (2.7a) can be decomposed into two parts $\lambda^\star = \lambda_b^\star + \lambda_n^\star$, where $\lambda_b^\star \in \text{Range}(B)$ and $\lambda_n^\star \in \text{Null}(B^\mathsf{T})$ – see [120]. Therefore, if $(w^\star, \lambda^\star)$ satisfies (2.7), then $(w^\star, \lambda_b^\star)$ also satisfies (2.7). We now show $\lambda_b^\star$ is unique by contradiction. Assume we have two *distinct* dual solutions $\lambda_{b_1}^\star = Bx_1$ and $\lambda_{b_2}^\star = Bx_2$ lying in the range space of B. Then, substituting into (2.7a) and subtracting, we get $B^\mathsf{T}B(x_1 - x_2) = 0$. It follows that $\|B(x_1 - x_2)\|^2 = 0$ and, consequently, $B(x_1 - x_2) = 0$. This means that $\lambda_{b_1}^\star = Bx_1 = Bx_2 = \lambda_{b_2}^\star$, which is a contradiction.

□

From (2.3b) we know that

$$\lambda_i = \lambda_{i-1} + \mu_\lambda(Bw_i - b).$$

Therefore, from the fact that $b = Bw^\star$, λ_i will be in the range space of B if λ_{i-1} belongs to the range space of B or $\lambda_{i-1} = 0$. Thus, $\{\lambda_i\}_{i\geq 0}$ will always remain in the range space of B if $\lambda_{-1} = 0$ or λ_{-1} belongs to the range space of B. This observation will allow us to utilize the bound (2.11) to establish linear convergence to the particular saddle-point $(w^\star, \lambda_b^\star)$ without requiring a rank condition on the matrix B. For analysis, we let

$$\widetilde{w}_i \triangleq w_i - w^\star \quad \text{and} \quad \widetilde{\lambda}_i \triangleq \lambda_i - \lambda_b^\star$$

denote the primal and dual errors, respectively. We are now ready to establish our main result whose proof is given in Appendix 2.A.

Theorem 2.1. (Linear convergence)*: Assume that the cost $J(w)$ is δ-smooth and ν-strongly-convex and the step-sizes are positive and satisfy:*

$$\mu_w < \frac{1}{\delta}, \quad \mu_\lambda \leq \frac{\nu}{\sigma^2_{\max}(B)} \tag{2.12}$$

If $\lambda_{-1} = Bw_{-1}$ *(or* $\lambda_{-1} = 0$*), then algorithm* (2.3) *converges linearly to the particular saddle-point* $(w^\star, \lambda_b^\star)$*, namely, it holds that*

$$\|\widetilde{w}_i\|^2_{c_w} + \|\widetilde{\lambda}_i\|^2_{c_\lambda} \leq \gamma\big(\|\widetilde{w}_{i-1}\|^2_{c_w} + \|\widetilde{\lambda}_{i-1}\|^2_{c_\lambda}\big) \tag{2.13}$$

where $c_\lambda > 0$*,* $c_w = 1 - \mu_w\mu_\lambda\sigma^2_{\max}(B) > 0$*, and*

$$\gamma \;\stackrel{\Delta}{=}\; \max\left\{1 - \mu_w\nu(1-\mu_w\delta), 1 - \mu_w\mu_\lambda\underline{\sigma}^2(B)\right\} < 1$$

$\square$

Theorem 2.1 guarantees linear convergence for any step-sizes satisfying (2.12). Moreover, the convergence rate is upper bounded by γ, which clearly shows the effect of B on the convergence rate. We remark that in the analysis of Theorem 2.1 , we did not utilize any augmented penalty term $\rho\|Bw - b\|^2$, which is used in augmented Lagrangian formulations. The presence of such term can allow us to remove the bound on μ_λ given in (2.12) and only require $\mu_w\mu_\lambda < 1/\sigma^2_{\max}(B)$. We will use this result in the following chapter to see the how the absence of such penalty term affects the performance of decentralized algorithms.

2.3 Non-Incremental Implementation

In algorithm (2.3), the dual update uses the most recent primal estimate w_i and not w_{i-1}. A more classical implementation updates both primal and dual variables based on the previous iterates as listed in (2.4). The following results relates the non-incremental implementation to the incremental implementation.

Lemma 2.3. *Consider the following cost:*

$$J'(w) \;\stackrel{\Delta}{=}\; J(w) - \frac{\mu_\lambda}{2}\|Bw - b\|^2 \tag{2.14}$$

Then, the primal iterates of the non-incremental implementation (2.4) *are equivalent to primal iterates of incremental recursion* (2.3) *with cost* $J'(w)$ *instead of* $J(w)$.

Proof. Applying the incremental implementation (2.3) using cost $J'(w)$, we get:

$$\begin{cases} w_i = w_{i-1} - \mu_w\big(\nabla J(w_{i-1}) + B^\mathsf{T}[\lambda_{i-1} - \mu_\lambda(Bw_{i-1} - b)]\big) & (2.15a) \\ \lambda_i = \lambda_{i-1} + \mu_\lambda(Bw_i - b) & (2.15b) \end{cases}$$

By introducing the change of variable $\lambda'_i = \lambda_i - \mu_\lambda(Bw_i - b)$, we can rewrite the previous recursion as in (2.4). Thus, the primal iterates of recursion (2.4) are equivalent to the primal iterates of recursion (2.15) if $\lambda'_{-1} = \lambda_{-1} - \mu_\lambda(Bw_{-1} - b)$. □

A direct consequence of the previous lemma and Theorem 2.1 is the following corollary.

Corollary 2.1. *Assume that the cost* $J(w)$ *is* δ*-smooth and* ν*-strongly-convex and the step-sizes satisfy:*

$$\mu_w < \frac{1}{\delta + \mu_\lambda \sigma^2_{\max}(B)}, \quad \mu_\lambda \leq \frac{\nu}{2\sigma^2_{\max}(B)} \tag{2.16}$$

Then, recursion (2.4) *converges linearly to the particular saddle-point* $(w^\star, \lambda^\star_b)$ *if* $\lambda'_{-1} = 0$. □

The proof of the previous result is given in Appendix 2.B.

Appendices

2.A Proof of Theorem 2.1

Proof of Theorem 2.1. Subtracting $w^\star$ and $\lambda^\star_b$ from both sides of (2.3) and using the optimality conditions (2.7) we get the coupled error recursion:

$$\widetilde{w}_i = \widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star) + B^\mathsf{T}\widetilde{\lambda}_{i-1}\big) \tag{2.17a}$$

$$\widetilde{\lambda}_i = \widetilde{\lambda}_{i-1} + \mu_\lambda B\widetilde{w}_i \tag{2.17b}$$

Squaring both sides of (2.17a) and (2.17b) we get

$$\begin{aligned}\|\widetilde{w}_i\|^2 &= \|\widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\|^2 + \mu_w^2\|B^\mathsf{T}\widetilde{\lambda}_{i-1}\|^2 \\ &\quad - 2\mu_w\widetilde{\lambda}_{i-1}^\mathsf{T}B\left(\widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\right)\end{aligned} \tag{2.18}$$

and

$$\begin{aligned}\|\widetilde{\lambda}_i\|^2 &= \|\widetilde{\lambda}_{i-1}\|^2 + \mu_\lambda^2\|B\widetilde{w}_i\|^2 + 2\mu_\lambda\widetilde{\lambda}_{i-1}^\mathsf{T}B\widetilde{w}_i \\ &\overset{(2.17a)}{=} \|\widetilde{\lambda}_{i-1}\|^2 + \mu_\lambda^2\|B\widetilde{w}_i\|^2 - 2\mu_\lambda\mu_w\|B^\mathsf{T}\widetilde{\lambda}_{i-1}\|^2 \\ &\quad + 2\mu_\lambda\widetilde{\lambda}_{i-1}^\mathsf{T}B\left(\widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\right)\end{aligned} \tag{2.19}$$

Using the bound $\|B\widetilde{w}_i\|^2 \leq \sigma_{\max}^2(B)\|\widetilde{w}_i\|^2$, multiplying equation (2.19) by $c_\lambda \overset{\Delta}{=} \mu_w/\mu_\lambda$ and adding to (2.18) gives:

$$\|\widetilde{w}_i\|_{c_w}^2 + \|\widetilde{\lambda}_i\|_{c_\lambda}^2 \leq \|\widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\|^2 + \|\widetilde{\lambda}_{i-1}\|_{c_\lambda}^2 - \mu_w^2\|B^\mathsf{T}\widetilde{\lambda}_{i-1}\|^2 \tag{2.20}$$

where $c_w \overset{\Delta}{=} 1 - \mu_w\mu_\lambda\sigma_{\max}^2(B)$. Note that from Lemma 2.2, $\lambda_b^\star$ lies in the range space of B. Moreover, since $\lambda_{-1} = 0$, then we know that $\widetilde{\lambda}_i$ will always lie in the range space of B. Thus, from (2.10) it holds that $\|B^\mathsf{T}\widetilde{\lambda}_{i-1}\|^2 \geq \underline{\sigma}^2(B)\|\widetilde{\lambda}_{i-1}\|^2$. Using this bound in (2.20), we get:

$$\|\widetilde{w}_i\|_{c_w}^2 + \|\widetilde{\lambda}_i\|_{c_\lambda}^2 \leq \|\widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\|^2 + (1 - \mu_w\mu_\lambda\underline{\sigma}^2(B))\|\widetilde{\lambda}_{i-1}\|_{c_\lambda}^2 \tag{2.21}$$

Since $J(w)$ is δ-smooth, we know from (1.8):

$$\|\nabla J(w_{i-1}) - \nabla J(w^\star)\|^2 \leq \delta\widetilde{w}_{i-1}^\mathsf{T}\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big) \tag{2.22}$$

Using this bound, it holds that

$$\begin{aligned}
&\|\widetilde{w}_{i-1} - \mu_w\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\|^2 \\
&= \|\widetilde{w}_{i-1}\|^2 - 2\mu_w \widetilde{w}_{i-1}^{\mathsf{T}}\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big) + \mu_w^2\|\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big)\|^2 \\
&\overset{(2.22)}{\leq} \|\widetilde{w}_{i-1}\|^2 - \mu_w(2-\delta\mu_w)\widetilde{w}_{i-1}^{\mathsf{T}}\big(\nabla J(w_{i-1}) - \nabla J(w^\star)\big) \\
&\leq \big(1 - \mu_w\nu(2-\mu_w\delta)\big)\|\widetilde{w}_{i-1}\|^2
\end{aligned}$$

where we used the strong-convexity bound (1.6) in the last step. Let $\gamma_1 = 1 - \mu_w\nu(1-\mu_w\delta)$. Since $c_w \overset{\Delta}{=} 1 - \mu_w\mu_\lambda\sigma^2_{\max}(B)$, it holds that:

$$\begin{aligned}
\big(1 - \mu_w\nu(2-\mu_w\delta)\big)\|\widetilde{w}_{i-1}\|^2 &= \gamma_1\|\widetilde{w}_{i-1}\|^2 - \mu_w\nu\|\widetilde{w}_{i-1}\|^2 \\
&= \gamma_1\|\widetilde{w}_{i-1}\|^2_{c_w} - \mu_w(\nu - \mu_\lambda\sigma^2_{\max}(B)\gamma_1)\|\widetilde{w}_{i-1}\|^2 \\
&\leq \big(1 - \mu_w\nu(1-\mu_w\delta)\big)\|\widetilde{w}_{i-1}\|^2_{c_w} \qquad (2.23)
\end{aligned}$$

where the last step we used the fact that the last term is non-positive under the conditions $\mu_w < \frac{1}{\delta}$ and $\mu_\lambda \leq \nu/\sigma^2_{\max}(B)$. Using the previous two equations in (2.21), we get:

$$\|\widetilde{w}_i\|^2_{c_w} + \|\widetilde{\lambda}_i\|^2_{c_\lambda} \leq \big(1 - \mu_w\nu(1-\mu_w\delta)\big)\|\widetilde{w}_{i-1}\|^2_{c_w} + (1 - \mu_w\mu_\lambda\underline{\sigma}^2(B))\|\widetilde{\lambda}_{i-1}\|^2_{c_\lambda} \qquad (2.24)$$

Equation (2.24) is exactly (2.13). Note that for positive step-sizes it holds that $c_\lambda = \frac{\mu_w}{\mu_\lambda} > 0$. Moreover, $c_w = 1 - \mu_w\mu_\lambda\sigma^2_{\max}(B) > 0$ and $0 < \gamma_2 = 1 - \mu_w\mu_\lambda\underline{\sigma}^2(B) < 1$ if $\mu_w\mu_\lambda < \frac{1}{\sigma^2_{\max}(B)}$. This condition is satisfied under condition (2.12) because under these conditions we have

$$\mu_w\mu_\lambda < \frac{\nu}{\delta\sigma^2_{\max}(B)} \leq \frac{1}{\sigma^2_{\max}(B)}$$

where the last inequality hold because $\nu \leq \delta$. $\square$

2.B Proof of Corollary 2.1

Proof of Corollary 2.1. We know from Lemma 2.3 that the analysis of (2.3) follows directly from Theorem 2.1 with cost $J'(w) = J(w) - \frac{\mu_\lambda}{2}\|Bw - b\|^2$ instead of $J(w)$. Hence, we only need to verify that $J'(w)$ is smooth and strongly-convex for small enough μ_λ. Using the triangle inequality and Lipschitz property of the gradient (1.7) it can be easily verified that:

$$\begin{aligned}\|\nabla J'(w_1) - \nabla J'(w_2)\| &= \|\nabla J(w_1) - \nabla J(w_2) - \mu_\lambda B^{\mathsf{T}} B(w_1 - w_2)\| \\ &\le \big(\delta + \mu_\lambda \sigma^2_{\max}(B)\big)\|w_1 - w_2\| \qquad (2.25)\end{aligned}$$

Moreover, from the strong-convexity property (1.6) it also holds that

$$\begin{aligned}\big(\nabla J'(w_1) - \nabla J'(w_2)\big)^{\mathsf{T}}(w_1 - w_2) &= \big(\nabla J(w_1) - \nabla J(w_2)\big)^{\mathsf{T}}(w_1 - w_2) - \mu_\lambda\|B(w_1 - w_2)\|^2 \\ &\ge \nu\|w_1 - w_2\|^2 - \mu_\lambda\|B(w_1 - w_2)\|^2 \\ &\ge \big(\nu - \mu_\lambda\sigma^2_{\max}(B)\big)\ \|w_1 - w_2\|^2 \qquad (2.26)\end{aligned}$$

Therefore, the cost $J'(w)$ is $\delta + \mu_\lambda\sigma^2_{\max}(B)$ smooth and $\nu - \mu_\lambda\sigma^2_{\max}(B) > 0$ strongly-convex if $\mu_\lambda < \nu/\sigma^2_{\max}(B)$. Using these Lipschitz and strong-convexity constants in (2.12) we get conditions (2.16). □

CHAPTER 3

Decentralized Primal-Dual Algorithms

In this chapter, we study the decentralized optimization problem (1.1) where a network of agents are interested in minimizing a sum of local cost functions. We propose a general algorithmic framework for the solution of such problems that captures various existing state-of-the-art algorithms. We then establish the linear convergence of this general algorithm when the aggregate cost function is strongly-convex. Our unified analysis reveals interesting facts about the performance of these various algorithms such as the advantages of the adapt-then-combine variants over the other variants. Finally, we study the effect of the augmented Lagrangian penalty term on the stability and performance of decentralized algorithms.

3.1 Decentralized Optimization Set-up

Consider a network of K agents that are connected through a static and undirected network. Through only local interactions (i.e., with agents only communicating with their immediate neighbors), each node is interested in finding a solution to the following problem:

$$w^\star = \arg\min_{w \in \mathbb{R}^M} \quad \bar{J}(w) \;\overset{\Delta}{=}\; \frac{1}{K}\sum_{k=1}^{K} J_k(w) \tag{3.1}$$

where $J_k(w) : \mathbb{R}^M \rightarrow \mathbb{R}$ is a local cost function associated with agent k. We adopt the following assumption throughout this chapter.

Assumption 3.1. (Cost function): *Each cost function $J_k(w)$ is convex with δ-Lipschitz continuous gradients:*

$$\|\nabla J_k(w^o) - \nabla J_k(w^\bullet)\| \leq \delta \|w^o - w^\bullet\| \tag{3.2}$$

for any w^o and $w^\bullet$ and some $\delta > 0$. Moreover, the aggregate cost $\bar{J}(w)$ is $\bar{\nu}$-strongly-convex:

$$(w^o - w^\bullet)^\mathsf{T}\big(\nabla \bar{J}(w^o) - \nabla \bar{J}(w^\bullet)\big) \geq \bar{\nu}\|w^o - w^\bullet\|^2 \tag{3.3}$$

for any w^o and $w^\bullet$ and constant $0 < \bar{\nu} \leq \delta$. $\square$

Note that from the strong-convexity condition (3.3), it holds that the global solution $w^\star$ is unique.

3.1.1 Related Works

Various gradient based algorithms have been proposed to solve decentralized optimization problems of the form (3.1) – see [39, 53–65]. Only few works have attempted to unify some of these algorithms [121–123]. For example, the work [122] proposed a general method that includes EXTRA [55] and DIGing [61, 62] (for static and undirected network) as special cases. However, the method in [122] does not include the adapt-then-combine[1] (ATC) gradient-tracking algorithms [57, 59, 60]. The work [121] proposed a canonical form that captures decentralized algorithms that require a single round of communication and gradient computation per iteration, which does not include the Aug-DGM (ATC-DIGing) [59, 60]. Reference [121] only focused on the canonical form without providing any analysis for this form. The work [123] studied a special class of the algorithms in [121] and provided worst case linear convergence rates through numerical solution of semidefinite programs. This special class does not include algorithms that require communicating two different vectors per iteration such as gradient tracking methods [57–62].

[1]The Adapt-then-Combine (ATC) structure was first proposed in [69] to distinguish between different implementations of diffusion learning strategies – see also [8, Ch. 7].

Different from existing works we propose an adapt-then-combine (ATC) primal-dual framework that captures more algorithms including EXTRA [55], Exact diffusion [39], NIDS [56], and different variation of the gradient tracking methods [57–61] including Aug-DGM [59]. Our framework is the first to show that the ATC gradient-tracking methods can be represented as primal-dual recursions. We then establish the linear convergence of this framework under Assumption 3.1. The linear convergence result clarifies the performance of these various algorithms with respect to each other. In particular, we show that the ATC implementations have larger step-size stability range than non-ATC implementations.

We will see that most state-of-the-art methods are based on *augmented Lagrangian* (AL) derivations. It was found in [124] that unlike AL methods, Lagrangian based methods (without AL penalty term) suffer from stability issues when the individual costs are not strongly-convex. However, it is unclear what the benefit of the AL penalty term is if the individual costs are strongly-convex. Due to our unified Lagrangian and AL proof technique from the previous chapter, we clarify the effect of the AL penalty term on the convergence rates of decentralized algorithms.

3.1.2 Contribution

Given the above, this chapter has three contributions: **I)** we propose an adapt-then-combine (ATC) unifying primal-dual framework that covers many existing state-of-the-art algorithms [39, 55–62]. To our knowledge, this is the first primal-dual interpretation of the ATC gradient tracking methods [57–60]. **II)** we provide a unifying linear convergence analysis for strongly-convex *aggregate* cost functions. Our step-size and convergence rate upper bounds shed light on the stability and performance of these various methods. Most notably, we show that the ATC implementations are more stable than non-ATC implementations. **III)** we show the effect of the augmented penalty term on the convergence rate of decentralized algorithms. It is found that the penalty term can greatly improve the convergence rate when the individual costs are ill-conditioned but the aggregate cost is well conditioned.

3.2 Adapt-then-Combine Framework

In this section, we present an adapt-then-combine (ATC) algorithmic framework that covers various state-of-the-art algorithms as special cases.

3.2.1 General Primal-Dual Algorithm

For algorithm derivation and motivation purposes, we will rewrite problem (3.1) in an equivalent manner. To do that, we let $w_k \in \mathbb{R}^M$ denote a local copy of w available at agent k and introduce the network quantities:

$$\mathcal{W} \triangleq \mathrm{col}\{w_1, \cdots, w_K\} \in \mathbb{R}^{MK} \tag{3.4}$$

$$\mathcal{J}(\mathcal{W}) \triangleq \sum_{k=1}^{K} J_k(w_k) \tag{3.5}$$

Further, we introduce two general symmetric matrices $\mathcal{B} \in \mathbb{R}^{MK \times MK}$ and $\mathcal{C} \in \mathbb{R}^{MK \times MK}$ that satisfy the following conditions:

$$\begin{cases} \mathcal{B}\mathcal{W} = 0 \iff w_1 = \cdots = w_K & \text{(3.6a)} \\ \mathcal{C}\mathcal{W} = 0 \iff \mathcal{B}\mathcal{W} = 0 \quad \text{or} \quad \mathcal{C} = 0 & \text{(3.6b)} \end{cases}$$

For algorithm derivation, the matrices $\{\mathcal{B}, \mathcal{C}\}$ can be any general symmetric consensus matrices. Later, we will see how to specifically choose these matrices to get different decentralized implementations – see Section 3.2.3. With these quantities, it is easy to see that problem (3.1) is equivalent to the following problem:

$$\underset{\mathcal{W} \in \mathbb{R}^{MK}}{\text{minimize}} \quad \mathcal{J}(\mathcal{W}) + \frac{\rho}{2}\|\mathcal{W}\|_{\mathcal{C}}^2, \quad \text{s.t. } \mathcal{B}\mathcal{W} = 0 \tag{3.7}$$

where $\rho \geq 0$, and the matrix $\mathcal{C} \in \mathbb{R}^{MK \times MK}$ is a positive semi-definite consensus penalty matrix. To solve problem (3.7), we consider the saddle-point formulation:

$$\min_{\mathcal{W}} \max_{\mathcal{Y}} \quad \mathcal{L}(\mathcal{W}, \mathcal{Y}) \triangleq \mathcal{J}(\mathcal{W}) + \frac{1}{\mu}\mathcal{Y}^{\mathsf{T}}\mathcal{B}\mathcal{W} + \frac{\rho}{2}\|\mathcal{W}\|^2_{\mathcal{C}} \tag{3.8}$$

where $\mu > 0$ and $\mathcal{Y} \in \mathbb{R}^{MK}$ is the dual variable. To solve (3.8), we propose the following algorithm: let $\mathcal{W}_{-1}$ take any arbitrary value and $\mathcal{Y}_{-1} = 0$ or $\mathcal{Y}_{-1} \in \text{Range}(\mathcal{B})$. Repeat for $i = 0, 1, \cdots$

$$\begin{cases} \mathcal{Z}_i = (I - \mathcal{C})\mathcal{W}_{i-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \mathcal{B}\mathcal{Y}_{i-1} & \textbf{(primal-descent)} \quad (3.9a) \\ \mathcal{Y}_i = \mathcal{Y}_{i-1} + \mathcal{B}\mathcal{Z}_i & \textbf{(dual-ascent)} \quad (3.9b) \\ \mathcal{W}_i = \bar{\mathcal{A}}\mathcal{Z}_i & \textbf{(Combine)} \quad (3.9c) \end{cases}$$

In the above algorithm, $\bar{\mathcal{A}} = \bar{A} \otimes I_M$ where $\bar{A}$ is a symmetric and doubly-stochastic combination matrix. Step (3.9a) is a gradient descent followed by a gradient ascent step in (3.9b), both applied to the saddle-point problem (3.8) with step-size μ and $\rho = \frac{1}{\mu}$. The last step (3.9c) is a combination step that enforces further agreement. Next we show that by proper choices of $\bar{\mathcal{A}}$, $\mathcal{B}$, and $\mathcal{C}$ we can recover many state of the art algorithms. To do that, we need to introduce the combination matrix associated with the network.

3.2.2 Network Combination Matrix

Recall from Section 1.B, that the network is associated with the matrix $A = [a_{sk}] \in \mathbb{R}^{K \times K}$ where the weight $a_{sk} = 0$ if there is no edge connecting agents k and s. Using this matrix, we introduce the augmented network combination matrix

$$\mathcal{A} = A \otimes I_M \tag{3.10}$$

Recall from Section 1.B that the matrix A (different from $\bar{A}$) is assumed to be symmetric, doubly stochastic matrix , and primitive. Under these conditions it holds that $(I_{MK} - \mathcal{A})\mathcal{W} = 0$

if, and only, if $w_k = w_s$ for all k, s — see Lemma 1.1. Note that

$$\mathcal{A}w = \begin{bmatrix} \sum\limits_{s\in\mathcal{N}_1} a_{s1} w_s \\ \vdots \\ \sum\limits_{s\in\mathcal{N}_K} a_{sK} w_s \end{bmatrix}$$

Hence, the k-th block $\sum\limits_{s\in\mathcal{N}_k} a_{sk} w_s$ can be computed by agent k through communicating with its neighbors $\mathcal{N}_k$ only. Therefore, by choosing the matrices $\{\bar{\mathcal{A}}, \mathcal{B}, \mathcal{C}\}$ from $\mathcal{A}$ we can recover different decentralized algorithms as we show next.

3.2.3 Specific Decentralized Algorithms

We start by rewriting recursion (3.9) in an equivalent manner by eliminating the dual variable $\mathcal{Y}_i$. Thus, from (3.9a) it holds that

$$\begin{aligned} \mathcal{Z}_i - \mathcal{Z}_{i-1} &= (I-\mathcal{C})(\mathcal{W}_{i-1} - \mathcal{W}_{i-2}) - \mathcal{B}(\mathcal{Y}_{i-1} - \mathcal{Y}_{i-2}) - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big) \\ &\overset{(3.9b)}{=} (I-\mathcal{C})(\mathcal{W}_{i-1} - \mathcal{W}_{i-2}) - \mathcal{B}^2\mathcal{Z}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big) \end{aligned}$$

for $i \geq 1$ and initialization can be set to $\mathcal{Z}_0 = (I-\mathcal{C})\mathcal{W}_{-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{-1})$ and $\mathcal{W}_0 = \bar{\mathcal{A}}\mathcal{Z}_0$ with arbitrary $\mathcal{W}_{-1}$. Rearranging the previous equation we can rewrite (3.9) equivalently as:

$$\mathcal{Z}_i = (I-\mathcal{B}^2)\mathcal{Z}_{i-1} + (I-\mathcal{C})(\mathcal{W}_{i-1} - \mathcal{W}_{i-2}) - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big) \tag{3.11a}$$

$$\mathcal{W}_i = \bar{\mathcal{A}}\mathcal{Z}_i \tag{3.11b}$$

for $i \geq 1$. Utilizing the above recursion, we will now choose specific matrices $\{\bar{\mathcal{A}}, \mathcal{B}, \mathcal{C}\}$ and show that we can recover many state of the art algorithms:

- **(DIGing [61, 62])**: If $\mathcal{B}^2 = (I-\mathcal{A})^2$, $\mathcal{C} = I - \mathcal{A}^2$, and $\bar{\mathcal{A}} = I$ in (3.11a), we get:

$$\mathcal{W}_i = \mathcal{A}(2\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2}) - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big) \tag{3.12}$$

where we used the fact $\mathcal{w}_i = \bar{\mathcal{A}}\mathcal{z}_i = \mathcal{z}_i$ from (3.11b). Consider the DIGing algorithm from [61, 62]:

$$\begin{cases} \mathcal{w}_i = \mathcal{A}\mathcal{w}_{i-1} - \mu \mathcal{x}_{i-1} & \text{(3.13a)} \\ \mathcal{x}_i = \mathcal{A}\mathcal{x}_{i-1} + \nabla \mathcal{J}(\mathcal{w}_i) - \nabla \mathcal{J}(\mathcal{w}_{i-1}) & \text{(3.13b)} \end{cases}$$

for $i \geq 0$ with $\mathcal{x}_{-1} = \nabla \mathcal{J}(\mathcal{w}_{-1})$ and arbitrary $\mathcal{w}_{-1}$. By eliminating the gradient tracking variable $\mathcal{x}_i$, it can be shown that recursion (3.12) is equivalent to the DIGing algorithm (3.13) – see [62, Section 2.2.1]. Since agent k is responsible for one block of $\mathcal{w}_i$ in (3.13), each agent k can update its block vector $w_{k,i}$ as follows:

$$w_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} w_{s,i-1} - \mu x_{k,i-1} \tag{3.14a}$$

$$x_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} x_{s,i-1} + \nabla J_k(w_{k,i}) - \nabla J_k(w_{k,i-1}) \tag{3.14b}$$

for $i \geq 0$ with $x_{k,-1} = \nabla J_k(w_{k,-1})$ and arbitrary $w_{k,-1}$.

- **(EXTRA [55])**: If $\mathcal{B}^2 = 0.5(I - \mathcal{A})$, $\mathcal{C} = 0.5(I - \mathcal{A})$, and $\bar{\mathcal{A}} = I$ in (3.11a), we get:

$$\mathcal{w}_i = 0.5(I + \mathcal{A})(2\mathcal{w}_{i-1} - \mathcal{w}_{i-2}) - \mu\big(\nabla \mathcal{J}(\mathcal{w}_{i-1}) - \nabla \mathcal{J}(\mathcal{w}_{i-2})\big) \tag{3.15}$$

where we used the fact $\mathcal{w}_i = \bar{\mathcal{A}}\mathcal{z}_i = \mathcal{z}_i$ from (3.11b). Recursion (3.15) is the EXTRA algorithm[2] from [55]. A per agent implementation of (3.15) can be obtained as follows. Let $\bar{A} = [\bar{a}_{sk}] = (I_K + A)/2$. Initialize $w_{k,0} = \sum_{s \in \mathcal{N}_k} \bar{a}_{sk} w_{s,-1} - \mu \nabla J_k(w_{k,-1})$ with $w_{k,-1}$ set to any arbitrary value . For every agent k, repeat for $i = 0, 1, 2, ...$

$$w_{k,i} = \sum_{s \in \mathcal{N}_k} \bar{a}_{sk}(2w_{s,i-1} - w_{s,i-2}) - \mu\big(\nabla J_k(w_{k,i-1}) - \nabla J_k(w_{k,i-2})\big) \tag{3.16}$$

[2]Recursion (3.15) is the recommended version of EXTRA with $\tilde{W} = 0.5(I + W)$ – see [55]

- **(Exact diffusion [39])**: If we choose $\bar{\mathcal{A}} = 0.5(I + \mathcal{A})$, $\mathcal{C} = 0$ and $\mathcal{B}^2 = 0.5(I - \mathcal{A})$ in (3.11a), we get:

$$\mathcal{Z}_i = \bar{\mathcal{A}}\mathcal{Z}_{i-1} + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)$$

  Multiplying the previous equation by $\bar{\mathcal{A}}$ and noting from (3.11b) that $\mathcal{W}_i = \bar{\mathcal{A}}\mathcal{Z}_i$, we get:

$$\mathcal{W}_i = 0.5(I + \mathcal{A})\Big(2\mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{3.17}$$

  The above recursion is the exact diffusion recursion first proposed in [39]. We also note that if we choose $\mathcal{C} = 0$, $\mathcal{B}^2 = c(I - \mathcal{A})$ ($c \in \mathbb{R}$), and $\bar{\mathcal{A}} = I - \mathcal{B}^2$ then we recover the smooth case of the NIDS algorithm from [56]. As highlighted in [56], NIDS is identical to exact diffusion for the smooth case when $c = 0.5$. At the agent level, recursion (3.17) can be represented as follows [39]. Let $\bar{A} = [\bar{a}_{sk}] = (I_K + A)/2$. Initialize $x_{k,-1} = \psi_{k,-1}$ and $w_{k,-1}$ arbitrary. For every agent k, repeat for $i = 0, 1, 2, ...$

$$\psi_{k,i} = w_{k,i-1} - \mu\nabla J_k(w_{k,i-1}) \tag{3.18a}$$

$$z_{k,i} = w_{k,i-1} + \psi_{k,i} - \psi_{k,i-1} \tag{3.18b}$$

$$w_{k,i} = \sum_{s\in\mathcal{N}_k} \bar{a}_{sk} z_{s,i} \tag{3.18c}$$

- **(Aug-DGM [59])**: Let $\mathcal{C} = 0$, $\bar{\mathcal{A}} = \mathcal{A}^2$, and $\mathcal{B} = I - \mathcal{A}$. Substituting into (3.11a):

$$\mathcal{Z}_i = (2\mathcal{A} - \mathcal{A}^2)\mathcal{Z}_{i-1} + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)$$

  By multiplying the previous equation by $\bar{\mathcal{A}} = \mathcal{A}^2$ and noting from (3.11b) that $\mathcal{W}_i = \mathcal{A}^2\mathcal{Z}_i$, we get the recursion:

$$\mathcal{W}_i = \mathcal{A}\Big(2\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\mathcal{A}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{3.19}$$

The above recursion is equivalent to the Aug-DGM [59] (also known as ATC-DIGing [60]) algorithm:

$$\begin{cases} \mathcal{W}_i = \mathcal{A}(\mathcal{W}_{i-1} - \mu \mathcal{X}_{i-1}) & \text{(3.20a)} \\ \mathcal{X}_i = \mathcal{A}\big(\mathcal{X}_{i-1} + \nabla \mathcal{J}(\mathcal{W}_i) - \nabla \mathcal{J}(\mathcal{W}_{i-1})\big) & \text{(3.20b)} \end{cases}$$

By eliminating the gradient tracking variable $\mathcal{X}_i$, we can rewrite the previous recursion as (3.19) – see Appendix 3.A. Since agent k is responsible for one block of $\mathcal{W}_i$ in (3.20), each agent k can update its block vector $w_{k,i}$ as follows:

$$w_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} \big(w_{s,i-1} - \mu x_{s,i-1}\big) \tag{3.21a}$$

$$x_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} \big(x_{s,i-1} + \nabla J_s(w_{s,i}) - \nabla J_s(w_{s,i-1})\big) \tag{3.21b}$$

for $i \geq 0$ with $x_{k,-1} = \nabla J_k(w_{k,-1})$ and arbitrary $w_{k,-1}$.

- **(ATC tracking method [57, 58])**: Let $\mathcal{C} = I - \mathcal{A}$ and $\mathcal{B} = I - \mathcal{A}$. Substituting into (3.11a):

$$\mathcal{Z}_i = (2\mathcal{A} - \mathcal{A}^2)\mathcal{Z}_{i-1} + \mathcal{A}\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2}))$$

By multiplying the previous equation by $\bar{\mathcal{A}} = \mathcal{A}$ and noting from (3.11b) that $\mathcal{W}_i = \mathcal{A}\mathcal{Z}_i$, we get the recursion:

$$\mathcal{W}_i = \mathcal{A}\Big(2\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{3.22}$$

The above recursion is equivalent to the following variant of the ATC tracking method [57, 58]:

$$\begin{cases} \mathcal{W}_i = \mathcal{A}(\mathcal{W}_{i-1} - \mu \mathcal{X}_{i-1}) & \text{(3.23a)} \\ \mathcal{X}_i = \mathcal{A}\mathcal{X}_{i-1} + \nabla \mathcal{J}(\mathcal{W}_i) - \nabla \mathcal{J}(\mathcal{W}_{i-1}) & \text{(3.23b)} \end{cases}$$

By eliminating the gradient tracking variable $\mathcal{X}_i$, we can show that the previous recursion is exactly (3.22) – see Appendix 3.A. Since agent k is responsible for one block of $\mathcal{W}_i$ in (3.23), each agent k can update its block vector $w_{k,i}$ as follows:

$$w_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk}\big(w_{s,i-1} - \mu x_{s,i-1}\big) \tag{3.24a}$$

$$x_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} x_{s,i-1} + \nabla J_k(w_{k,i}) - \nabla J_k(w_{k,i-1}) \tag{3.24b}$$

for $i \geq 0$ with $x_{k,-1} = \nabla J_k(w_{k,-1})$ and arbitrary $w_{k,-1}$. It can also be shown that (3.23) is equivalent to the following recursion as well (see Appendix 3.A):

$$\begin{cases} \mathcal{W}_i = \mathcal{A}\mathcal{W}_{i-1} - \mu \mathcal{X}_{i-1} & (3.25\text{a}) \\ \mathcal{X}_i = \mathcal{A}\big(\mathcal{X}_{i-1} + \nabla \mathcal{J}(\mathcal{W}_i) - \nabla \mathcal{J}(\mathcal{W}_{i-1})\big) & (3.25\text{b}) \end{cases}$$

Remark 3.1 (Communication cost)**.** Notice that EXTRA (3.15) and exact diffusion (3.17) communicate one vector per iteration. On the other hand, the gradient tracking method (3.23) and DIGing (3.13) requires communicating two vectors at each iteration i. Similarly, the Aug-DGM (ATC-DIGing) method (3.20) also requires sharing two variables; moreover, it requires communicating the two variables , $\mathcal{W}_{i-1} - \mu \mathcal{X}_{i-1}$ and $\mathcal{X}_{i-1} + \nabla \mathcal{J}(\mathcal{W}_i) - \nabla \mathcal{J}(\mathcal{W}_{i-1})$, sequentially (at different communication rounds). □

3.3 Convergence Results

In this section, we will give the auxiliary results leading to the main convergence result. To this end, we will first derive the error recursion.

It can be easily verified that the optimality conditions of problem (3.7) (see e.g. (2.7)) must satisfy the following conditions:

$$\begin{cases} 0 = \mu \nabla \mathcal{J}(\mathcal{W}^\star) + \mathcal{B}\mathcal{Y}^\star & (3.26\text{a}) \\ 0 = \mathcal{B}\mathcal{W}^\star & (3.26\text{b}) \end{cases}$$

Since problems (3.1) and (3.7) are equivalent, it holds that $\mathcal{W}^\star = \mathbb{1}_K \otimes w^\star$ where $w^\star$ is the unique solution for problem (3.1). Recall from Lemma 2.2 in Chapter 2 that there exists a particular saddle-point $(\mathcal{W}^\star, \mathcal{Y}_b^\star)$ where $\mathcal{Y}_b^\star$ is a unique vector that belongs to the range space of $\mathcal{B}$. In the following we will show that the iterates $(\mathcal{W}_i, \mathcal{Y}_i)$ in (3.9) converge linearly to the point $(\mathcal{W}^\star, \mathcal{Y}_b^\star)$. To this end, we introduce the error quantities:

$$\begin{aligned}
\widetilde{\mathcal{W}}_i &\triangleq \mathcal{W}_i - \mathcal{W}^\star \\
\widetilde{\mathcal{Y}}_i &\triangleq \mathcal{Y}_i - \mathcal{Y}_b^\star \\
\widetilde{\mathcal{Z}}_i &\triangleq \mathcal{Z}_i - \mathcal{W}^\star
\end{aligned}$$

Note that from condition (3.6) it holds that $\mathcal{C}\mathcal{W}^\star = 0$ since $\mathcal{W}^\star = \mathbb{1}_K \otimes w^\star$. Thus, using (3.26a)–(3.26b) in (3.9a)–(3.9c) we can reach the following error recursions:

$$\begin{cases}
\widetilde{\mathcal{Z}}_i = (I - \mathcal{C})\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}^\star)\big) - \mathcal{B}\widetilde{\mathcal{Y}}_{i-1} & \text{(3.27a)} \\
\widetilde{\mathcal{Y}}_i = \widetilde{\mathcal{Y}}_{i-1} + \mathcal{B}\widetilde{\mathcal{Z}}_i & \text{(3.27b)} \\
\widetilde{\mathcal{W}}_i = \bar{\mathcal{A}}\mathcal{Z}_i - \mathcal{W}^\star = \bar{\mathcal{A}}\widetilde{\mathcal{Z}}_i & \text{(3.27c)}
\end{cases}$$

where equation (3.27c) holds since $\bar{\mathcal{A}}\mathcal{W}^\star = \mathcal{W}^\star$. This follows from the fact that $\mathcal{W}^\star = \mathbb{1}_K \otimes w^\star$ and $\bar{\mathcal{A}} = \bar{A} \otimes I_M$ with $\bar{A}$ being doubly-stochastic. To analyze (3.27), we need the following technical conditions.

Assumption 3.2. (Combination matrices) *It is assumed that either one of the following conditions hold:*

$$\begin{cases}
0 < \bar{\mathcal{A}} \leq I - \mathcal{B}^2 & \text{(3.28a)} \\
0 \leq \mathcal{C} < 2I & \text{(3.28b)}
\end{cases}$$

or

$$\begin{cases}
\bar{\mathcal{A}} = I & \text{(3.29a)} \\
0 \leq \mathcal{B}^2 \leq \mathcal{C} < I & \text{(3.29b)}
\end{cases}$$

□

Conditions (3.28a)–(3.28b) will be utilized for the analysis of the ATC algorithms and conditions (3.29a)–(3.29b) will be used for the analysis of NON-ATC algorithms ($\bar{\mathcal{A}} = I$).

Remark 3.2 (Convergence conditions)**.** Recall that for exact diffusion [93] (see equation (3.17)) we have $\bar{\mathcal{A}} = 0.5(I + \mathcal{A}) = I - \mathcal{B}^2$ and $\mathcal{C} = 0$. Note that any primitive symmetric and doubly stochastic matrix A satisfies $-1 < A \leq 1$ – see Section [8, Lemma F.4]. Thus, exact diffusion satisfies conditions (3.28a)–(3.28b) for any primitive symmetric and doubly stochastic matrix A. Similar to exact diffusion, NIDS [56] also satisfies similar conditions. Recall also for EXTRA [55] (see equation (3.15)) we have $\bar{\mathcal{A}} = \mathcal{A}$ and $\mathcal{B}^2 = \mathcal{C} = 0.5(I - \mathcal{A})$, which satisfy conditions (3.29a)–(3.29b) for any primitive symmetric and doubly stochastic matrix A. For the ATC tracking methods [57–60], the conditions translate to the requirement that the eigenvalues of A must be in $(0, 1]$, rather than the typical $(-1, 1]$ — see e.g. [60]. Although this condition is not necessary, it can be easily satisfied by redefining $A \leftarrow 0.5(I+A)$. Note that most works that analyze decentralized methods under more relaxed conditions on the network topology impose restrictive step-size conditions that depend on the network and on the order of $O(\nu^{\theta_1}/\delta^{\theta_2})$ where $0 < \theta_1 \leq 1$ and $\theta_2 > 1$ – see [60, 61, 63, 122]. On the other hand, we require step sizes on the order of $O(1/\delta)$. Moreover, we will show that any algorithm that fits into our setup with $\mathcal{C} = 0$ can use a step-size as large as the centralized gradient descent – see discussion after Theorem 3.1.

$\square$

For the statement of our next results, we let $\underline{\sigma}(\mathcal{B}^2) = \underline{\sigma}^2(\mathcal{B})$ denote the minimum non-zero singular value of the matrix $\mathcal{B}^2$. Since $\mathcal{B}^2$ is symmetric, its singular values are equal to its eigenvalues and condition (3.28a) or (3.29b) implies $0 < \underline{\sigma}^2(\mathcal{B}) < 1$. We also let $\sigma_{\max}(\mathcal{C})$ denote the maximum singular value of $\mathcal{C}$.

In our analysis, we will utilize the following useful result from [55, Proposition 3.6].

Lemma 3.1. (Augmented Cost) *Under Assumption 3.1, the penalized augmented cost*

$\mathcal{J}(\mathcal{W}) + \frac{\rho}{2}\|\mathcal{W}\|^2_{\mathcal{B}^2}$ *with any* $\rho > 0$ *is restricted strongly-convex with respect to* $\mathcal{W}^\star$*:*

$$(\mathcal{W} - \mathcal{W}^\star)^\mathsf{T}\big(\nabla \mathcal{J}(\mathcal{W}) - \nabla \mathcal{J}(\mathcal{W}^\star)\big) + \rho\|\mathcal{W} - \mathcal{W}^\star\|^2_{\mathcal{B}^2} \geq \bar{\nu}_\rho \|\mathcal{W} - \mathcal{W}^\star\|^2 \tag{3.30}$$

where

$$\bar{\nu}_\rho = \min\left\{\bar{\nu} - 2\delta c, \frac{\rho \underline{\sigma}^2(\mathcal{B})c^2}{4(c^2+1)}\right\} > 0, \quad \textit{for any } c \in \left(0, \frac{\bar{\nu}}{2\delta}\right) \tag{3.31}$$

for any $\mathcal{W}$ *with* $\mathcal{W}^\star = \mathbb{1} \otimes w^\star$ *and where* $w^\star$ *denotes the minimizer of* (3.1)*. Moreover,* $\bar{\nu}_\rho \to \bar{\nu}$ *as* $\rho \to \infty$*.* □

The following lemma gives a useful inequality that will be used in our next results. The proof is provided in Appendix 3.B.

Lemma 3.2. (Useful inequality) *If Assumption 3.1 and step-size condition* $\mu < \frac{2-\sigma_{\max}(\mathcal{C})}{\delta}$ *hold, then the following inequality holds for any* $\rho > 0$ *and for all* $i \geq 0$*:*

$$\begin{aligned}\|\widetilde{\mathcal{Z}}_i\|^2_{\mathcal{Q}} + \|\widetilde{\mathcal{Y}}_i\|^2 \leq \big(1 - \mu\bar{\nu}_\rho(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)\big)\|\widetilde{\mathcal{W}}_{i-1}\|^2 + \big(1 - \underline{\sigma}^2(\mathcal{B})\big)\|\widetilde{\mathcal{Y}}_{i-1}\|^2 \\ + \mu\rho(2 - \mu\delta_\mu)\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{B}^2} - (2 - \mu\delta_\mu)\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{C}}\end{aligned} \tag{3.32}$$

where $\mathcal{Q} = I - \mathcal{B}^2$ *and* $\bar{\nu}_\rho$ *denote the strongly-convex parameter given in* (3.31)*.*

□

The previous lemma applies for the ATC primal-dual case (ATC-PD) where $\bar{\mathcal{A}}$ is a primitive combination matrix, which covers exact-diffusion and ATC-gradient-tracking methods; and the NON-ATC primal-dual case (NON-ATC-PD) where $\bar{\mathcal{A}} = I$, which covers EXTRA and DIGing. For the convergence result we will distinguish between these two cases. We start by stating the result for the ATC-PD case.

Theorem 3.1. (ATC-PD) *Let Assumption 3.1 and condition* (3.28) *given in Assumption*

3.2 hold. If the step-size satisfies

$$\mu < \frac{2 - \sigma_{\max}(\mathcal{C})}{\delta} \tag{3.33}$$

then it holds that $\|\widetilde{w}_i\|^2_{I+\mathcal{B}^2} + \|\widetilde{y}_i\|^2 \leq \gamma^i C_o$ *where* $C_o \geq 0$ *and*

$$\gamma = \max\left\{1 - \mu\bar{\nu}_\rho(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta),\ 1 - \underline{\sigma}^2(\mathcal{B}),\ \mu\rho(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)\right\} < 1 \tag{3.34}$$

for any $0 < \rho < 1/\mu(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)$ *with* $\bar{\nu}_\rho$ *denoting the strongly-convex parameter given in* (3.31).

$\square$

The proof of Theorem 3.1 is given in Appendix 3.C. We see that the convergence rate γ given in (3.34) clearly shows how the matrices $\mathcal{B}$ and $\mathcal{C}$ affect the convergence rate. For example, recall that for exact diffusion, $\underline{\sigma}^2(\mathcal{B})$ denote the smallest non-zero singular value (or eigenvalue) of the matrix $0.5(I - A)$. Thus, the effect of the network on the convergence rate is evident through the term $1 - \underline{\sigma}^2(\mathcal{B})$, which becomes close to one as the network becomes sparser. Another useful conclusion is for the case when $\mathcal{C} = 0$, the step-size bound becomes (3.33) becomes $\mu < \frac{2}{\delta}$, which is as large as the centralized gradient descent. This means that exact diffusion and Aug-DGM (ATC-DIGing) have wider stability range than the other variation discussed in Section 3.2 with $\mathcal{C} \neq 0$. Moreover, for $\mathcal{C} = 0$ the convergence rate becomes $\gamma = \max\{1 - \mu\nu(2 - \mu\delta), 1 - \underline{\sigma}^2(\mathcal{B})\} < 1$, which separates the network effect from the cost function. We now give the result for the NON-ATC case, which covers EXTRA as special case.

Theorem 3.2. (NON-ATC-PD) *Let Assumption 3.1 and condition* (3.29) *given in Assumption 3.2 hold. If the step-size satisfies*

$$\mu \leq \frac{1 - \sigma_{\max}(\mathcal{C})}{\delta} \tag{3.35}$$

then it holds that $\|\widetilde{\mathcal{W}}_i\|^2_{\mathcal{Q}} + \|\widetilde{\mathcal{Y}}_i\|^2 \leq \gamma^i C_o$ *where* $\mathcal{Q} > 0$, $C_o \geq 0$, *and*

$$\gamma \overset{\Delta}{=} \max\left\{1 - \mu(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)\bar{\nu}_\rho, 1 - \underline{\sigma}(\mathcal{B}^2)\right\} < 1 \tag{3.36}$$

with $\bar{\nu}_\rho$ *denoting the strongly-convex parameter given in* (3.31) *for any* ρ *satisfying* (3.55).

□

The proof of Theorem 3.2 is given in Appendix 3.D. Note that for the EXTRA case ($\mathcal{C} = \mathcal{B}^2$) our step-size condition (3.35) does not depend on the strongly-convex parameter. On the other hand, the bound given in the original work [55, Theorem 3.7] is $\mu < \frac{2\bar{\nu}_\rho(1-\sigma_{\max}(\mathcal{B}^2))}{\delta^2}$, which scales badly for ill-conditioned problems. We see that the step-size condition (3.35) range is smaller than the NON-ATC case. This means that the ATC case has larger stability range than the NON-ATC case. We can see from this that exact-diffusion (3.17) has larger stability range than EXTRA (3.15), which is inline with the results from [93]. Similarly, ATC-DIGing has larger stability range than DIGing, which has not been shown prior to this work.

3.4 Influence of Augmented Lagrangian Term

So far we have utilized the existence of the matrices $\mathcal{C}$ or $\bar{\mathcal{A}}$ in (3.9) to establish our result under strongly-convex aggregate costs. In this section, we ask our self what is the affect of these terms on the performance of decentralized algorithms. To answer this question it is sufficient to study the primal-dual gradient algorithms (2.3) applied on any specific instance of problem (3.7). Thus, a direct application of (2.3) to the Lagrangian of problem (3.7) with $\mathcal{C} = \mathcal{B}^2$ gives:

$$\mathcal{W}_i = \mathcal{W}_{i-1} - \mu_w \nabla \mathcal{J}_\rho(\mathcal{W}_{i-1}) - \mu_w \mathcal{B}\lambda_{i-1} \tag{3.37a}$$

$$\lambda_i = \lambda_{i-1} + \mu_\lambda \mathcal{B}\mathcal{W}_i \tag{3.37b}$$

where $\mathcal{J}_\rho(\mathcal{w}) = \mathcal{J}(\mathcal{w}) + \frac{\rho}{2}\|\mathcal{w}\|^2_{\mathcal{B}^2}$ and we are allowing $\rho = 0$. We will next show how the presence of the term $\frac{\rho}{2}\|\mathcal{w}\|^2_{\mathcal{B}^2}$ affects the performance of recursion (3.37).

As before, one can choose different choices for $\mathcal{B}$ and recover different algorithms. However, for illustration and later simulation, we will choose $\mathcal{B} = (I - \mathcal{A})^{\frac{1}{2}}$. Recursion (3.37) is not decentralized yet because $\mathcal{B} = (I - \mathcal{A})^{\frac{1}{2}}$ need not have the network structure. However, this can be easily handled by removing the dual variable or by a change of variable. We choose the later and let $\mathcal{y}_i = \mathcal{B}\lambda_i$ and multiply (3.37b) by $\mathcal{B}$ so that (3.37) becomes:

$$\begin{cases} \mathcal{w}_i = \mathcal{w}_{i-1} - \mu_w \nabla \mathcal{J}_\rho(\mathcal{w}_{i-1}) - \mu_w \mathcal{y}_{i-1} & (3.38a) \\ \mathcal{y}_i = \mathcal{y}_{i-1} + \mu_\lambda \mathcal{B}^2 \mathcal{w}_i & (3.38b) \end{cases}$$

Since $\mathcal{B}^2 = (I - \mathcal{A})$ has the network structure, then the k-th block of $\mathcal{B}^2\mathcal{w}_i = \text{col}\{u_{k,i}\}_{k=1}^K$ (of size M) has the decentralized form $u_{k,i} = w_{k,i} - \sum_{s\in\mathcal{N}_k} a_{sk} w_{s,i}$ where $\mathcal{N}_k$ denotes the neighbors of agent k, including agent k. Therefore, recursion (3.38) is decentralized and each can update its corresponding k-th blocks in $\mathcal{w}_i$ and $\mathcal{y}_i$. The convergence of (3.38) easily follows from the convergence of recursion (3.37), which follows from Theorem (2.1).

Corollary 3.1. *Let $(\mathcal{w}^\star, \lambda_b^\star)$ be the optimal point for the Lagrangian of (3.7) where $\lambda_b^\star$ lies in the range space of $\mathcal{B}$. Assume that the cost $\mathcal{J}_\rho(\mathcal{w})$ is ν-strongly convex and $\mathcal{J}(\mathcal{w})$ has δ-Lipschitz gradients. Under the step-size conditions in Theorem 2.1 with $B = \mathcal{B}$ and Lipschitz constant $\delta_\rho = \delta + \rho\sigma^2_{\max}(\mathcal{B})$, recursion (3.38a)–(3.38b) converges linearly to $(\mathcal{w}^\star, \mathcal{B}\lambda_b^\star)$ if $\mathcal{y}_{-1} = 0$.*

Proof. Since $\mathcal{J}(\mathcal{w})$ has δ-Lipschitz gradients, it holds that $\mathcal{J}_\rho(\mathcal{w}) = \mathcal{J}(\mathcal{w}) + \frac{\rho}{2}\|\mathcal{B}^2\mathcal{w}\|^2$ is $\delta_\rho = \delta + \rho\sigma^2_{\max}(\mathcal{B})$-smooth. We know that under the conditions in Theorem 2.1, recursion (3.37a)–(3.37b) converges to the point $(\mathcal{w}^\star, \lambda_b^\star)$ linearly if $\lambda_{-1} = 0$. Now, note that if $\lambda_{-1} = 0$ and $\mathcal{y}_{-1} = 0$, then from (3.37b) and (3.38b) it holds that $\mathcal{y}_i = \mathcal{B}\lambda_i$ for all $i \geq -1$. Since λ_i lies in the range space of $\mathcal{B}$, it follows from Lemma 2.1 that $\mathcal{y}_i = 0 \iff \lambda_i = 0$. Therefore, if recursions (3.37a)–(3.37b) converge linearly to $(\mathcal{w}^\star, \lambda_b^\star)$, then recursions (3.38a)–(3.38b) converge linearly to $(\mathcal{w}^\star, \mathcal{B}\lambda_b^\star)$. $\square$

It can be seen that the addition of the augmented term with $\rho > 0$ leads to a larger Lipschitz constant $\delta_\rho = \delta + \rho\sigma^2_{\max}(\mathcal{B}) > \delta$ and consequently to a smaller stability range for μ_w compared to the case when $\rho = 0$. However, the augmented term can be beneficial as we now explain. From Corollary (3.1) and Theorem 2.1 we have that the convergence rate of recursion (3.38) is upper bounded by

$$\gamma = \max\left\{1 - \mu_w\nu(1 - \mu_w\delta_\rho), 1 - \mu_w\mu_\lambda\underline{\sigma}^2(\mathcal{B})\right\} \tag{3.39}$$

If the aggregate cost $\bar{J}(w) = \frac{1}{K}\sum_{k=1}^{K} J_k(w) : \mathbb{R}^M \to \mathbb{R}$ is $\bar{\nu}$-strongly-convex, then from Lemma 3.1 we know that the augmented penalized cost $\mathcal{J}_\rho(\mathcal{W})$ is $\bar{\nu}_\rho$- strongly convex for $\rho > 0$. Therefore, for $\rho > 0$ recursion (3.38) is guaranteed to converges linearly as long as the aggregate cost $\bar{J}(w)$ is strongly-convex, which does not necessarily imply that each individual cost is strongly-convex. In this case $\nu = \bar{\nu}_\rho$ and γ will depend on $\bar{\nu}_\rho$. On the other hand, if $\rho = 0$ then $\mathcal{J}_0(\mathcal{W}) = \mathcal{J}(\mathcal{W}) = \sum_{k=1}^{K} J_k(w_k) : \mathbb{R}^{MK} \to \mathbb{R}$ is ν-strongly-convex if, and only, if each local cost $J_k(w_k)$ is ν_k-strongly convex, so that $\nu = \min \nu_k$. This implies that for $\rho = 0$, the cost $\mathcal{J}_0(\mathcal{W}) = \mathcal{J}(\mathcal{W})$ is not strongly-convex (can even be non-convex) unless each individual cost is strongly convex, which unlike the case $\rho > 0$ can lead to convergence issues. Moreover, the convergence rate γ will depend on $\nu = \min \nu_k$. Since from (3.31) we know that $\bar{\nu}_\rho \approx \bar{\nu}$ for large enough ρ, the convergence rate when $\rho > 0$ depends on $\bar{\nu}$. On the other hand, for $\rho = 0$ the convergence rate depends on $\nu = \min \nu_k$. Therefore, the AL method can converge much faster if $\bar{\nu}$ is much larger than $\nu = \min \nu_k$. However, when $\bar{\nu} \approx \nu$, then the AL penalty term is not that beneficial. These findings will be illustrated by means of simulations in the next section.

3.5 Numerical Simulations

We consider the distributed optimization problem (3.1) with quadratic costs $J_k(w) = w^\mathsf{T} R_k w + r_k^\mathsf{T} w$ where $w \in \mathbb{R}^{20}$, $R_k \in \mathbb{R}^{20\times 20}$, and $r_k \in \mathbb{R}^{20}$. A network of $K = 20$ agents shown in top rightmost side of Fig. 3.1 was randomly generated across a 1.2×1.2 grid and two agents are neighbors if they are less than $d = 0.25$ distance away from each other. This network is used for all implementations. The matrix A is generated using the Metropolis rule [8]. In all results, PD non-distributed refers to (3.37) ($\rho = 0$), PD distributed refers to (3.38) ($\rho = 0$), and AL PD distributed refers to (3.38) with $\rho > 0$. The step-sizes are manually chosen to get the best possible convergence rate for each algorithm.

The top leftmost plot of Fig. 3.1 shows the evolution of the relative error $\|\mathcal{W}_i - \mathcal{W}^\star\|^2 / \|\mathcal{W}^\star\|^2$ over iterations. The matrix $R_k > 0$ is a random diagonal matrix with integer diagonal entries, each chosen between $[2, 8]$, which is well conditioned because $8/2$ is not very large. Similarly, r_k is randomly generated vector with each entry uniformly selected between $[0, 2]$. The dual step-size is set to $\mu_\lambda = 30$ for all algorithms in the top leftmost plot of Fig. 3.1. All algorithms converge linearly with close performance. It is seen that PD non-distributed and PD distributed have identical performance. We also see that unlike PD distributed, AL PD distributed has a smaller primal stability range for a fixed dual step-size – see explanation below Corollary 3.1. To test the dual stability range, we fixed $\mu_w = 0.07$ and simulated PD distributed and AL PD distributed for different dual step-size. The results are shown in the middle leftmost plot of Fig. 3.1. We see that in this settings, the AL PD distributed also has a smaller dual stability range for a fixed primal step-size. Under the same set-up, we compare the algorithm with the EXTRA algorithm from [55] and exact diffusion from [39], both are AL based methods. The results are shown in the middle rightmost plot of Fig. 3.1. We see that all algorithms perform similarly. In this scenario, we do not see any advantages of AL methods compared to the Lagrangian method ($\rho = 0$).

To show the advantages of the AL method, we consider the same setting as before but under two scenarios: non-convex individual costs or ill-conditioned strongly-convex costs. For the non-convex case, we let all entries to be zero except for the following diagonal entries: the (k, k)-th diagonal entry for each agent $(R_k(k, k))$ are chosen randomly between $[2, 8]$, the entries $R_k(k-1, k-1) = -R_{k-1}(k-1, k-1)/2$ for all $k \geq 2$. In this case, the aggregate cost $\sum_{k=1}^{K}(w^\mathsf{T} R_k w + r_k^\mathsf{T} w)$ is strongly convex since $R = \sum_{k=1}^{K} R_k > 0$, but the individual costs $\{J_k(w)\}_{k\geq 2}$ are non-convex and therefore $\mathcal{J}(\mathcal{W}) = \sum_{k=1}^{K}(w_k^\mathsf{T} R_k w_k + r_k^\mathsf{T} w_k)$ is non-convex as well. This is because the Hessian $\nabla^2 \mathcal{J}(\mathcal{w}) = \text{blkdiag}\{R_k\}_{k=1}^K$ is indefinite. The results of this set-up is shown in the bottom leftmost plot of Fig. 3.1. We see that the AL PD distributed method still converges linearly. However, the PD distributed method diverges even under much smaller step-sizes than the previous set-ups. The bottom rightmost plot of Fig. 3.1 shows the ill conditioned case. We let all entries to be zero except for the diagonal entries: the (k, k)-th diagonal entry for each agent $(R_k(k, k))$ are chosen randomly between $[2, 8]$ and the other diagonal entries are chosen uniformly between $(0, 1)$, which can be very small. In this case, we see that the Lagrangian methods performs poorly compared to the other PD AL method ($\rho = 50$) and the other AL methods, which is in agreement with our conclusion from the previous section.

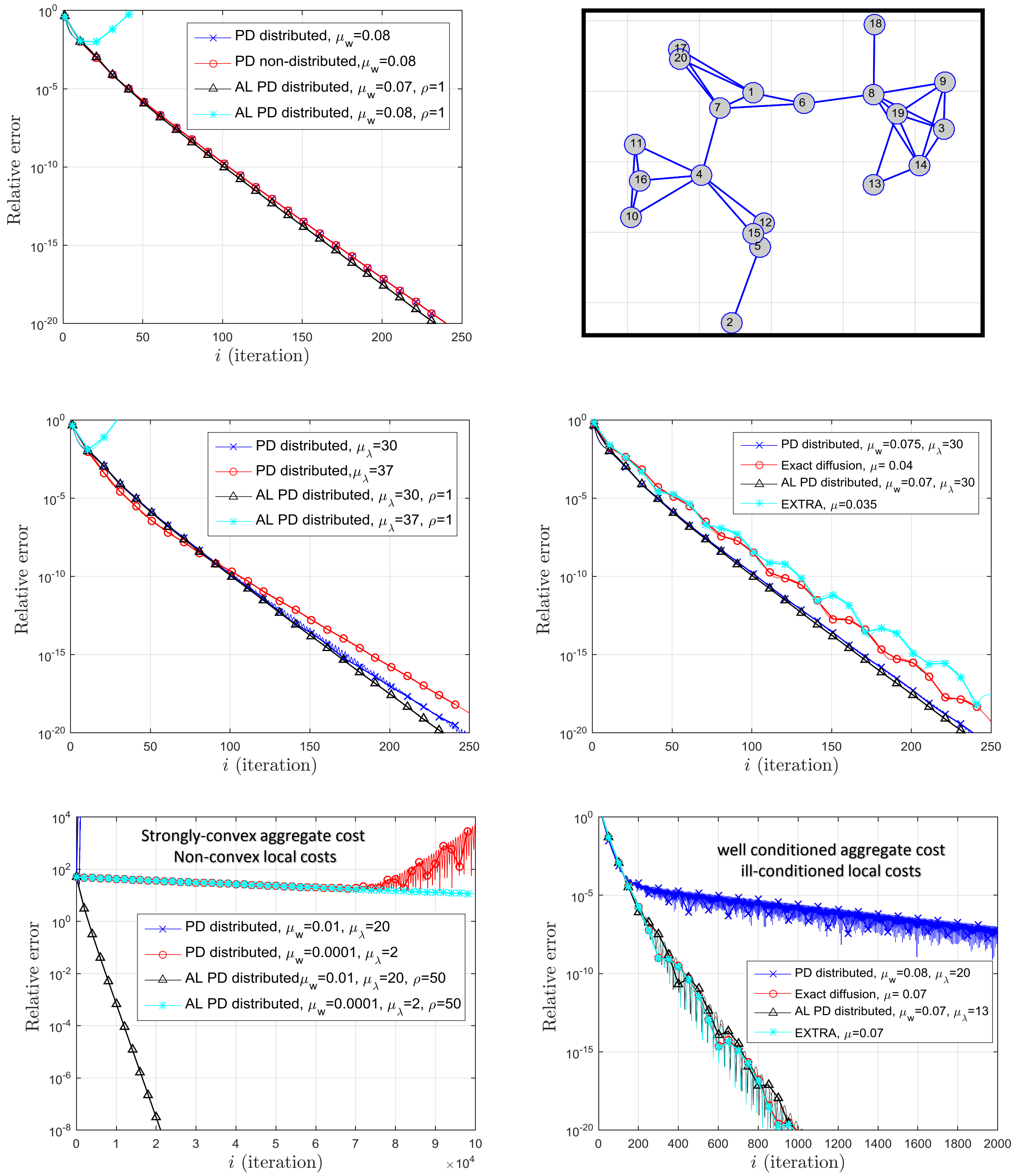


Figure 3.1: Simulation results illustrating the influence of the AL term.

Appendices

3.A Equivalent Representation

3.A.1 Aug-DGM (ATC-DIGing)

Here we show that (3.20) is equivalent to (3.19). From (3.20a) we have

$$\begin{aligned}
\mathcal{W}_i - \mathcal{A}\mathcal{W}_{i-1} &= \mathcal{A}\Big(\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\mathcal{X}_{i-1} - \mathcal{A}\mathcal{X}_{i-2}\big)\Big) \\
&\overset{(3.20b)}{=} \mathcal{A}\Big(\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\mathcal{A}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big)
\end{aligned}$$

Rearranging the previous equation we get:

$$\mathcal{W}_i = \mathcal{A}\Big(2\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\mathcal{A}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big)$$

which is recursion (3.19).

3.A.2 ATC-Tracking I

In a similar manner we can show that (3.23) is equivalent to (3.22). From (3.23a) we have

$$\begin{aligned}
\mathcal{W}_i - \mathcal{A}\mathcal{W}_{i-1} &= \mathcal{A}\Big(\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\mathcal{X}_{i-1} - \mathcal{A}\mathcal{X}_{i-2}\big)\Big) \\
&\overset{(3.23b)}{=} \mathcal{A}\Big(\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big)
\end{aligned}$$

Rearranging the previous equation we get:

$$\mathcal{W}_i = \mathcal{A}\Big(2\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big)$$

which is recursion (3.22).

3.A.3 ATC-Tracking II

Repeating similar arguments to the previous section, we can show that (3.25) is equivalent to (3.22). From (3.25a) we have

$$
\begin{aligned}
\mathcal{W}_i - \mathcal{A}\mathcal{W}_{i-1} &= \mathcal{A}\big(\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2}\big) - \mu\big(\mathcal{X}_{i-1} - \mathcal{A}\mathcal{X}_{i-2}\big) \\
&\overset{(3.25b)}{=} \mathcal{A}\big(\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2}\big) - \mu\mathcal{A}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)
\end{aligned}
$$

Rearranging the previous equation we get:

$$
\mathcal{W}_i = \mathcal{A}\Big(2\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}_{i-2})\big)\Big)
$$

which is recursion (3.22).

3.B Proof of Lemma 3.2

Proof. Squaring both sides of (3.27a) and (3.27b) we get

$$
\begin{aligned}
\|\widetilde{\mathcal{Z}}_i\|^2 &= \|(I-\mathcal{C})\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big)\|^2 + \|\mathcal{B}\widetilde{\mathcal{Y}}_{i-1}\|^2 \\
&\quad - 2\widetilde{\mathcal{Y}}_{i-1}^{\mathsf{T}}\mathcal{B}\left((I-\mathcal{C})\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big)\right)
\end{aligned}
\tag{3.40}
$$

and

$$
\begin{aligned}
\|\widetilde{\mathcal{Y}}_i\|^2 = \|\widetilde{\mathcal{Y}}_{i-1} + \mathcal{B}\widetilde{\mathcal{Z}}_i\|^2 &= \|\widetilde{\mathcal{Y}}_{i-1}\|^2 + \|\mathcal{B}\widetilde{\mathcal{Z}}_i\|^2 + 2\widetilde{\mathcal{Y}}_{i-1}^{\mathsf{T}}\mathcal{B}\widetilde{\mathcal{Z}}_i \\
&\overset{(3.27a)}{=} \|\widetilde{\mathcal{Y}}_{i-1}\|^2 + \|\widetilde{\mathcal{Z}}_i\|^2_{\mathcal{B}^2} - 2\|\mathcal{B}\widetilde{\mathcal{Y}}_{i-1}\|^2 \\
&\quad + 2\widetilde{\mathcal{Y}}_{i-1}^{\mathsf{T}}\mathcal{B}\left((I-\mathcal{C})\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big)\right)
\end{aligned}
\tag{3.41}
$$

Adding equation (3.41) to (3.40) and rearranging, we get

$$
\|\widetilde{\mathcal{Z}}_i\|^2_{\mathcal{Q}} + \|\widetilde{\mathcal{Y}}_i\|^2 = \|(I-\mathcal{C})\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big)\|^2 + \|\widetilde{\mathcal{Y}}_{i-1}\|^2 - \|\mathcal{B}\widetilde{\mathcal{Y}}_{i-1}\|^2 \tag{3.42}
$$

where $\mathcal{Q} = I - \mathcal{B}^2$ is positive definite from (3.28a). Since $\mathcal{Y}_0 = 0$ and $\mathcal{Y}_i = \mathcal{Y}_{i-1} + \mathcal{B}\mathcal{Z}_i$, we know $\mathcal{Y}_i \in \text{Range}(\mathcal{B})$ for any i. Thus, both $\mathcal{Y}_i$ and $\mathcal{Y}_b^\star$ lie in the range space of $\mathcal{B}$, and it holds that $\|\mathcal{B}\widetilde{\mathcal{Y}}_{i-1}\|^2 \geq \underline{\sigma}(\mathcal{B}^2)\|\widetilde{\mathcal{Y}}_{i-1}\|^2$. Therefore, we can bound (4.41) by

$$\|\widetilde{\mathcal{Z}}_i\|^2_{\mathcal{Q}} + \|\widetilde{\mathcal{Y}}_i\|^2 \leq \|\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{\mathcal{W}}_{i-1}\big)\|^2 + (1 - \underline{\sigma}(\mathcal{B}^2))\|\widetilde{\mathcal{Y}}_{i-1}\|^2 \tag{3.43}$$

Since $\mathcal{J}(\mathcal{W}) + \frac{1}{2\mu}\|\mathcal{W}\|^2_{\mathcal{C}}$ is $\delta_\mu = \delta + \frac{1}{\mu}\sigma_{\max}(\mathcal{C})$-smooth, it holds from (1.8) that:

$$\big\|\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{\mathcal{W}}_{i-1}\big\|^2 \leq \delta_\mu \widetilde{\mathcal{W}}_{i-1}^\mathsf{T}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{\mathcal{W}}_{i-1}\big) \tag{3.44}$$

Moreover, recall from (3.30) that

$$-\widetilde{\mathcal{W}}_{i-1}^\mathsf{T}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big) \leq -\bar{\nu}_\rho\|\widetilde{\mathcal{W}}_{i-1}\|^2 + \rho\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{B}^2} \tag{3.45}$$

for any $\rho > 0$. Using the above two bounds, it holds that:

$$\begin{aligned}
&\|\widetilde{\mathcal{W}}_{i-1} - \mu\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{\mathcal{W}}_{i-1}\big)\|^2 \\
&\overset{(3.44)}{\leq} \|\widetilde{\mathcal{W}}_{i-1}\|^2 - \mu(2-\mu\delta_\mu)\widetilde{\mathcal{W}}_{i-1}^\mathsf{T}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{\mathcal{W}}_{i-1}\big) \\
&= \|\widetilde{\mathcal{W}}_{i-1}\|^2 - \mu(2-\mu\delta_\mu)\widetilde{\mathcal{W}}_{i-1}^\mathsf{T}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big) - (2-\mu\delta_\mu)\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{C}} \\
&\overset{(3.45)}{\leq} \big(1 - \mu\bar{\nu}_\rho(2-\mu\delta_\mu)\big) + \mu\rho(2-\mu\delta_\mu)\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{B}^2} - (2-\mu\delta_\mu)\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{C}}
\end{aligned} \tag{3.46}$$

where in the last inequality step we used the fact that $2 - \mu\delta_\mu > 0$, which follows from the condition $\mu < (2 - \sigma_{\max}(\mathcal{C}))/\delta$. Substituting (3.46) into (3.43) gives (3.32). $\square$

3.C Proof of Theorem 3.1

Proof of Theorem 3.1. Recall from Lemma 3.2 that

$$\begin{aligned}\|\widetilde{\mathcal{z}}_i\|^2_{\mathcal{Q}} + \|\widetilde{\mathcal{y}}_i\|^2 \le \big(1-\mu\bar{\nu}_\rho(2-\sigma_{\max}(\mathcal{C})-\mu\delta)\big)\|\widetilde{\mathcal{w}}_{i-1}\|^2 + \big(1-\underline{\sigma}^2(\mathcal{B})\big)\|\widetilde{\mathcal{y}}_{i-1}\|^2 \\ + \mu\rho(2-\mu\delta_\mu)\|\widetilde{\mathcal{w}}_{i-1}\|^2_{\mathcal{B}^2} - (2-\mu\delta_\mu)\|\widetilde{\mathcal{w}}_{i-1}\|^2_{\mathcal{C}}\end{aligned} \tag{3.47}$$

Note that:

$$\big(1-\mu\nu(2-\sigma_{\max}(\mathcal{C})-\mu\delta)\big) < 1 \iff \mu < \frac{2-\sigma_{\max}(\mathcal{C})}{\delta} \tag{3.48}$$

Let $\gamma_1 = 1-\mu\bar{\nu}_\rho\big(2-\sigma_{\max}(\mathcal{C})-\mu\delta\big)$ and $\gamma_2 = 1-\underline{\sigma}^2(\mathcal{B})$. Then inequality (3.47) becomes

$$\begin{aligned}\|\widetilde{\mathcal{z}}_i\|^2_{\mathcal{Q}} + \|\widetilde{\mathcal{y}}_i\|^2 &\le \gamma_1\|\widetilde{\mathcal{w}}_{i-1}\|^2 + \gamma_2\|\widetilde{\mathcal{y}}_{i-1}\|^2 + \mu\rho(2-\mu\delta_\mu)\|\widetilde{\mathcal{w}}_{i-1}\|^2_{\mathcal{B}^2} - (2-\mu\delta_\mu)\|\widetilde{\mathcal{w}}_{i-1}\|^2_{\mathcal{C}} \\ &\le \gamma_1\|\widetilde{\mathcal{w}}_{i-1}\|^2 + \gamma_2\|\widetilde{\mathcal{y}}_{i-1}\|^2 + \mu\rho(2-\mu\delta_\mu)\|\widetilde{\mathcal{w}}_{i-1}\|^2_{\mathcal{B}^2}\end{aligned} \tag{3.49}$$

We will now show that $\|\widetilde{\mathcal{w}}_i\|^2_{I+\mathcal{B}^2} \le \|\widetilde{\mathcal{z}}_i\|^2_{\mathcal{Q}}$. Since $\bar{\mathcal{A}}$ is doubly stochastic and from condition (3.28) we know that $0 < \bar{\mathcal{A}} \le I$. Therefore it holds $I-\bar{\mathcal{A}} \ge 0$ and thus

$$0 \le \bar{\mathcal{A}}^{\frac{1}{2}}(I-\bar{\mathcal{A}})^2\bar{\mathcal{A}}^{\frac{1}{2}} \iff \bar{\mathcal{A}}^2 - \bar{\mathcal{A}}^3 \le \bar{\mathcal{A}} - \bar{\mathcal{A}}^2 \tag{3.50}$$

Utilizing the above bound it holds that

$$\bar{\mathcal{A}}\mathcal{B}^2\bar{\mathcal{A}} \overset{(3.28a)}{\le} \bar{\mathcal{A}}(I-\bar{\mathcal{A}})\bar{\mathcal{A}} = \bar{\mathcal{A}}^2 - \bar{\mathcal{A}}^3 \overset{(3.50)}{\le} \bar{\mathcal{A}} - \bar{\mathcal{A}}^2 \tag{3.51}$$

From the above equation we conclude that

$$\begin{aligned}\|\widetilde{\mathcal{w}}_i\|^2 \overset{(3.27c)}{=} \|\widetilde{\mathcal{z}}_i\|^2_{\bar{\mathcal{A}}^2} = \|\widetilde{\mathcal{z}}_i\|^2_{\bar{\mathcal{A}}} - \|\widetilde{\mathcal{z}}_i\|^2_{\bar{\mathcal{A}}-\bar{\mathcal{A}}^2} &\overset{(3.51)}{\le} \|\widetilde{\mathcal{z}}_i\|^2_{\bar{\mathcal{A}}} - \|\widetilde{\mathcal{z}}_i\|^2_{\bar{\mathcal{A}}\mathcal{B}^2\bar{\mathcal{A}}} \\ &\le \|\widetilde{\mathcal{z}}_i\|^2_{\mathcal{Q}} - \|\widetilde{\mathcal{z}}_i\|^2_{\bar{\mathcal{A}}\mathcal{B}^2\bar{\mathcal{A}}}\end{aligned} \tag{3.52}$$

where we used condition (3.28a) in the last step, i.e., $\bar{\mathcal{A}} \leq \mathcal{Q} = I - \mathcal{B}^2$. Using

$$\|\widetilde{z}_i\|^2_{\bar{\mathcal{A}}\mathcal{B}^2\bar{\mathcal{A}}} = \|\bar{\mathcal{A}}\widetilde{z}_i\|^2_{\mathcal{B}^2} = \|\widetilde{w}_i\|^2_{\mathcal{B}^2}$$

in (3.52) gives $\|\widetilde{w}_i\|^2_{I+\mathcal{B}^2} \leq \|\widetilde{z}_i\|^2_{\mathcal{Q}}$. Using this bound in (3.49) yields:

$$\begin{aligned}\|\widetilde{w}_i\|^2_{I+\mathcal{B}^2} + \|\widetilde{y}_i\|^2 &\leq \gamma_1\|\widetilde{w}_{i-1}\|^2 + \gamma_2\|\widetilde{y}_{i-1}\|^2 + \gamma_3\|\widetilde{w}_{i-1}\|^2_{\mathcal{B}^2} \\ &\leq \max\{\gamma_1, \gamma_2, \gamma_3\}\left(\|\widetilde{w}_{i-1}\|^2_{I+\mathcal{B}^2} + \|\widetilde{y}_{i-1}\|^2\right) \end{aligned} \tag{3.53}$$

where we introduced $\gamma_3 = \mu\rho(2 - \mu\delta_\mu) = \mu\rho(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)$. Iterating we reach our result. □

3.D Proof of Theorem 3.2

Proof of Theorem 3.2. From condition (3.29b) we know that $\bar{\mathcal{A}} = I$; thus, from (3.27c) we have $\widetilde{w}_i = \widetilde{z}_i$. Using this the inequality in Lemma 3.2 gives

$$\begin{aligned}\|\widetilde{w}_i\|^2_{\mathcal{Q}} + \|\widetilde{y}_i\|^2 &\leq \big(1 - \mu\bar{\nu}_\rho(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)\big)\|\widetilde{w}_{i-1}\|^2 + \big(1 - \underline{\sigma}^2(\mathcal{B})\big)\|\widetilde{y}_{i-1}\|^2 \\ &\quad + \mu\rho(2 - \mu\delta_\mu)\|\widetilde{w}_{i-1}\|^2_{\mathcal{B}^2} - (2 - \mu\delta_\mu)\|\widetilde{w}_{i-1}\|^2_{\mathcal{C}} \\ &\leq \big(1 - \mu\bar{\nu}_\rho(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta)\big)\|\widetilde{w}_{i-1}\|^2 + \big(1 - \underline{\sigma}^2(\mathcal{B})\big)\|\widetilde{y}_{i-1}\|^2 \\ &\quad - (2 - \mu\delta_\mu)(1 - \mu\rho)\|\widetilde{w}_{i-1}\|^2_{\mathcal{B}^2}\end{aligned}$$

where we used (3.29) in the last step. Let $\gamma_1 = (1 - \mu(2 - \mu\delta_\mu)\bar{\nu}_\rho)$ and $\gamma_2 = 1 - \underline{\sigma}^2(\mathcal{B})$. Adding and subtracting $\gamma_1\|\widetilde{w}_{i-1}\|^2_{\mathcal{B}^2}$ to the previous inequality gives

$$\|\widetilde{w}_i\|^2_{\mathcal{Q}} + \|\widetilde{y}_i\|^2 \leq \gamma_1\|\widetilde{w}_{i-1}\|^2_{\mathcal{Q}} + \gamma_2\|\widetilde{y}_{i-1}\|^2 - \big((2 - \mu\delta_\mu)(1 - \mu\rho) - \gamma_1\big)\|\widetilde{w}_{i-1}\|^2_{\mathcal{B}^2} \tag{3.54}$$

We now need to ensure that $-\big((2-\mu\delta_\mu)(1-\mu\rho)-\gamma_1\big)\|\widetilde{w}_{i-1}\|^2_{\mathcal{B}^2} \leq 0$. To do that it is sufficient to find μ and ρ such that

$$0 < \rho \leq \frac{2-\gamma_1-\sigma_{\max}(\mathcal{C})-\mu\delta}{\mu(2-\mu\delta_\mu)} = \frac{1-\sigma_{\max}(\mathcal{C})-\mu\delta+\mu(2-\mu\delta_\mu)\bar{\nu}_\rho}{\mu(2-\mu\delta_\mu)} \tag{3.55}$$

It remains to find μ such that ρ is non-negative. To do that, we note that $0 < \gamma_1 < 1$ if

$$\mu < \frac{2}{\delta_\mu} \iff \mu < \frac{2-\sigma_{\max}(\mathcal{C})}{\delta} \tag{3.56}$$

for $\gamma_1 < 1$, it easy to verify that ρ is strictly positive if μ satisfy:

$$\mu \leq \frac{1-\sigma_{\max}(\mathcal{C})}{\delta} < \frac{2-\sigma_{\max}(\mathcal{C})}{\delta} \tag{3.57}$$

Under this condition, we can upper bound (3.54) by

$$\|\widetilde{w}_i\|^2_{\mathcal{Q}} + \|\widetilde{y}_i\|^2 \leq \gamma_1\|\widetilde{w}_{i-1}\|^2_{\mathcal{Q}} + \gamma_2\|\widetilde{y}_{i-1}\|^2 \leq \max\{\gamma_1,\gamma_2\}(\|\widetilde{w}_{i-1}\|^2_{\mathcal{Q}} + \|\widetilde{y}_{i-1}\|^2) \tag{3.58}$$

Iterating we reach our result. $\square$

CHAPTER 4

Decentralized Proximal Primal-dual Algorithms

This chapter studies decentralized composite optimization problems with a non-smooth regularization term. For smooth optimization problems, decentralized methods have convergence rates that match centralized methods. However, some gap between decentralized and centralized *proximal* gradient methods continues to exist in the presence of a non-smooth term. While centralized proximal gradient methods have been shown to converge linearly for strongly convex objectives, it remains an open question to establish the linear convergence of *decentralized* proximal gradient based methods. This chapter closes this gap by proposing a proximal gradient decentralized algorithmic framework that is shown to converge linearly to the desired solution. Next we explain the problem set-up and comment on existing related works.

4.1 Problem Set-up

We consider a static and undirected network of K agents connected over some graph where each agent k owns a private cost function $J_k(w) : \mathbb{R}^M \to \mathbb{R}$. Through only local interactions (i.e., with agents only communicating with their immediate neighbors), each node is interested in finding a solution to the following problem:

$$w^\star = \underset{w \in \mathbb{R}^M}{\arg\min} \quad \frac{1}{K} \sum_{k=1}^{K} J_k(w) + R(w) \tag{4.1}$$

where $R(w) : \mathbb{R}^M \to \mathbb{R} \cup \{+\infty\}$ is a convex function (not necessarily differentiable). We adopt the following assumption throughout this chapter.

Assumption 4.1. (Cost function): *There exists a solution* $w^\star$ *to problem* (4.1) *and each cost function* $J_k(w)$ *is first-order differentiable and* ν*-strongly-convex:*

$$(w^o - w^\bullet)^\mathsf{T}\big(\nabla J_k(w^o) - \nabla J_k(w^\bullet)\big) \geq \nu\|w^o - w^\bullet\|^2 \tag{4.2}$$

with δ*-Lipschitz continuous gradients:*

$$\|\nabla J_k(w^o) - \nabla J_k(w^\bullet)\| \leq \delta\|w^o - w^\bullet\| \tag{4.3}$$

for any w^o *and* $w^\bullet$*. Constants* ν *and* δ *are strictly positive and satisfy* $0 < \nu \leq \delta$*. We also assume* $R(w)$ *to be a proper*[1] *and lower-semicontinuous convex function.* □

Note that from the strong-convexity condition (4.2), we know the objective function in (4.1) is also strongly convex and, thus, the global solution $w^\star$ is unique.

4.1.1 Related Work

The core problem in decentralized optimization is to design methods with convergence rates that are comparable to their centralized counterparts. For the smooth case ($R(w) = 0$), the decentralized primal methods from [37, 41, 42, 125] can only converge linearly to a *biased* solution and not the exact solution. For convergence to the exact solution, these primal methods require employing a decaying step-size that slows down the convergence rate making it sublinear at $O(1/i)$ in general. The works [50–52] established linear convergence to the exact solution for decentralized methods based on ADMM and inexact augmented Lagrangian techniques. After that other works established linear convergence for simpler implementations including linearized D-ADMM [53, 54], EXTRA [55] , ESOM [126], gradient tracking methods [61, 62], exact diffusion [39], NIDS [56], and others. The work [127] study the problem from the dual domain and propose accelerated dual gradient descent to reach an optimal convergence rate for smooth strongly-convex problems.

[1] The function $f(.)$ is proper if $-\infty < f(x)$ for all x in its domain and $f(x) < \infty$ for at least one x.

Many works exist on decentralized composite optimization problems with a non-smooth regularization term. The work [43] considered a similar set-up to this chapter and proposed a proximal gradient method combined with Nesterov's acceleration that is shown to achieve $O(1/i^2)$ convergence rate, but requires an increasing number of inner loop consensus steps with each iteration, which is expensive. Other works focused on the case where each agent k has a local regularizer $R_k(w)$ possibly different from other agents. For example, a proximal decentralized linearized ADMM (DL-ADMM) approach is proposed in [53] to solve such composite problems with convergence guarantees, while the work [128] establishes a sublinear convergence rate $O(1/i)$ for DL-ADMM when each $J_k(w)$ is smooth with Lipschitz continuous gradient. PG-EXTRA [129] extends EXTRA [55] to handle non-smooth regularization local terms and it establishes an improved rate $o(1/i)$. The NIDS algorithm [56] also has an $o(1/i)$ rate and can use larger step-sizes compared to PG-EXTRA. Based on existing results, there is still a clear gap between decentralized algorithms and centralized ones when when using proximal gradient methods.

The work [130] established the *asymptotic* linear convergence[2] of a proximal decentralized algorithm under the condition that all functions $\{J_k(w), R_k(w)\}$ (possibly different regularizers) are *piecewise linear-quadratic* (PLQ) functions. While this result is encouraging, it does not cover the global linear convergence rate we seek in this chapter since their linear rate occurs only after a sufficiently large number of iterations and requires all costs to be PLQ. Another useful work [131] extends the CoCoA algorithm [4] to COLA algorithm for decentralized settings and shows linear convergence in the presence of a non-differentiable regularizer. Like most other dual coordinate methods, COLA considers decentralized learning for *generalized linear models* (e.g., linear regression, logistic regression, SVM, etc). This is because COLA requires solving (4.1) from the dual domain and the linear model facilitates the derivation of the dual functions, which do not necessarily admit a closed form expression under our set-up. Additionally, different from this work, COLA is not a proximal gradient-based method; it requires solving an inner minimization problem to a satisfactory accuracy, which

[2]A sequence $\{x_i\}_{i=0}^{\infty}$ has asymptotic linear convergence to $x^\star$ if there exists a sufficiently large i_o such that $\|x_i - x^\star\| \leq \gamma^i C$ for some $C > 0$ and all $i \geq i_o$.

is often computationally expensive but necessary for the linear convergence analysis.

In order to establish global linear convergence, this chapter assumes the non-smooth term to be common across all agents. One might wonder whether it is possible for a decentralized proximal gradient algorithm to achieve global linear convergence in the presence of different local non-smooth $R_k(w)$ terms. As far as we know, this question has not been explicitly answered in the literature. Many decentralized optimization problems where each agent k has a local non-smooth term $R_k(w)$ possibly different from other agents have been proposed [53, 56, 128, 129, 132]. None of these methods have been shown to achieve global linear convergence in the presence of general non-smooth terms. By adjusting the results from [133] to the decentralized optimization set-up with agent-specific non-smooth terms $\{R_k(w)\}$, we show that it is *impossible* for any proximal gradient based algorithm to achieve linear convergence in the *worst case*. Note that the work [134] showed that global linear convergence is not possible for non-smooth strongly-convex functions in the worst case for the class of algorithms limited to one communication round but unlimited in the amount of computation and access to the functions per iteration. In contrast, we consider algorithms unlimited in the number of communications rounds but limited to one gradient and proximal computations per iteration.

4.1.2 Contribution

This chapter considers the composite optimization problem (4.1) and has two main contributions. First, for the case of a common non-smooth regularizer $R(w)$ across all computing agents, we propose a proximal decentralized algorithm framework whose fixed point coincides with the desired global solution $w^\star$. We then provide a short proof to establish its linear convergence for the general loss function $J_k(w)$ that is smooth and strongly convex. This result closes the existing gap between decentralized proximal gradient based methods and the centralized proximal gradient descent for strongly-convex objectives. Second, by tailoring a result from [133], we show that if each agent owns a non-smooth term, then linear convergence cannot be achieved in the worst case for the class of decentralized algorithms where each agent can compute one gradient and one proximal mapping per iteration for the smooth and non-smooth part, respectively. We further provide a numerical counter example where PG-EXTRA [129] and proximal linearized ADMM [53, 128] fail to achieve global linear convergence for strongly-convex objectives.

4.2 Proximal ATC Algorithms

In this section, we extend the ATC framework (3.9) to handle the non-differentiable component $R(w)$. To do that, let $w_k \in \mathbb{R}^M$ denote a local copy of w available at agent k and introduce the network quantities:

$$\mathcal{W} \triangleq \text{col}\{w_1, \cdots, w_K\} \in \mathbb{R}^{MK} \tag{4.4}$$

$$\mathcal{J}(\mathcal{W}) \triangleq \sum_{k=1}^{K} J_k(w_k) \tag{4.5}$$

$$\mathcal{R}(\mathcal{W}) \triangleq \sum_{k=1}^{K} R(w_k) \tag{4.6}$$

Recall from the Chapter 3, we introduced two general consensus symmetric matrices $\mathcal{B} \in \mathbb{R}^{MK\times MK}$ and $\mathcal{C} \in \mathbb{R}^{MK\times MK}$ that are assumed satisfy (3.6). Similar to reformulation (3.7), it is easy to see that problem (4.1) with is equivalent to the following problem:

$$\underset{\mathcal{W}\in\mathbb{R}^{KM}}{\text{minimize}} \quad \mathcal{J}(\mathcal{W}) + \mathcal{R}(\mathcal{W}) + \frac{1}{2\mu}\|\mathcal{W}\|^2_{\mathcal{C}}, \quad \text{s.t. } \mathcal{B}\mathcal{W} = 0 \tag{4.7}$$

where $\mu > 0$, and the matrix $\mathcal{C} \in \mathbb{R}^{MK\times MK}$ is a positive semi-definite consensus penalty matrix satisfying (3.6b). We propose the following recursion: let $\mathcal{Y}_{-1} = 0$ and $\mathcal{W}_{-1}$ take any arbitrary value. Repeat for $i = 0, 1, \ldots$

$$\begin{cases} \mathcal{Z}_i = (I - \mathcal{C})\mathcal{W}_{i-1} - \mu\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \mathcal{B}\mathcal{Y}_{i-1} & \text{(4.8a)} \\ \mathcal{Y}_i = \mathcal{Y}_{i-1} + \mathcal{B}\mathcal{Z}_i & \text{(4.8b)} \\ \mathcal{W}_i = \mathbf{prox}_{\mu\mathcal{R}}\big(\bar{\mathcal{A}}\mathcal{Z}_i\big) & \text{(4.8c)} \end{cases}$$

We refer the reader to Appendix 4.B for specific instances of the above algorithm and how to implement them in a decentralized manner.

Remark 4.1 (CONVENTIONAL UPDATE)**.** Recursion (4.8) differs from conventional proximal primal-dual algorithms in the dual update (4.8b). Different from conventional dual updates that use $\mathcal{W}_i$, we use $\mathcal{Z}_i$ instead of $\mathcal{W}_i$. This subtle (yet novel) difference changes the complexity of the algorithm and allows us to close the linear convergence gap between centralized and decentralized algorithms for problems of the form (4.1). $\square$

In the following, we will show that $\mathcal{W}_i$ in the above recursion converges to $\mathbb{1}_K \otimes w^\star$ where $w^\star$ is the desired solution of (4.1). We first prove the existence and optimality of the fixed points of recursion (4.8).

Lemma 4.1. (Optimality point) *Under Assumption 4.1 and condition* (3.6)*, a fixed point* $(\mathcal{W}^\star, \mathcal{Y}^\star, \mathcal{Z}^\star)$ *exists for recursions* (4.8a)–(4.8c)*, i.e., it holds that*

$$\begin{cases} \mathcal{Z}^\star = \mathcal{W}^\star - \mu\nabla\mathcal{J}(\mathcal{W}^\star) - \mathcal{B}\mathcal{Y}^\star & (4.9a) \\ 0 = \mathcal{B}\mathcal{Z}^\star & (4.9b) \\ \mathcal{W}^\star = \mathbf{prox}_{\mu\mathcal{R}}(\bar{\mathcal{A}}\mathcal{Z}^\star) & (4.9c) \end{cases}$$

Moreover, $\mathcal{W}^\star$ *and* $\mathcal{Z}^\star$ *are unique and each block element of* $\mathcal{W}^\star = \mathrm{col}\{w_1^\star, \cdots, w_K^\star\}$ *coincides with the unique solution* $w^\star$ *to problem* (4.1)*, i.e.,* $w_k^\star = w^\star$ *for all* k*.*

Proof. See Appendix 4.A. $\square$

4.3 Linear Convergence

Similar to Lemma 2.2, it can be easily shown that there exists a particular fixed point $(\mathcal{W}^\star, \mathcal{Y}_b^\star, \mathcal{Z}^\star)$ where $\mathcal{Y}_b^\star$ is a unique vector that belongs to the range space of $\mathcal{B}$. In the following we will show that the iterates $(\mathcal{W}_i, \mathcal{Y}_i, \mathcal{Z}_i)$ converge linearly to this particular fixed point

$(w^\star, y_b^\star, z^\star)$. To this end, we introduce the error quantities:

$$\widetilde{w}_i \overset{\Delta}{=} w_i - w^\star, \quad \widetilde{y}_i \overset{\Delta}{=} y_i - y_b^\star, \quad \widetilde{z}_i = z_i - z^\star \tag{4.10}$$

Note that from condition (3.6) we have $\mathcal{C}w^\star = 0$. Therefore, from (4.8a)–(4.8c) and (4.9a)–(4.9c) we can reach the following error recursions:

$$\begin{cases} \widetilde{z}_i = (I - \mathcal{C})\widetilde{w}_{i-1} - \mu\big(\nabla \mathcal{J}(w_{i-1}) - \nabla \mathcal{J}(w^\star)\big) - \mathcal{B}\widetilde{y}_{i-1} & (4.11\text{a}) \\ \widetilde{y}_i = \widetilde{y}_{i-1} + \mathcal{B}\widetilde{z}_i & (4.11\text{b}) \\ \widetilde{w}_i = \mathbf{prox}_{\mu\mathcal{R}}\big(\bar{\mathcal{A}}z_i\big) - \mathbf{prox}_{\mu\mathcal{R}}(\bar{\mathcal{A}}z^\star) & (4.11\text{c}) \end{cases}$$

For our convergence result, we need the following technical conditions.

Assumption 4.2. (Combination matrices) *It is assumed that both condition* (3.6) *and the following condition hold:*

$$\begin{cases} 0 < I - \mathcal{B}^2 & (4.12\text{a}) \\ \bar{\mathcal{A}}^2 \leq I - \mathcal{B}^2 \text{ and } 0 \leq \mathcal{C} < 2I & (4.12\text{b}) \end{cases}$$

□

Remark 4.2 (Convergence conditions)**.** In this chapter, we focus on the adapt-then-combine framework. See Remark 3.2 for more details regarding the above conditions.

□

To state our result, we let $\underline{\sigma}(\mathcal{B}^2)$ denote the minimum non-zero singular value of the matrix $\mathcal{B}^2$. Since $\mathcal{B}^2$ is symmetric, its singular values are equal to its eigenvalues and condition (4.12a) implies $0 < \underline{\sigma}(\mathcal{B}^2) < 1$. We also let $\sigma_{\max}(\mathcal{C}) < 2$ denote the maximum singular value of $\mathcal{C}$.

Theorem 4.1. (Linear convergence) *Under Assumptions 4.1–4.2, if* $y_0 = 0$ *and the step-size satisfies* $\mu < \frac{2-\sigma_{\max}(\mathcal{C})}{\delta}$, *it holds that*

$$\|\widetilde{w}_i\|^2 + \|\widetilde{y}_i\|^2 \leq \gamma\big(\|\widetilde{w}_{i-1}\|^2 + \|\widetilde{y}_{i-1}\|^2\big) \tag{4.13}$$

where $\gamma = \max\left\{1 - \mu\nu(2 - \sigma_{\max}(\mathcal{C}) - \mu\delta), 1 - \underline{\sigma}(\mathcal{B}^2)\right\} < 1$.

Proof. See Appendix 4.C $\square$

In the above theorem we established the linear convergence in the presence of a common non-smooth term. Now, a natural question is what about the case where each agent owns a local non-smooth term. Unfortunately, the answer to that question is that it is not possible to establish global linear convergence for that case in general as we now show.

4.4 Separate non-smooth terms: Sublinear rate

In this section, we will show that global linear convergence cannot be attained (in the worst case) if there exists more than one non-smooth term. Consider the more general problem with agent specific regularizers:

$$\min_{w \in \mathbb{R}^M} \frac{1}{K} \sum_{k=1}^{K} J_k(w) + R_k(w), \tag{4.14}$$

where $J_k(w)$ is a strongly convex smooth function and $R_k(w)$ is non-smooth convex with closed form proximal mappings (each $J_k(w)$ and $R_k(w)$ are further assumed to be closed and proper functions). Although many algorithms (centralized and decentralized) exist that solve (4.14), none have been shown to achieve linear convergence in the presence of general non-smooth proximal terms $R_k(w)$. In the following, by tailoring the results from [133], we show that this is not possible when having access to the proximal mapping of each individual non-smooth term $R_k(w)$ separately.

Let $\mathcal{H}$ be a deterministic algorithm that queries

$$\{J_k(\cdot), R_k(\cdot), \nabla J_k(\cdot), \mathbf{prox}_{\mu_{i,k} R_k}(\cdot) \,|\, \mu_{i,k} > 0,\, k = 1, \ldots, K\}$$

once for each iteration $i = 0, 1, \ldots$. To clarify, the scalar parameter $\mu_{i,k} > 0$ can differ for $i = 0, 1, \ldots$ and $k = 1, \ldots, K$ or they can be constants (e.g. $\mu_{i,k} = \mu > 0$). Note that $\mathcal{H}$ has the option to combine the queried values in any possible combination (it can only use certain information from certain communications). Thus, $\mathcal{H}$ includes decentralized algorithms in which communication is restricted to edges on a graph.

Consider the specific instance of (4.14)

$$\min_{w\in\mathbb{R}^M} \; F_\nu(w) = \frac{\nu}{2}\|w\|^2 + \frac{1}{K}\sum_{k=1}^{K} R_k(w) \tag{4.15}$$

where $\nu > 0$ and $J_k(w) = \frac{\nu}{2K}\|w\|^2$. Assume $R_k(w) < \infty$ if and only if $\|w\| \leq B$ and $|R_k(w_1) - R_k(w_2)| \leq G\|w_1 - w_2\|$ for all w_1, w_2 (where B and G are some positive constants) such that $\|w_1\| \leq B$ and $\|w_2\| \leq B$. To prove that linear convergence is not possible, we will reduce our setup to $\min_{w\in\mathbb{R}^M} F_0(w)$, which has a known lower bound [133]. Let $\mathcal{H}_o$ be a deterministic algorithm that queries

$$\{R_k(\cdot), \mathbf{prox}_{\mu_{i,k}R_k(\cdot)}(\cdot) \,|\, \mu_{i,k} > 0,\; k = 1,\ldots,K\}$$

once for each iteration $i = 0, 1, \ldots$ and communicates through a fully connected network. The following result is a special case of the more general result [133, Theorem 1].

Theorem 4.2. *Let $0 < B$, $0 < G$, $2 \leq K$, and $0 < \varepsilon < GB/12$. For a large enough problem dimension $M = \mathcal{O}(KGB/\varepsilon)$, the algorithm $\mathcal{H}_o$ (in the worst case) requires $\mathcal{O}(GB/\varepsilon)$ or more iterations to find a $\hat{w}$ such that $F_0(\hat{w}) - \inf_w F_0(w) < \varepsilon$.* □

We argue that algorithm $\mathcal{H}$ cannot be too efficient at solving $\min_w F_\nu(w)$ with $\nu > 0$ as otherwise it can be used to efficiently solve $\min_w F_0(w)$ and contradict Theorem 4.2.

Theorem 4.3. *Let $0 < \nu$, $0 < B$, $0 < G$, $2 \leq K$, and $0 < \varepsilon < G^2/(288\nu)$. For a large enough problem dimension $M = \mathcal{O}(KG/\sqrt{\nu\varepsilon})$, the algorithm $\mathcal{H}$ (in the worst case) requires $\mathcal{O}(G/\sqrt{\nu\varepsilon})$ or more iterations to find a $\hat{w}$ such that $F_\nu(\hat{w}) - \inf_w F_\nu(w) < \varepsilon$.*

Proof. This argument modifies the proof of [133, Theorem 2], which makes a similar but slightly different claim. Let $\nu = \varepsilon/B^2$ and $w_\nu^\star$ denotes the minimizer of F_ν. Assume for contradiction that $\mathcal{H}$ can find a $\hat{w}$ such that

$$F_\nu(\hat{w}) - F_\nu(w_\nu^\star) < \frac{\varepsilon}{2} \tag{4.16}$$

in $o(G/\sqrt{\nu\varepsilon})$ iterations. Note that for all w such that $\|w\| \le B$, it holds from (4.15):

$$F_\nu(w) \le F_0(w) + \frac{\nu B^2}{2} = F_0(w) + \frac{\varepsilon}{2}. \tag{4.17}$$

Putting these together, we get

$$F_0(\hat{w}) - F_0(w_0^\star) - \frac{\varepsilon}{2} \overset{(4.17)}{\le} F_0(\hat{w}) - F_\nu(w_0^\star) \overset{(a)}{\le} F_\nu(\hat{w}) - F_\nu(w_\nu^\star) < \frac{\varepsilon}{2}, \tag{4.18}$$

where in step (a) we used $F_0(w) \le F_\nu(w)$ and $F_\nu(w_\nu^\star) \le F_\nu(w_0^\star)$. We conclude that $F_0(\hat{w}) - F_0(w_0^\star) < \varepsilon$. Since $\nabla J_k(\cdot) = \frac{\nu}{K} I$ is just a scaled identity, querying $\nabla J_k(\cdot)$ does not provide a new direction that $\mathcal{H}_o$ could otherwise not use. Thus, algorithm $\mathcal{H}$ applied to minimizing F_ν is an instance of algorithm $\mathcal{H}_o$. This means that we have an algorithm for minimizing F_0 in $o(G/\sqrt{\nu\varepsilon}) = o(GB/\varepsilon)$ iterations, which contradicts Theorem 4.2. Note that $0 < \varepsilon < GB/12$ from Theorem 4.2 and by using $\nu = \varepsilon/B^2$ we require $0 < \varepsilon < G^2/(288\nu)$ (an extra factor of 2 appears because of (4.16)). □

Corollary 4.1. *For the problem setup of* (4.14) *with strongly convex* $J_k(\cdot)$ *for all* $k = 1, 2, \ldots, K$*, any algorithm that accesses the functions through evaluations of* $J_k(\cdot)$ *and* $R_k(\cdot)$*, the gradients of* $J_k(\cdot)$*, and proximal operators of* $R_k(\cdot)$ *is not globally linearly convergent (in the worst case).* □

Remark 4.3. The lower bound of Theorem 4.3 is *dimension independent* in the same way other Nesterov-type lower bounds are [95, 133]. The result implies that it is not possible to establish linear convergence of $\mathcal{H}$ with a rate depending on ν and G, but not on the problem dimension K. That said, a dimension dependent linear convergence may be established. For example, *asymptotic* linear convergence[3] has been established in [130] when the functions

[3] A sequence $\{x_i\}_{i=0}^\infty$ has asymptotic linear convergence to $x^\star$ if there exists a sufficiently large i_o such

$\{J_k(\cdot), R_k(\cdot)\}$ are piecewise linear quadratic. This result does not contradict our result as the linear rate and the number of iterations needed to observe the linear rate are *dependent* on the problem dimension. Our linear convergence result of Theorem 4.1 is dimension independent as it holds for any dimension M. $\square$

4.5 Simulations

4.5.1 Simulations of the Proposed Method

In this section we test the performance of three different instances of the proposed method (4.8) along with some state-of-the-art algorithms. We consider the following sparse logistic regression problem:

$$\min_{w \in \mathbb{R}^M} \frac{1}{K} \sum_{k=1}^{K} J_k(w) + \rho \|w\|_1 \quad \text{where} \quad J_k(w) = \frac{1}{L} \sum_{\ell=1}^{L} \ln(1 + \exp(-y_{k,\ell} x_{k,\ell}^{\mathsf{T}} w)) + \frac{\lambda}{2} \|w\|^2.$$

where $\{x_{k,\ell}, y_{k,\ell}\}_{\ell=1}^{L}$ are local data kept by agent k and L is the size of the local dataset. We consider three real datasets: Covtype.binary, MNIST, and CIFAR10. The last two datasets have been transformed into binary classification problems by considering data with two labels, digital two and four ('2' and '4') classes for MNIST, and cat and dog classes for CIFAR-10. In Covtype.binary we use 50,000 samples as training data and each data has dimension 54. In MNIST we use 10,000 samples as training data and each data has dimension 784. In CIFAR-10 we use 10,000 training data and each data has dimension 3072. All features have been preprocessed and normalized to the unit vector.

For the network, we generated a randomly connected network with $K = 20$ nodes, which is shown in Fig. 4.1. The associated combination matrix A is generated according to the Metropolis rule [8, 92]. For all simulations, we assign data evenly to each agent. We set $\lambda = 10^{-4}$ and $\rho = 2 \times 10^{-3}$ for Covtype, $\lambda = 10^{-2}$ and $\rho = 5 \times 10^{-4}$ for CIFAR-10, and $\lambda = 10^{-4}$ and $\rho = 2 \times 10^{-3}$ for MNIST. The simulation results are shown in Figure 4.2.

that $\|x_i - x^\star\| \leq \gamma^i C$ for some $C > 0$ and all $i \geq i_o$.

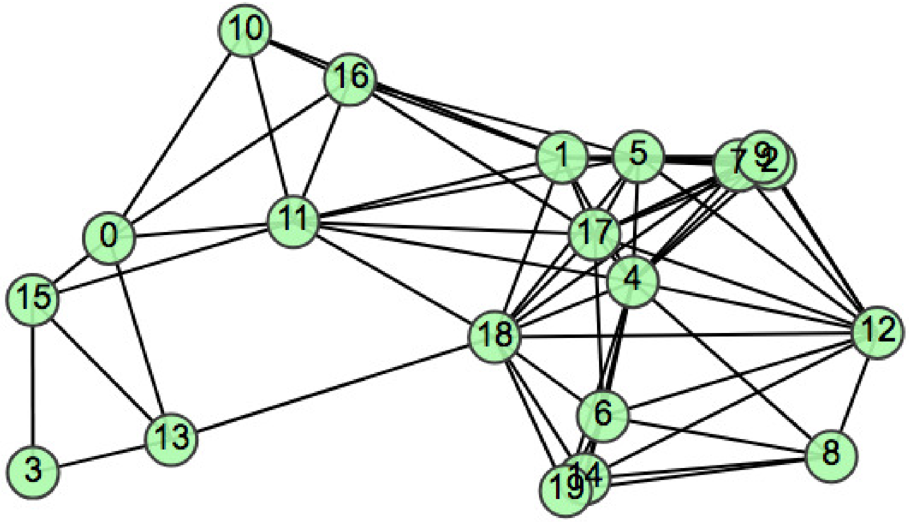


Figure 4.1: The network topology used in the simulation.

The decentralized implementations of Prox-ED, Prox-ATC I, and prox-ATC II are given in Appendix 4.B. For each algorithm, we tune the step-sizes manually to achieve the best possible convergence rate. We notice that the performance of each algorithm differs in each data set. Prox-ED always performs the best in our simulation setup. Prox-ATC I and prox-ATC II have comparable performance to Prox-ED in the MNIST data set but performs worst in th other two data-sets.

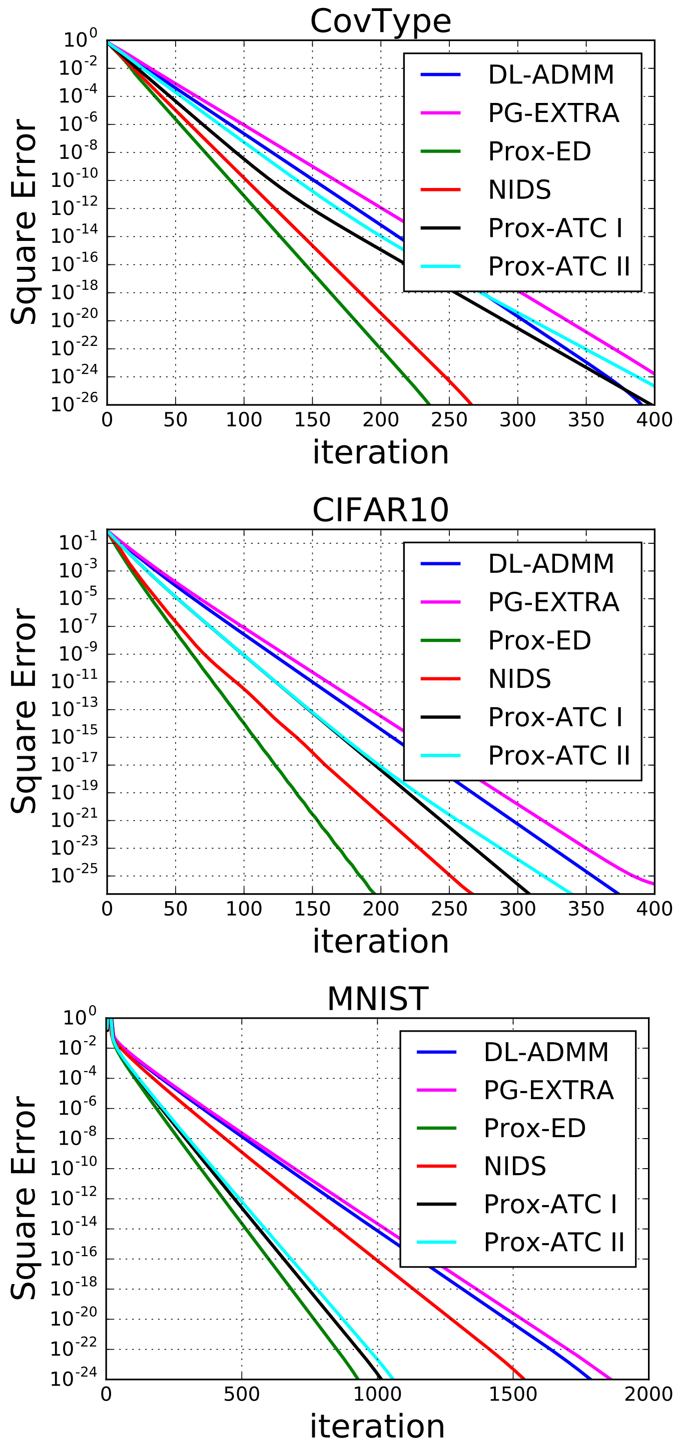


Figure 4.2: Simulation results. The y-axis indicates the relative squared error $\sum_{k=1}^{K} \|w_{k,i} - w^\star\|^2/\|w^\star\|^2$. Prox-ED refers to (4.8) with $\bar{\mathcal{A}} = 0.5(I+\mathcal{A})$, $\mathcal{B}^2 = 0.5(I-\mathcal{A})$, and $\mathcal{C} = 0$. Prox-ATC I refers to (4.8) with $\bar{\mathcal{A}} = \mathcal{A}^2$, $\mathcal{B} = I - \mathcal{A}$, and $\mathcal{C} = 0$. Prox-ATC II refers to (4.8) with $\bar{\mathcal{A}} = \mathcal{A}$, $\mathcal{B} = I - \mathcal{A}$, and $\mathcal{C} = I - \mathcal{A}$. DL-ADMM [53], PG-EXTRA [129], NIDS [56].

4.5.2 Numerical counter example

In this section, we provide a numerical counter example, which validates that linear convergence is not possible for problem (4.14) in general. We consider an instance of (4.14) with $K = 2$, M is a very large even number, and quadratic smooth terms $J_k(w) = \eta/2\|w\|^2$ for some $\eta > 0$. We let the non-smooth terms be

$$R_1(w) = |\sqrt{2}w(1) - 1| + |w(2) - w(3)| + |w(4) - w(5)| + \cdots + |w(M-2) - w(M-1)| \tag{4.19a}$$

$$R_2(w) = |w(1) - w(2)| + |w(3) - w(4)| + \cdots + |w(M-1) - w(M)| \tag{4.19b}$$

Both $\mathbf{prox}_{R_1}$ and $\mathbf{prox}_{R_2}$ have closed forms — see Appendix 4.D for details. The above example is related to the one in [134], which was used to derive lower bounds for a different class of algorithms as explained in the introduction.

In the numerical experiment, we test the performance of two well known decentralized proximal methods, PG-EXTRA [129] and DL-ADMM [53, 128]. We set $M = 2000$ and $\eta = 1$. The step-sizes for both PG-EXTRA and DL-ADMM are set to 0.005. The combination matrix is set as $A = \frac{1}{2}\mathbb{1}_2\mathbb{1}_2^{\mathsf{T}}$. The numerical results in Fig. 4.3 shows that both PG-EXTRA and DL-ADMM perform almost the same, and they converge sublinearly to the solution. No global linear convergence is observed in the simulation for sufficiently large dimension M and algorithms independent of M, which is consistent with our discussion in Remark 4.3.

Appendices

4.A Proof of Lemma 4.1

To establish existence we will construct a point $(\mathcal{W}^\star, \mathcal{Y}^\star, \mathcal{Z}^\star)$ that satisfies equations (4.9a)–(4.9c). Since each $J_k(w)$ is strongly convex, there exists a unique solution $w^\star$ for problem (4.1), i.e., $0 \in \frac{1}{K}\sum_{k=1}^{K} \nabla J_k(w^\star) + \partial R(w^\star)$. This also indicates that there must exist a subgradient

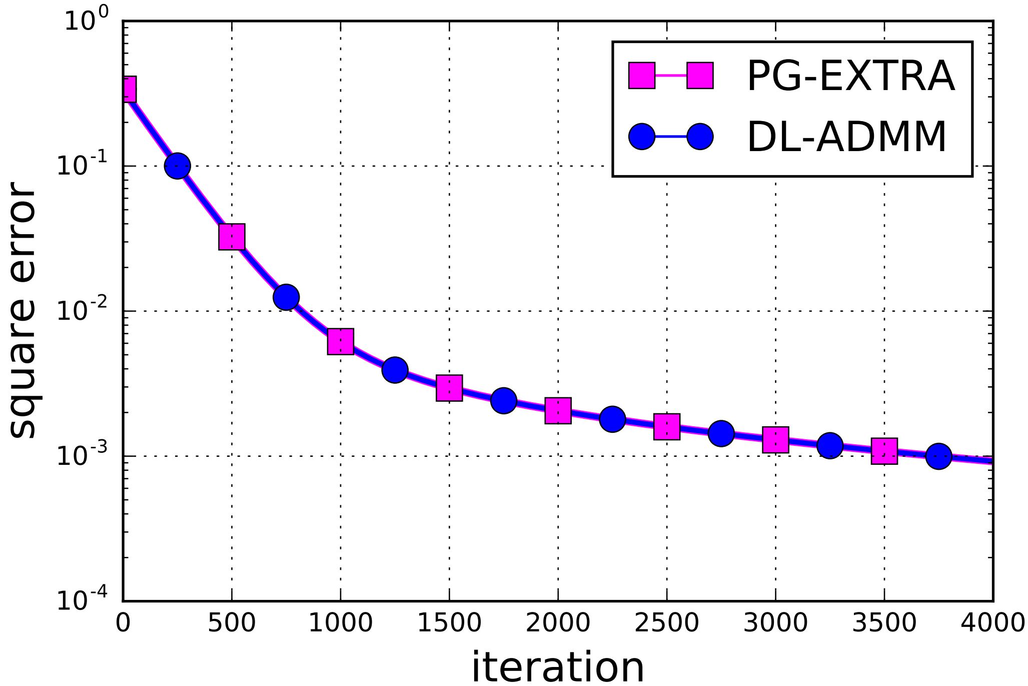


Figure 4.3: Both PG-EXTRA [129] and DL-ADMM [53, 128] converge sublinearly to the solution of the proposed numerical counter example.

$r^\star \in \partial R(w^\star)$ such that

$$\frac{1}{K}\sum_{k=1}^{K} \nabla J_k(w^\star) + r^\star = 0 \tag{4.20}$$

Now we define $z^\star \triangleq \mu r^\star + w^\star$. It holds that $r^\star + (w^\star - z^\star)/\mu = 0$, i.e., $0 \in \partial R(w^\star) + (1/\mu)(w^\star - z^\star)$. This implies that

$$w^\star = \arg\min_{w} \left\{ R(w) + \frac{1}{2\mu}\|w - z^\star\|^2 \right\}. \tag{4.21}$$

We next define $\mathcal{w}^\star \triangleq \mathbb{1}_K \otimes w^\star$ and $\mathcal{z}^\star \triangleq \mathbb{1}_K \otimes z^\star$. Since $\mathcal{z}^\star = \mathbb{1}_K \otimes z^\star$, it belongs to the null space of $\mathcal{B}$ so that $\mathcal{B}\mathcal{z}^\star = 0$ and, moreover, $\bar{\mathcal{A}}\mathcal{z}^\star = \mathcal{z}^\star$ since $\bar{\mathcal{A}} = \bar{A} \otimes I_M$ where $\bar{A}$ is doubly stochastic. Therefore, relation (4.21) implies that equation (4.9c) holds. It remains to construct $\mathcal{Y}^\star$ that satisfies equation (4.9a). Note that

$$(\mathbb{1}_N \otimes I_M)^\mathsf{T}\big(\mathcal{W}^\star - \mathcal{Z}^\star - \mu\nabla\mathcal{J}(\mathcal{W}^\star)\big) = -\mu K r^\star - \mu\sum_{k=1}^{K}\nabla J_k(w^\star) = 0, \tag{4.22}$$

where the last equality holds because $w^\star$ is the optimal solution of problem (4.1). Equation (4.22) implies

$$\frac{1}{\mu}\big(\mathcal{W}^\star - \mathcal{Z}^\star - \mu\nabla\mathcal{J}(\mathcal{W}^\star)\big) \in \mathrm{Null}(\mathbb{1}_N \otimes I_M) = \mathrm{Null}(\mathcal{B})^\perp = \mathrm{Range}(\mathcal{B}). \tag{4.23}$$

where $\perp$ denotes the orthogonal complement. Therefore, there exist a vector $\mathcal{Y}^\star$ satisfying equation (4.9a).

We now establish that any fixed point is of the form $\mathcal{W}^\star = \mathbb{1}_K \otimes w^\star$ and $w^\star$ is the solution to problem (4.1). From (4.9b) and (3.6), it holds that the block elements of $\mathcal{Z}^\star$ are equal to each other, i.e. $z_1^\star = \cdots = z_K^\star$, and we denote each block element by $z^\star$. Thus, $\bar{\mathcal{A}}\mathcal{Z}^\star = \mathcal{Z}^\star = \mathbb{1}_K \otimes z^\star$ because $\bar{\mathcal{A}} = \bar{A} \otimes I_M$ where $\bar{A}$ is doubly stochastic. Therefore, from (4.9c) and the definition of the proximal operator it holds that

$$w_k^\star = \arg\min_{w_k} \{R(w_k) + \|w_k - z^\star\|^2/2\mu\} \tag{4.24}$$

where we used $z_k^\star = z^\star$ for each k. Thus, we must have $w_1^\star = \cdots = w_K^\star \overset{\Delta}{=} w^\star$. It is easy to verify that (4.24) implies

$$0 \in \partial R(w^\star) + (w^\star - z^\star)/\mu. \tag{4.25}$$

Multiplying $(\mathbb{1}_K \otimes I_M)^\mathsf{T}$ from the left to both sides of equation (4.9a), we get

$$K z^\star = K w^\star - \mu\sum_{k=1}^{K}\nabla J_k(w^\star) \tag{4.26}$$

Combining (4.25) and (4.26), we get $0 \in \frac{1}{K}\sum_{k=1}^{K}\nabla J_k(w^\star) + \partial R(w^\star)$. Thus, $w^\star$ is the unique solution to problem (4.1). Due to the uniqueness of $w^\star$, we see from (4.26) that $z^\star$ is unique. Consequently, $\mathcal{W}^\star = \mathbb{1}_K \otimes w^\star$ and $\mathcal{Z}^\star = \mathbb{1}_K \otimes z^\star$ must be unique.

4.B Implementation of (4.8)

We introduce the combination matrices

$$A = [a_{sk}] \in \mathbb{R}^{K\times K}, \quad \mathcal{A} = A \otimes I_M \tag{4.27}$$

where the entry $a_{sk} = 0$ if there is no edge connecting agents k and s. The matrix A is assumed to be symmetric and doubly stochastic matrix (different from $\bar{A}$).

4.B.1 Prox-ED: $\bar{\mathcal{A}} = 0.5(I + \mathcal{A})$, $\mathcal{B}^2 = 0.5(I - \mathcal{A})$, and $\mathcal{C} = 0$

Recursion (4.8) with $\bar{\mathcal{A}} = 0.5(I + \mathcal{A})$, $\mathcal{B}^2 = 0.5(I - \mathcal{A})$, and $\mathcal{C} = 0$ is equivalent to the proximal exact diffusion (Prox-ED) recursion listed in (4.32a)–(4.32d). To see this, note that for $i = 0$, it is straight forward to check that each block in (4.8c) is the same as $w_{k,0}$ in (4.32). Now we will show the equivalence for $i \geq 1$. From (4.8a), we know that:

$$\begin{aligned}
\mathcal{Z}_i - \mathcal{Z}_{i-1} &= \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big) - \mathcal{B}(\mathcal{Y}_{i-1} - \mathcal{Y}_{i-2}) \\
&= \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big) - \mathcal{B}^2 \mathcal{Z}_{i-1}
\end{aligned} \tag{4.28}$$

where we used (4.8b) in the last step. Re-arranging and noting that $\mathcal{B}^2 = 0.5(I - \mathcal{A})$ we get

$$\mathcal{Z}_i = \bar{\mathcal{A}}\mathcal{Z}_{i-1} + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big) \tag{4.29}$$

By multiplying $\bar{\mathcal{A}}$ to both sides of the previous equation and introducing $\mathcal{X}_i \overset{\Delta}{=} \bar{\mathcal{A}}\mathcal{Z}_i$ we get

$$\mathcal{X}_i = \bar{\mathcal{A}}\Big(\mathcal{X}_{i-1} + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{4.30}$$

Thus from (4.8c) we get

$$\mathcal{X}_i = \bar{\mathcal{A}}\Big(\mathcal{X}_{i-1} + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{4.31a}$$

$$\mathcal{W}_i = \mathbf{prox}_{\mu R}(\mathcal{X}_i) \tag{4.31b}$$

The above recursion is equivalent to (4.32a)–(4.32d). This can be easily seen by substituting (4.32a)–(4.32b) into (4.32c).

Algorithm (Prox-ED)

Setting: Let $\bar{A} = [\bar{a}_{sk}] = (I_K + A)/2$. Initialize $x_{k,-1} = \psi_{k,-1}$ and $w_{k,-1}$ arbitrary. For every agent k, repeat for $i = 0, 1, 2, ...$

$$\psi_{k,i} = w_{k,i-1} - \mu \nabla J_k(w_{k,i-1}) \tag{4.32a}$$

$$z_{k,i} = x_{k,i-1} + \psi_{k,i} - \psi_{k,i-1} \tag{4.32b}$$

$$x_{k,i} = \sum_{s \in \mathcal{N}_k} \bar{a}_{sk} z_{s,i} \quad \textbf{(Communication step)} \tag{4.32c}$$

$$w_{k,i} = \mathbf{prox}_{\mu R}(x_{k,i}) \tag{4.32d}$$

4.B.2 Prox-ATC I: $\bar{\mathcal{A}} = \mathcal{A}^2$, $\mathcal{B}^2 = (I - \mathcal{A})^2$, and $\mathcal{C} = 0$

For the choice $\bar{\mathcal{A}} = \mathcal{A}^2$, $\mathcal{B}^2 = (I - \mathcal{A})^2$, and $\mathcal{C} = 0$, we can represent (4.8) as listed in (4.35). This can be seen by following the same approach as the previous subsection. To see this, note that with $\mathcal{B}^2 = (I - \mathcal{A})^2$ to get

$$\mathcal{Z}_i = (2\mathcal{A} - \mathcal{A}^2)\mathcal{Z}_{i-1} + \mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big) \tag{4.33}$$

By multiplying $\mathcal{A}^2$ to both sides of the previous equation and introducing $\mathcal{X}_i \triangleq \mathcal{A}^2 \mathcal{Z}_i$ we get

$$\mathcal{X}_i = \mathcal{A}\Big((2I - \mathcal{A})\mathcal{X}_{i-1} + \mathcal{A}\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\mathcal{A}\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{4.34}$$

Thus from (4.8c) we have $\mathcal{W}_i = \mathbf{prox}_{\mu R}(\mathcal{X}_i)$.

Algorithm: Prox-ATC I

Setting: Initialize $x_{k,-1} = \psi_{k,-1} = 0$ and $w_{k,-1}$ arbitrary. For every agent k, repeat for $i = 0, 1, 2, ...$

$$\psi_{k,i} = w_{k,i-1} - \mu \nabla J_k(w_{k,i-1}) \tag{4.35a}$$

$$z_{k,i} = 2x_{k,i-1} - \sum_{s \in \mathcal{N}_k} a_{sk}(x_{s,i-1} - \psi_{s,i} + \psi_{s,i-1}) \qquad \textbf{(Communication step)} \tag{4.35b}$$

$$x_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} z_{s,i} \qquad \textbf{(Communication step)} \tag{4.35c}$$

$$w_{k,i} = \mathbf{prox}_{\mu R}(x_{k,i}) \tag{4.35d}$$

4.B.3 Prox-ATC II: $\bar{\mathcal{A}} = \mathcal{A}$, $\mathcal{B} = I - \mathcal{A}$, and $\mathcal{C} = I - \mathcal{A}$

For the choice $\bar{\mathcal{A}} = \mathcal{A}$, $\mathcal{B} = I - \mathcal{A}$, and $\mathcal{C} = I - \mathcal{A}$, we can represent (4.8) as listed in (4.38). This can be seen by following the same approach as the previous subsection. To see this, note that with $\mathcal{B}^2 = (I - \mathcal{A})^2$ to get

$$\mathcal{Z}_i = (2\mathcal{A} - \mathcal{A}^2)\mathcal{Z}_{i-1} + \mathcal{A}\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big) \tag{4.36}$$

By multiplying $\mathcal{A}$ to both sides of the previous equation and introducing $\mathcal{X}_i \overset{\Delta}{=} \mathcal{A}\mathcal{Z}_i$ we get

$$\mathcal{X}_i = \mathcal{A}\Big((2I - \mathcal{A})\mathcal{X}_{i-1} + \mathcal{A}\mathcal{W}_{i-1} - \mathcal{A}\mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}(\mathcal{W}_{i-2})\big)\Big) \tag{4.37}$$

Thus from (4.8c) we have $\mathcal{W}_i = \mathbf{prox}_{\mu R}(\mathcal{X}_i)$.

Algorithm: Prox-ATC II

Setting: Initialize $x_{k,-1} = \psi_{k,-1} = 0$ and $w_{k,-1}$ arbitrary. For every agent k, repeat for $i = 0, 1, 2, ...$

$$\psi_{k,i} = 2x_{k,i-1} - \mu\big(\nabla J_k(w_{k,i-1}) - \nabla J_k(w_{k,i-2})\big) \tag{4.38a}$$

$$z_{k,i} = \psi_{k,i} - \sum_{s \in \mathcal{N}_k} a_{sk}(x_{s,i-1} - w_{s,i-1} + w_{s,i-2}) \qquad \textbf{(Communication step)} \tag{4.38b}$$

$$x_{k,i} = \sum_{s \in \mathcal{N}_k} a_{sk} z_{s,i} \qquad \textbf{(Communication step)} \tag{4.38c}$$

$$w_{k,i} = \mathbf{prox}_{\mu R}(x_{k,i}) \tag{4.38d}$$

4.C Proof Theorem 4.1

Squaring both sides of (4.11a) and (4.11b) we get

$$\begin{aligned}\|\widetilde{z}_i\|^2 &= \|(I-\mathcal{C})\widetilde{w}_{i-1} - \mu\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big)\|^2 + \|\mathcal{B}\widetilde{y}_{i-1}\|^2 \\ &\quad - 2\widetilde{y}_{i-1}^{\mathsf{T}}\mathcal{B}\left((I-\mathcal{C})\widetilde{w}_{i-1} - \mu\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big)\right)\end{aligned} \tag{4.39}$$

and

$$\begin{aligned}\|\widetilde{y}_i\|^2 = \|\widetilde{y}_{i-1} + \mathcal{B}\widetilde{z}_i\|^2 &= \|\widetilde{y}_{i-1}\|^2 + \|\mathcal{B}\widetilde{z}_i\|^2 + 2\widetilde{y}_{i-1}^{\mathsf{T}}\mathcal{B}\widetilde{z}_i \\ &\overset{(4.11a)}{=} \|\widetilde{y}_{i-1}\|^2 + \|\widetilde{z}_i\|^2_{\mathcal{B}^2} - 2\|\mathcal{B}\widetilde{y}_{i-1}\|^2 \\ &\quad + 2\widetilde{y}_{i-1}^{\mathsf{T}}\mathcal{B}\left((I-\mathcal{C})\widetilde{w}_{i-1} - \mu\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big)\right)\end{aligned} \tag{4.40}$$

Adding equation (4.40) to (4.39) and rearranging, we get

$$\|\widetilde{z}_i\|^2_{\mathcal{Q}} + \|\widetilde{y}_i\|^2 = \|(I-\mathcal{C})\widetilde{w}_{i-1} - \mu\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big)\|^2 + \|\widetilde{y}_{i-1}\|^2 - \|\mathcal{B}\widetilde{y}_{i-1}\|^2 \tag{4.41}$$

where $\mathcal{Q} = I - \mathcal{B}^2$ is positive definite from (4.12a). Since $y_0 = 0$ and $y_i = y_{i-1} + \mathcal{B}z_i$, we know $y_i \in \text{range}(\mathcal{B})$ for any i. Thus, both y_i and $y_b^\star$ lie in the range space of $\mathcal{B}$, and it holds that $\|\mathcal{B}\widetilde{y}_{i-1}\|^2 \geq \underline{\sigma}(\mathcal{B}^2)\|\widetilde{y}_{i-1}\|^2$. Therefore, we can bound (4.41) by

$$\|\widetilde{z}_i\|^2_{\mathcal{Q}} + \|\widetilde{y}_i\|^2 \leq \|\widetilde{w}_{i-1} - \mu\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{w}_{i-1}\big)\|^2 + (1 - \underline{\sigma}(\mathcal{B}^2))\|\widetilde{y}_{i-1}\|^2 \tag{4.42}$$

Also, since $\mathcal{J}(w) + \frac{1}{2\mu}\|w\|^2_{\mathcal{C}}$ is $\delta_\mu = \delta + \frac{1}{\mu}\sigma_{\max}(\mathcal{C})$-smooth, it holds from (1.8) that:

$$\|\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{w}_{i-1}\big)\|^2 \leq \delta_\mu \widetilde{w}_{i-1}^{\mathsf{T}}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{w}_{i-1}\big) \tag{4.43}$$

Using this bound, it can be easily verified that:

$$
\begin{aligned}
&\|\widetilde{w}_{i-1} - \mu\big(\nabla \mathcal{J}(w_{i-1}) - \nabla \mathcal{J}(w^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{w}_{i-1}\big)\|^2 \\
&\leq \|\widetilde{w}_{i-1}\|^2 - \mu(2-\mu\delta_\mu)\widetilde{w}_{i-1}^\mathsf{T}\big(\nabla \mathcal{J}(w_{i-1}) - \nabla \mathcal{J}(w^\star) + \frac{1}{\mu}\mathcal{C}\widetilde{w}_{i-1}\big) \\
&\leq \big(1-\mu\nu(2-\mu\delta_\mu)\big)\|\widetilde{w}_{i-1}\|^2 = \big(1-\mu\nu(2-\sigma_{\max}(\mathcal{C})-\mu\delta)\big)\|\widetilde{w}_{i-1}\|^2
\end{aligned} \tag{4.44}
$$

where in the last step we used the fact that $2-\mu\delta_\mu \geq 0$, which follows from the condition $\mu < (2-\sigma_{\max}(\mathcal{C}))/\delta$, and the fact that $\mathcal{J}(w) + \frac{1}{2\mu}\|w\|^2_{\mathcal{C}}$ is ν-strongly convex. Thus, we can substitute the previous inequality in (4.42) and get

$$
\|\widetilde{z}_i\|^2_{\mathcal{Q}} + \|\widetilde{y}_i\|^2 \leq \big(1-\mu\nu(2-\sigma_{\max}(\mathcal{C})-\mu\delta)\big)\|\widetilde{w}_{i-1}\|^2 + (1-\underline{\sigma}(\mathcal{B}^2))\|\widetilde{y}_{i-1}\|^2 \tag{4.45}
$$

From (4.11c) and the nonexpansive property of the proximal operator, we have

$$
\|\widetilde{w}_i\|^2 = \|\mathbf{prox}_{\mu\mathcal{R}}\big(\bar{\mathcal{A}}z_i\big) - \mathbf{prox}_{\mu\mathcal{R}}(\bar{\mathcal{A}}z^\star)\|^2 \leq \|\bar{\mathcal{A}}\widetilde{z}_i\|^2 \leq \|\widetilde{z}_i\|^2_{\mathcal{Q}} \tag{4.46}
$$

where the last step holds because of condition (4.12b) so that $\|\bar{\mathcal{A}}\widetilde{z}_i\|^2 = \|\widetilde{z}_i\|^2_{\bar{\mathcal{A}}^2} \leq \|\widetilde{z}_i\|^2_{\mathcal{Q}}$. Substituting (4.46) into (4.45) we reach our result. Finally we note that:

$$
\big(1-\mu\nu(2-\sigma_{\max}(\mathcal{C})-\mu\delta)\big) < 1 \iff \mu < \frac{2-\sigma_{\max}(\mathcal{C})}{\delta} \tag{4.47}
$$

4.D Proximal mapping of (4.19)

To rewrite the non-smooth terms (4.19) more compactly, we introduce

$$D_1 \triangleq \begin{bmatrix} \sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 & -1 & 0 \end{bmatrix} \in \mathbb{R}^{\frac{M}{2}\times M} \tag{4.48}$$

$$D_2 \triangleq \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & -1 \end{bmatrix} \in \mathbb{R}^{\frac{M}{2}\times M} \tag{4.49}$$

and $b_1 \triangleq e_1$ where e_1 is the first column of the identity matrix $I_{M/2}$. With D_1, D_2 and b_1, we can rewrite $R_1(w)$ and $R_2(w)$ in (4.19) as

$$R_1(w) = \|D_1 w - b_1\|_1, \quad R_2(w) = \|D_2 w\|_1. \tag{4.50}$$

Let us introduce $g(w) = \|w\|_1$ so that $R_1(w) = g(D_1 w - b_1)$ and $R_2(w) = g(D_2 w)$. It can be verified that $D_1 D_1^\mathsf{T} = 2I$ and $D_2 D_2^\mathsf{T} = 2I$. Thus, from [135, Theorem 6.15] it holds that

$$\mathbf{prox}_{\mu R_1}(w) = w + \frac{1}{2\mu} D_1^\mathsf{T}[\mathbf{prox}_{2\mu^2 g}(\mu D_1 w - \mu b_1) - \mu D_1 w + \mu b_1], \tag{4.51a}$$

$$\mathbf{prox}_{\mu R_2}(w) = w + \frac{1}{2\mu} D_2^\mathsf{T}[\mathbf{prox}_{2\mu^2 g}(\mu D_2 w) - \mu D_2 w]. \tag{4.51b}$$

In other words, both $\mathbf{prox}_{\mu R_1}(w)$ and $\mathbf{prox}_{\mu R_2}(w)$ have closed forms which are easy to calculate since $\mathbf{prox}_{\kappa g}(w) = \mathrm{col}\{\mathrm{sgn}(w_i)\max\{|w_i| - \kappa, 0\}\} \in \mathbb{R}^M$.

CHAPTER 5

Multi-Coupled Consensus Problem

In Chapters 3 and 4, we studied decentralized optimization problems where all agents need to agree on one common variable. However, there exist many scenarios where the global cost function may involve multiple variables, and, moreover, each local cost may be a function of only a subset of these variables. This situation motivates us to study in this chapter a broader problem, where each local cost contains multiple variables that get to be chosen by the network cooperatively. Examples of applications where this general scenario arises include web page categorization [136], web-search ranking [137], minimum-cost flow problems [138], decentralized model predictive control in smart energy systems [80], decentralized wireless acoustic sensor networks [139], decentralized wireless localization [140], and decentralized power systems monitoring [141].

5.1 Problem Set-Up

Consider a network of K agents that are connected through some network topology. Assume we have L vector variables denoted by $\{w^1, \cdots, w^L\}$, where $w^\ell \in \mathbb{R}^{M_\ell}$. Let $w \triangleq \mathrm{col}\{w^1, w^2, ..., w^L\} \in \mathbb{R}^M$ denote the $L \times 1$ block column vector formed by collecting all those variables. The partitioning is used to represent the possibility of multiple independent arguments for the cost functions. Without loss of generality, we assume that the variables $\{w^\ell\}$ are distinct in that they do not share common entries. Let $\mathcal{I}_k$ denote the set of variable indices that affect the cost of agent k – Figure 5.1 illustrates this situation for a simple

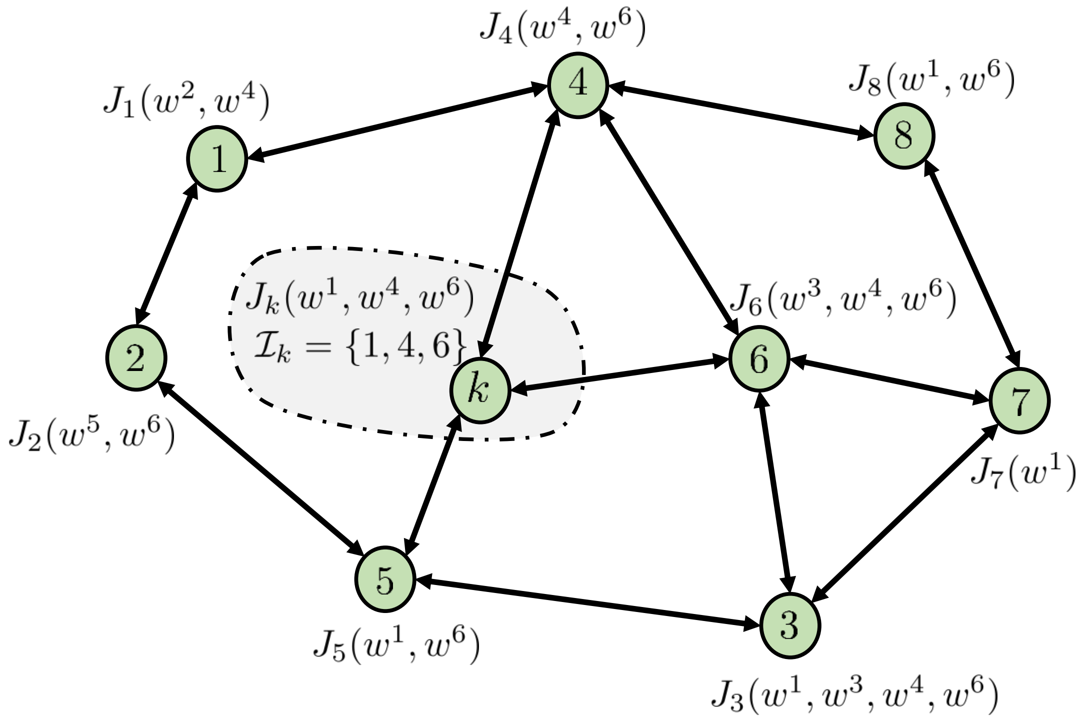


Figure 5.1: A connected network of agents where different agents generally depend on different subsets of parameter vectors. For this example, we have $w = [w^1, w^2, w^3, w^4, w^5, w^6]$.

network. If we let w_k denote the components of w that affect this same agent:

$$w_k \triangleq \text{col}\{w^\ell\}_{\ell \in \mathcal{I}_k} \in \mathbb{R}^{Q_k}, \quad Q_k \triangleq \sum_{\ell \in \mathcal{I}_k} M_\ell. \tag{5.1}$$

Then we are interested in determining the solution of the following optimization problem:

$$\underset{w^1, w^2, \cdots, w^L}{\text{minimize}} \quad \sum_{k=1}^{K} J_k(w_k), \quad w_k = \text{col}\{w^\ell\}_{\ell \in \mathcal{I}_k} \tag{5.2}$$

where $J_k(w_k) : \mathbb{R}^{Q_k} \to \mathbb{R}$ is a local cost function known by agent k. In this formulation, each agent depends only on part of the decision vector $w = \text{col}\{w^1, w^2, ..., w^L\} \in \mathbb{R}^M$, namely, $w_k = \text{col}\{w^\ell\}_{\ell \in \mathcal{I}_k} \in \mathbb{R}^{Q_k}$.

Assumption 5.1. (Cost functions) *: Each cost function $J_k(w_k)$ is δ-smooth and the aggregate cost $\frac{1}{K}\sum_{k=1}^{K} J_k(w_k)$ is $\bar{\nu}$-strongly-convex cost for some $\delta \geq \bar{\nu} > 0$.* □

Under the above condition, a unique solution $w^\star$ exists. We denote its block entries by

$$w^\star = \mathrm{col}\{w^{1,\star}, \cdots, w^{L,\star}\} \triangleq \arg\min_{w^1,\cdots,w^L} \sum_{k=1}^{K} J_k(w_k) \tag{5.3}$$

We note that algorithms that solve (1.1) can be used to solve (5.2). For example, this can be achieved by extending each local variable w_k into the longer global variable w. However, this solution method would require unnecessary communications and memory allocation. This is because in (5.2) each local function contains only a subset of the global variable w. Therefore, solving (5.2) directly and more effectively is important for large scale networks. Conversely, we also note that algorithms that solve (5.2) are more general and can be used to solve (1.1). To see this, let $L = 1$ and $\mathcal{I}_k = \{L\}$, then problem (5.2) will depend only on one variable $w = w^L$. In this case, the cost function becomes $\sum_{k=1}^{K} J_k(w)$, which is of the same exact form as problem (1.1). For sparse networks with a large number of parameters to estimate, it is much more efficient to devise decentralized techniques that solve (5.2) *directly* rather than transform (5.2) into the form in (1.1) via vector extension. We will show that this extension technique not only increases complexity but it often degrades convergence performance as well. Therefore, it is desirable to address the solution of problem (5.2) directly.

Remark 5.1. (Useful case): We illustrate a special case of (5.2), which is common in many applications. Consider the scenario where every agent wants to estimate its own variable w^k and is coupled with every neighboring agent, i.e., $L = K$ and $\mathcal{I}_k = \mathcal{N}_k$ so that $w_k = \mathrm{col}\{w^\ell\}_{\ell \in \mathcal{N}_k}$, where $\mathcal{N}_k$ denotes the neighborhood of agent k (including agent k). To explicitly indicate that w_k and the corresponding constraint depend exclusively on the neighborhood variables, we let

$$w_{\mathcal{N}_k} \triangleq w_k, \quad \mathbb{W}_{\mathcal{N}_k} \triangleq \mathbb{W}_k \tag{5.4}$$

Then, problem (5.2) becomes

$$\min_{w} \quad \sum_{k=1}^{K} J_k(w_{\mathcal{N}_k}), \text{ s.t.} \quad w \in \mathbb{W}_{\mathcal{N}_1} \cap \cdots \cap \mathbb{W}_{\mathcal{N}_K} \tag{5.5}$$

Many important applications fit into problem (5.5). For, instance, such formulations arise in wireless localization where each agent aims to estimate its position based on distance measurements from its neighbors. Two other examples are decentralized model predictive control [142] and minimum cost flow problems [138]. □

Example 5.1. (Power system state estimation): We describe one example in power system state estimation [141], which is a special case of formulation (5.2). Thus, consider a system consisting of K interconnected sub-systems with each sub-system consisting of some subset of buses (or edges). Let $w = [w^1, \cdots, w^L]$ denote the state of the system (e.g., voltages and currents across all buses). Suppose each subsystem collects measurements related to the voltages and currents across its local buses and voltages and currents across the interconnection between neighboring sub-systems (see Figure 5.2).

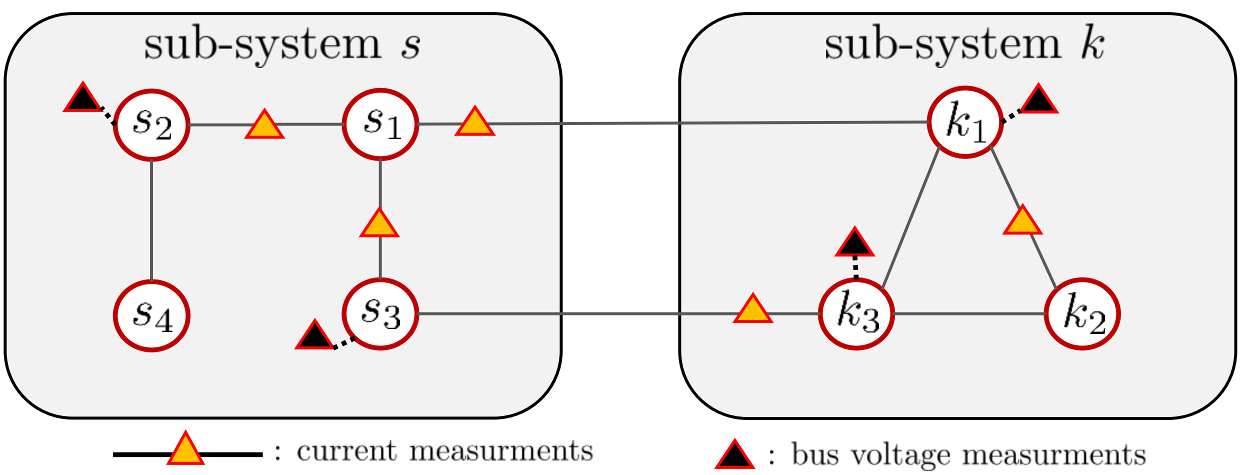


Figure 5.2: Two neighboring sub-systems sharing states across their interconnection, i.e., buses k_1-s_1 and k_3-s_3.

We let w_k denote a vector that collects the states $\{w^\ell\}$ of system k (i.e., voltages and currents across the buses of system k and across the buses to its neighboring subsystems).

Then, the goal of each subsystem is to estimate the states w_k from S_k observations:

$$y_k = H_k w_k + v_k \tag{5.6}$$

where $H_k \in \mathbb{R}^{S_k \times Q_k}$ is the measurement matrix and $v_k \in \mathbb{R}^{S_k}$ is a zero-mean measurement noise with known covariance matrix. One way to estimate the $\{w_k\}$ is by solving the following problem:

$$\min_{w} \quad \sum_{k=1}^{K} \|y_k - H_k w_k\|^2, \text{ s.t.} \quad w \in \mathbb{W} \tag{5.7}$$

where $\{\mathbb{W}\}$ are a convex sets that capture some prior information about about the $\{w\}$. Now, since neighboring agents measure some similar quantities across their interconnections, it holds that w_k and w_s partially overlap if $s \in \mathcal{N}_k$ and hence this problem is a special case of (5.2). □

5.1.1 Related works

Problems of the form (5.2) have received less attention in the literature compared to the case of one consensus variable. Dual methods have been used to solve (5.2) or its special case (5.5) in [141, 143–147]. For example, to solve (5.5) in [144, 145] ADMM methods are employed, while the works [146, 147] propose other dual decomposition techniques. The work [141] applies an ADMM method to solve a decentralized power system state estimation problem of the form (5.2). The work [143] solves (5.2) by employing an extended ADMM method to reduce communications at the expense of some stronger assumptions. In all of these methods, a second auxiliary (sub-minimization) problem needs to be solved at each iteration, which requires an inner iteration unless a closed form solution exists. In [148] a quadratic problem is solved, where every agent has their own variable w^k (i.e, $L = K$) and the agents are coupled through linear constraints with neighboring node's variables $\{w^\ell\}_{\ell \in \mathcal{N}_k}$. Moreover, it is further assumed that the agents involved in a constraint are fully connected, i.e., they can communicate directly.

Since in problem (5.2) different agents are influenced by different block vectors w^ℓ, the network will be divided into overlapping clusters and each cluster ℓ will involve the agents that need to agree on w^ℓ – see Equation (5.9). Similar clustering was used in [143] where the ADMM method was employed with identical penalty factors across all clusters. There are works that deal with problem (1.1) where all agents need to agree on the same w, however to reduce communication, different agents transmit different blocks $\{w^\ell\}$ at each time instant (see e.g. [149]). In this case, each cluster involves different agents at each time instant and, over time, all agents will be involved in all clusters. Note also that the group diffusion algorithm used in [150] deals with the problem where each agent is interested in its own minimizer $w_k^\bullet = \arg\min_w J_k(w)$ and can solve the problem individually but cooperation is used since the estimation accuracy can be enhanced by cooperation if part of the minimizers $\{w_k^\bullet\}$ are common across neighbors. To take advantage of this overlap, each agent assigns different weights to different blocks w^ℓ.

5.1.2 Contribution

In this chapter, we develop a first-order method for solving (5.2), which unlike the methods from [141, 143] does not require inner minimization steps. More importantly, we analytically show that algorithms that ignore the problem structure can be detrimental to its performance compared to algorithms that exploit such structure. Such conclusion have been observed in [143] but no analytical explanation was provided. We analytically explain how the performance of an algorithm is effected if the sparsity structure in (5.2) is ignored.

5.2 Problem Reformulation for Decentralized Solution

In order to solve (5.2) in a decentralized manner, we first need to adjust the notation to account for one additional degree of freedom. Recall from (5.2) that the costs of two different agents, say, agents k and s, may depend on the same sub-vector, say, w^ℓ. Since these two agents will be learning w^ℓ over time, each one of them will have its own local estimate for w^ℓ.

We create a local copy of w^ℓ at agent k denoted by w_k^ℓ and to the copy of w^ℓ at agent s by w_s^ℓ. With this in mind, recall that we denoted the collection of all sub-vectors that influence agent k by w_k; defined earlier in (5.1). In that definition, the sub-vectors $\{w^\ell\}$ influencing $J_k(\cdot)$ were used to construct w_k. In view of the new notation using virtual copies, we now redefine the same w_k using the local copies instead, namely, we now write

$$w_k \;\; \stackrel{\Delta}{=} \;\; \text{col}\left\{w_k^\ell\right\}_{\ell \in \mathcal{I}_k} \; \in \mathbb{R}^{Q_k} \tag{5.8}$$

where $w_k^\ell \in \mathbb{R}^{M_\ell}$ is the local copy of the variable w^ℓ at agent k. We further let $\mathcal{C}_\ell$ denote the cluster of nodes that contains the variable w^ℓ in their costs:

$$\mathcal{C}_\ell = \{k \mid \ell \in \mathcal{I}_k\} \tag{5.9}$$

We can view the cluster $\mathcal{C}_\ell$ as a smaller network (or sub-graph) where all agents in this sub-network are interested in the same parameter w^ℓ. To require all local copies w_k^ℓ to coincide with each other, we need to introduce the constraint

$$w_k^\ell = w_s^\ell, \;\; \forall \; k, s \in \mathcal{C}_\ell \tag{5.10}$$

Using relations (5.8) and (5.10), we rewrite problem (5.2) as

$$\begin{aligned} \underset{w_1,\ldots,w_K}{\text{minimize}} \quad & \sum_{k=1}^{K} J_k(w_k), \quad w_k = \text{col}\{w_k^\ell\}_{\ell \in \mathcal{I}_k} \\ \text{subject to} \quad & w_k^\ell = w_s^\ell \;\; \forall\, k, s \in \mathcal{C}_\ell, \; \forall\, \ell \in \{1, \cdots, L\}. \end{aligned} \tag{5.11}$$

For ease of reference, we summarize the main notation used in this chapter in following Table 5.1.

Table 5.1: A listing of the main symbols used in this chapter

Symbol	Meaning
$\mathcal{I}_k$	The set of variable indices that influence the cost of agent k.
w_k^ℓ	Local copy of w^ℓ at agent k.
w_k	Collection of parameters influencing agent k, $w_k \overset{\Delta}{=} \text{col}\{w_k^\ell\}_{\ell\in\mathcal{I}_k}$
$\mathcal{C}_\ell$	Cluster of nodes that are influenced by the variable w^ℓ.
K_ℓ	Number of agents in cluster $\mathcal{C}_\ell$.
$\mathcal{W}^\ell$	Stacks all local copies of w^ℓ across $\mathcal{C}_\ell$, $\mathcal{W}^\ell = \text{col}\{w_k^\ell\}_{k\in\mathcal{C}_\ell}$
$\mathcal{W}$	Stacks $\mathcal{W}^\ell$ for all parameters, $\mathcal{W} = \text{col}\{\mathcal{W}^\ell\}_{\ell=1}^L$

5.3 Cluster Combination Matrices

To solve (5.11), we associate weights $\{a_{\ell,sk}\}_{s,k\in\mathcal{C}_\ell}$ with each cluster $\mathcal{C}_\ell$ and and introduce the cluster combination matrix

$$A_\ell = [a_{\ell,sk}]_{s,k\in\mathcal{C}_\ell} \tag{5.12}$$

Assumption 5.2. (Each cluster is a connected sub-graph): *The neighboring agents can communicate in both directions and the doubly-stochastic matrix A_ℓ is primitive and symmetric. This implies that for any two arbitrary agents in cluster $\mathcal{C}_\ell$, there exists at least one path with nonzero weights $\{a_{\ell,sk}\}_{s,k\in\mathcal{C}_\ell}$ linking one agent to the other.* □

We remark that two agents are coupled if they share the same variable w^ℓ and we are only requiring the coupled agents to be connected. If all agents share the entire w, then all agents are coupled by w and the above assumption translates into requiring the network to be strongly-connected. Assumption 5.2 is satisfied for most networks of interest. For example, all applications that fit into problem (5.5) given in Remark 5.1 naturally satisfy this assumption, including but not limited to applications in decentralized power system monitoring, decentralized control, and maximum-flow — see [141,144,148]. This is due to the construction of problem (5.5): there exist $L = K$ clusters, where w^ℓ affects the neighborhood of agent $k = \ell$, and hence $\mathcal{C}_k = \mathcal{N}_k$ forms a star shaped graph (i.e., all agents $s \in \mathcal{C}_k$ are connected through agent k), and hence this cluster is connected. Moreover, multitask applications satisfy this assumption [136, 137, 151, 152].

We emphasize that Assumption 5.2 is not limited to the case described in Remark 5.1 since it can be satisfied for any connected network, as we further clarify. To being with, independently of the clusters, let us assume that the entire network is connected. Now, if some cluster $\mathcal{C}_\ell$ happens to be unconnected, we can embed it into a larger *connected* cluster $\mathcal{C}'_\ell$ such that $\mathcal{C}_\ell \subset \mathcal{C}'_\ell$. For example, consider the network shown in Figure 5.3. In this network,

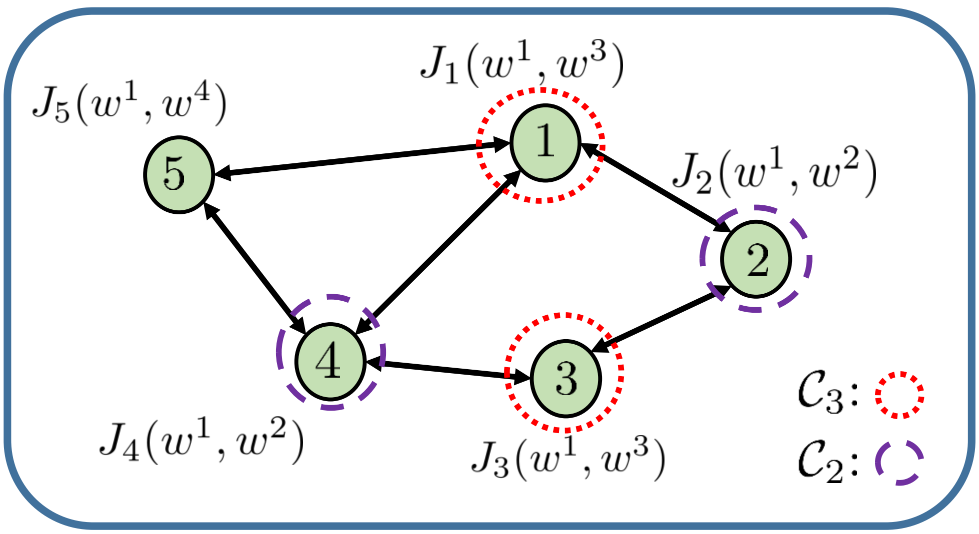


Figure 5.3: A five-agent network with unconnected $\mathcal{C}_2$ and $\mathcal{C}_3$.

we have

$$\mathcal{C}_1 = \{1, 2, 3, 4, 5\},\ \mathcal{C}_2 = \{2, 4\},\ \mathcal{C}_3 = \{1, 3\},\ \mathcal{C}_4 = \{5\} \tag{5.13}$$

In these clusters, we find that $\mathcal{C}_4$ is a singleton. Therefore, w^4 will be optimized solely and separately by agent 5, and no communication is needed for that variable. Cluster $\mathcal{C}_1$ is connected, and agents $\{1, 2, 3, 4, 5\}$ cooperate in order to optimize w^1, with each agent sharing its estimate with neighbors. However, clusters $\mathcal{C}_2$ and $\mathcal{C}_3$ have disconnected graphs. This implies that agents 2 and 4 cannot communicate to optimize and reach consensus on w^2. Likewise, for agents $\{1, 3\}$ regarding the variable w^3. To circumvent this issue, we redefine

$J_1(w^1, w^3)$ and $J_2(w^1, w^2)$ as:

$$J_1'(w^1, w^2, w^3) \triangleq J_1(w^1, w^3) + 0 \cdot w^2 \tag{5.14}$$

$$J_2'(w^1, w^2, w^3) \triangleq J_2(w^1, w^2) + 0 \cdot w^3 \tag{5.15}$$

By doing so, the augmented costs $J_1'(w^1, w^2, w^3)$ and $J_2'(w^1, w^2, w^3)$ now involve w^2 and w^3, respectively, and the new clusters become

$$\mathcal{C}_2' = \{1, 2, 4\}, \quad \mathcal{C}_3' = \{1, 2, 3\} \tag{5.16}$$

which are connected and satisfy $\mathcal{C}_2 \subset \mathcal{C}_2'$ and $\mathcal{C}_3 \subset \mathcal{C}_3'$. Therefore, in this scenario, agents $\{1, 2, 4\}$ will now cooperate to optimize w^2 with agent 1 acting as a connection that allows information about w^2 to diffuse in the cluster. Likewise, for agents $\{1, 2, 3\}$, with agent 2 allowing information about w^3 to diffuse in the cluster. A second extreme approach would be to extend each local variable w_k to the global variable w, which reduces problem (5.2) to the formulation (1.1). This way of embedding the clusters into larger connected clusters can be done in a decentralized fashion – see for example [143].

5.4 Coupled Exact Diffusion

In this chapter, for illustration and simplicity we focus on one specific algorithm. In particular, we will generalize the exact diffusion algorithm (3.17) to the case of multiple coupled parameters.

Similar to the arguments in the previous chapters, we utilize Lemma 1.1 to rewrite the constraints in (5.11) in an equivalent form that is amenable to decentralized implementations. First, for each parameter vector w^ℓ, we collect its local copies into the extended vector

$$\mathcal{W}^\ell \triangleq \mathrm{col}\{w_k^\ell\}_{k \in \mathcal{C}_\ell} \in \mathbb{R}^{M_\ell K_\ell} \tag{5.17}$$

where $K_\ell = |\mathcal{C}_\ell|$ denotes the number of agents in cluster $\mathcal{C}_\ell$. With this notation, we can rewrite the cost function in problem (5.11) as

$$\mathcal{J}(\mathcal{w}^1, \mathcal{w}^2, \cdots, \mathcal{w}^L) \triangleq \sum_{k=1}^{K} J_k(w_k). \tag{5.18}$$

Recall that each cluster is associated with a primitive doubly stochastic symmetric matrix A_ℓ. If we define

$$\mathcal{A}_\ell \triangleq A_\ell \otimes I_{M_\ell} \tag{5.19}$$

$$\mathcal{B}_\ell \triangleq \frac{1}{2}(I_{M_\ell K_\ell} - \mathcal{A}_\ell)^{\frac{1}{2}} \tag{5.20}$$

Then using Lemma 1.1 and the definition of $\mathcal{w}^\ell$ in (5.17) we get

$$w_k^\ell = w_s^\ell,\ \forall\ k, s \in \mathcal{C}_\ell \Longleftrightarrow \mathcal{B}_\ell \mathcal{w}^\ell = 0, \quad \forall\ \ell. \tag{5.21}$$

Using relations (5.18) and (5.21), we can rewrite problem (5.11) equivalently as

$$\begin{aligned} \underset{\mathcal{w}^1,\ldots,\mathcal{w}^L}{\text{minimize}} \quad & \mathcal{J}(\mathcal{w}^1, \cdots, \mathcal{w}^L) \\ \text{subject to} \quad & \mathcal{B}_\ell \mathcal{w}^\ell = 0 = 0,\ \forall\ \ell \end{aligned} \tag{5.22}$$

To rewrite problem (5.22) more compactly, we introduce

$$\mathcal{B} \triangleq \text{blkdiag}\{\mathcal{B}_\ell\}_{\ell=1}^{L} = \frac{1}{2}(I_S - \mathcal{A}) \tag{5.23}$$

where $S \triangleq \sum_{\ell=1}^{L} K_\ell M_\ell$ and

$$\mathcal{A} \triangleq \text{diag}\{\mathcal{A}_\ell\}_{\ell=1}^{L}. \tag{5.24}$$

We further define the following

$$\mathcal{W} \overset{\Delta}{=} \text{col}\{\mathcal{W}^\ell\}_{\ell=1}^L \in \mathbb{R}^S \tag{5.25}$$

$$\mathcal{J}(\mathcal{W}) \overset{\Delta}{=} \mathcal{J}(\mathcal{W}^1, \cdots, \mathcal{W}^L) \tag{5.26}$$

Then, problem (5.22) becomes:

$$\underset{\mathcal{W}}{\text{minimize}} \quad \mathcal{J}(\mathcal{W}), \text{ s.t. } \mathcal{B}\mathcal{W} = 0 \tag{5.27}$$

The above problem is simply a linearly constrained problem. Therefore, similar to how (3.17) is derived we can employ the following adapt-then-combine algorithm:

$$\begin{cases} \mathcal{Z}_i = \mathcal{W}_{i-1} - \mu \nabla \mathcal{J}(\mathcal{W}_{i-1}) - \mathcal{B}\mathcal{Y}_{i-1} & \textbf{(primal-descent)} & \text{(5.28a)} \\ \mathcal{Y}_i = \mathcal{Y}_{i-1} + \mathcal{B}\mathcal{Z}_i & \textbf{(dual-ascent)} & \text{(5.28b)} \\ \mathcal{W}_i = 0.5(I + \mathcal{A})\mathcal{Z}_i & \textbf{(Combine)} & \text{(5.28c)} \end{cases}$$

By eliminating the dual variable we get

$$\mathcal{W}_i = 0.5(I + \mathcal{A})\Big(2\mathcal{W}_{i-1} - \mathcal{W}_{i-2} - \mu\big(\nabla \mathcal{J}_{\mathcal{W}}(\mathcal{W}_{i-1}) - \nabla \mathcal{J}_{\mathcal{W}}(\mathcal{W}_{i-2})\big)\Big) \tag{5.29}$$

Algorithm (5.29) looks similar to the one in (3.17). However, there are two subtle differences. First, the combination matrix $\mathcal{A} = \text{diag}\{\mathcal{A}_\ell\}_{\ell=1}^L$ has a block diagonal structure and, second, the gradient $\nabla_{\mathcal{W}} \mathcal{J}(\mathcal{W})$ couples the variables $\{\mathcal{W}^\ell\}$ across the clusters. To see this, we note that the gradient vector is given by

$$\nabla_{\mathcal{W}} \mathcal{J}(\mathcal{W}) = \begin{bmatrix} \nabla_{\mathcal{W}^1} \mathcal{J}(\mathcal{W}) \\ \vdots \\ \nabla_{\mathcal{W}^L} \mathcal{J}(\mathcal{W}) \end{bmatrix}, \quad \nabla_{\mathcal{W}^\ell} \mathcal{J}(\mathcal{W}) = \text{col}\{\nabla_{w_k^\ell} J_k(w_k)\}_{k \in \mathcal{C}_\ell} \tag{5.30}$$

It is clear that each block $\text{col}\{\nabla_{w_k^\ell} J_k(w_k)\}_{k \in \mathcal{C}_\ell}$ depends on other clusters since the argument in $J_k(w_k)$ is w_k and agent k may belong to more than one cluster. For the special case that

Algorithm (Coupled Exact Diffusion Strategy)

Setting: Let $\bar{A}_\ell = (I_{K_\ell} + A_\ell)/2$, and $w_{k,-1} = \psi_{k,-1}$ arbitrary. For every agent k, repeat for $i = 0, 1, 2, ...$

$$\psi_{k,i} = w_{k,i-1} - \mu \nabla_{w_k} J_k(w_{k,i-1}) \tag{5.31a}$$

$$\phi_{k,i} = \psi_{k,i} + w_{k,i-1} - \psi_{k,i-1} \tag{5.31b}$$

$$w^\ell_{k,i} = \sum_{s \in \mathcal{N}_k \cap \mathcal{C}_\ell} \bar{a}_{\ell,sk} \phi^\ell_{s,i}, \;\; \forall\, \ell \in \mathcal{I}_k \tag{5.31c}$$

---

there exists only one cluster (i.e, $L = 1$, $w_k = w_k^1$, and $\mathcal{A} = \mathcal{A}_1$), we recover the exact diffusion algorithm (3.17). The decentralized implementation of (5.29) is listed in (5.31a)–(5.31c). In this listing, the variables $\{\psi_{k,i}, \phi_{k,i}\}$ have the same structure as $w_{k,i}$, i.e., $\psi_{k,i} = \mathrm{col}\{\psi^\ell_{k,i}\}_{\ell \in \mathcal{I}_k}$ and $\phi_{k,i} = \mathrm{col}\{\phi^\ell_{k,i}\}_{\ell \in \mathcal{I}_k}$.

Corollary 5.1. (Coupled Exact-Diffusion) *Let Assumptions 5.1 and 5.2 hold. If the step-size satisfies $\mu < \frac{2}{\delta}$, then it holds that each sub-block $w^\ell_{k,i}$ in 5.31 converges linearly to the optimal corresponding $w^{\ell,\star}$ defined in* (5.3) *with rate*

$$\gamma = \max\left\{1 - \mu\bar{\nu}_\rho(2 - \mu\delta),\ 1 - \underline{\sigma}^2(\mathcal{B}),\ \mu\rho(2 - \mu\delta)\right\} < 1 \tag{5.32}$$

for any $0 < \rho < 1/\mu(2 - \mu\delta)$ with $\bar{\nu}_\rho$ denoting the strongly-convex parameter given in (3.31) *with $\mathcal{B}$ defined in* (5.23). □

The proof of Corollary 5.1 directly follows from the proof of Theorem 3.1. Notice how the convergence rate γ (5.32) depends on matrix $\mathcal{B}$. Recall from (5.23) that in this chapter $\mathcal{B} = \mathrm{blkdiag}\{0.5(I_{M_\ell K_\ell} - \mathcal{A}_\ell)^{\frac{1}{2}}\}$ has a block diagonal structure. Recall further that $\underline{\sigma}^2(\mathcal{B})$ denote the smallest non-zero singular value (or eigenvalue) of the matrix $\mathcal{B} = \mathrm{blkdiag}\{0.5(I_{M_\ell K_\ell} - \mathcal{A}_\ell)\}$. Thus, the effect of the clusters on the convergence rate is evident through the term $1 - \underline{\sigma}^2(\mathcal{B}) = 1 - \min\{\underline{\sigma}(\mathcal{B}^2_\ell)\}$. We see that in this case, the convergence rate is affected by the clusters through the matrices $\mathcal{B}^2_\ell = \{0.5(I - A_\ell)\}$. This means that only the connectivity of each cluster affects the convergence rate and not the connectivity of the whole network. Therefore, for sparse networks but well connected clusters an algorithm that does not exploit the structure will perform poorly. This is illustrated in the next section by means of simulations.

5.5 Simulation Results

In this section we illustrate the operation of the algorithm by considering a situation in which the individual costs are quadratic. Each agent k seeks to estimate its own variable $w^k \in \mathbb{R}^{M_k}$ but is affected by the neighboring variables $\{w^\ell; \ell \in \mathcal{N}_k\}$ (i.e., $L = N$ and $\mathcal{I}_k = \mathcal{N}_k$), through a cost of the form:

$$\begin{aligned} J_k(w_k) &= w_k^\mathsf{T} R_k w_k + b_k^\mathsf{T} w_k + r_k \\ &= \sum_{s\in\mathcal{N}_k}\sum_{\ell\in\mathcal{N}_k} (w^s)^\mathsf{T} R_{k,s\ell} w^\ell + \sum_{\ell\in\mathcal{N}_k} b_{k,\ell}^\mathsf{T} w^\ell + r_k \end{aligned} \tag{5.33}$$

where $w_k \triangleq \mathrm{col}\{w^\ell\}_{\ell\in\mathcal{N}_k}$, R_k is a $Q_k \times Q_k$ positive definite matrix, and $b_k \in \mathbb{R}^{Q_k}$. We partition R_k and b_k into block matrices $\{R_{k,s\ell} \in \mathbb{R}^{M_s\times M_\ell}\}$ and block vectors $\{b_{k,\ell} \in \mathbb{R}^{M_\ell}\}$ according to the block structure of w_k. Each agent k only knows its local quantities $\{R_k, b_k\}$. The aggregate cost is given by

$$J^{\mathrm{glob}}(w) \triangleq \sum_{k=1}^{K} J_k(w_k) = w^\mathsf{T}\mathcal{R}w + \bar{b}^\mathsf{T} w \tag{5.34}$$

where

$$\mathcal{R} \triangleq \begin{bmatrix} \sum_{k=1}^{N} R_{k,11} & \cdots & \sum_{k=1}^{N} R_{k,1L} \\ \vdots & & \vdots \\ \sum_{k=1}^{N} R_{k,L1} & \cdots & \sum_{k=1}^{N} R_{k,LL} \end{bmatrix}, \tag{5.35}$$

$$\bar{b} \triangleq \begin{bmatrix} \sum_{k=1}^{N} b_{k,1} \\ \vdots \\ \sum_{k=1}^{N} b_{k,L} \end{bmatrix} \tag{5.36}$$

with

$$R_{k,s\ell} = 0, \;\; b_{k,\ell} = 0, \;\; \text{if } \ell \notin \mathcal{N}_k \text{ or } s \notin \mathcal{N}_k \tag{5.37}$$

In our simulation, we consider a randomly generated network with $K = 20$ agents shown in Figure 6.4a, where neighbors are decided by closeness in distance. We randomly generate R_k and b_k. All combination matrices are generated using the Metropolis rule. Figure Figure 5.5 shows the relative error $(\|\mathcal{W}_i - \mathcal{W}^\star\|^2/\|\mathcal{W}_{-1} - \mathcal{W}^\star\|^2)$ for the coupled exact-diffusion algorithm (5.31) and the exact diffusion algorithm (3.17). In this figure we used $M_\ell = 5$ for all variables and step size $\mu = 0.02$ for both algorithms. We also used the Metropolis rule to create the combination matrices. We conclude that, in the case of problem formulation (5.2), it is not efficient to extend each local vector to the global one and then solve this extended problem [39,93]. Note that in each iteration i each agent k using the exact diffusion algorithm needs to communicate an $5 \times K = 100$ long vector, as opposed to the coupled exact diffusion algorithm where each agent k only communicates $5 \times |\mathcal{N}_k| < 100$.

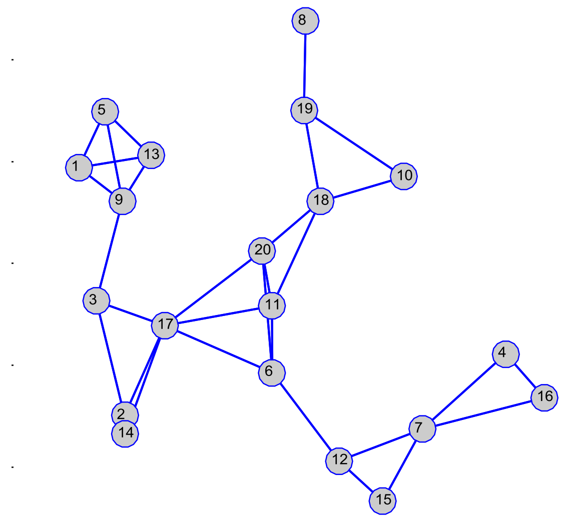


Figure 5.4: Network topology used in the simulation results.

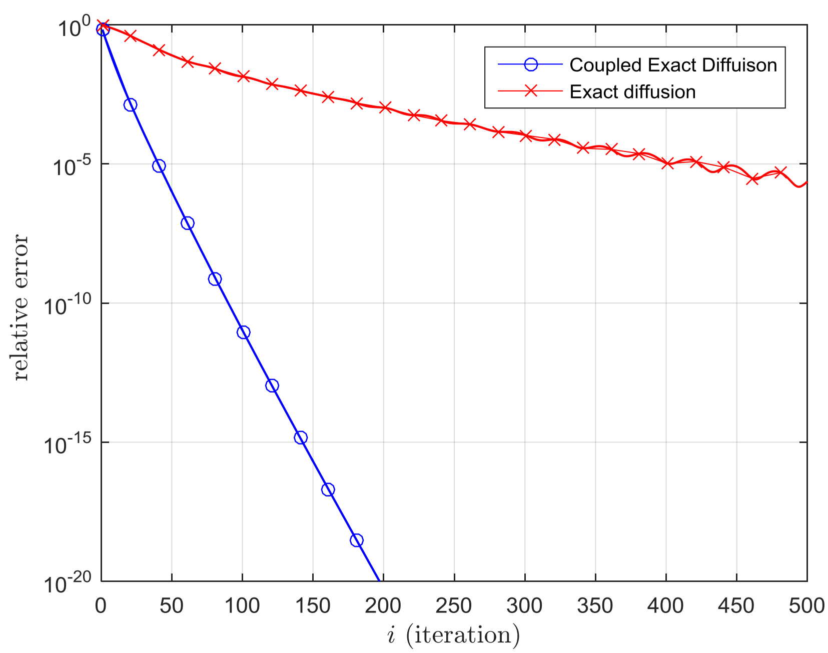


Figure 5.5: Relative errors for the coupled diffusion and the exact diffusion algorithms.

CHAPTER 6

Multi-Coupled Resource Sharing Problem

In this chapter we develop a proximal dual exact diffusion strategy with guaranteed exact convergence for a multi-agent optimization problems with generally coupled affine constraints. Unlike other works, we will show analytically, and by means of simulations, the superior convergence properties of an algorithm that considers the sparsity structure in the constraints compared to others that ignore this structure.

6.1 Motivation

In this section, we will motivate the problem considered in this chapter. Recall from Section 1.3 we briefly described the resource sharing problem (1.2) where a collection of K interconnected agents are coupled through an optimization problem of the following form:

$$\underset{w_1,w_2,\cdots,w_K}{\text{minimize}} \quad \sum_{k=1}^{K} J_k(w_k), \text{ s.t. } \sum_{k=1}^{K} B_k w_k - b_k = 0, \tag{6.1}$$

where $J_k(.)$: $\mathbb{R}^{Q_k} \rightarrow \mathbb{R}$ is a cost function associated with agent k and $w_k \in \mathbb{R}^{Q_k}$ is the variable for the same agent. The matrix $B_k \in \mathbb{R}^{S \times Q_k}$ and the vector $b_k \in \mathbb{R}^S$ are known locally by agent k only.

However, in many other applications, the constraint is sparse in the sense that some rows of B_k are zero. For example, in network flow optimization [138], multitask problems [148], distributed model predictive control [142], and optimal power flow [153, 154], the constraint has a special sparse structure. Specifically, each agent s is coupled with its neighboring nodes through an individual affine constraint of the form:

$$\sum_{k\in\mathcal{N}_s} B_{s,k} w_k = b_s, \quad \forall\ s = 1, \cdots, K \tag{6.2}$$

where $B_{s,k} \in \mathbb{R}^{S_s \times Q_k}$, $b_s \in \mathbb{R}^{S_s}$, and $\mathcal{N}_s$ denotes the neighborhood of agent s including agent s itself. Note that we can rewrite the constraints (6.2) into a single constraint of the form given in (6.1) by choosing B_k to be a block column matrix with blocks $\{B_{1,k}, \cdots, B_{K,k}\}$ and by setting $B_{s,k} = 0$ if $s \notin \mathcal{N}_k$. However, under *decentralized* settings, applying an algorithm that solves (6.1) directly and ignores the sparsity structure scales badly for large networks and its performance deteriorates as shown in this chapter. In some other applications (see Example 6.2 below), unlike (6.2), the number of constraints is arbitrary, and independent of the number of agents K. Moreover, each constraint may include any subset of agents and not only the agents in the neighborhood of some agent. Therefore, a general scalable algorithm that can exploit the sparsity in the constraint set is necessary for large scale networks.

Example 6.1. (Distributed Model Predictive Control) We examine a distributed finite-horizon control problem [142]. Thus, consider K subsystems. Given initial states $\{x_{s,0}\}$, each subsystem evolves over $t \geq 0$ according to the dynamics:

$$x_{s,t+1} = F_s x_{s,t} + G_{ss} u_{s,t} + \sum_{k\in\mathcal{N}_s'} \big(F_{sk} x_{k,t} + G_{sk} u_{k,t}\big) \tag{6.3}$$

where the matrices in (6.3) are of appropriate dimensions, $u_{s,t}$ is the input of subsystem s at time t, and $\mathcal{N}_s'$ denotes the neighborhood of agent s, excluding agent s. If we introduce the T block finite horizon vectors:

$$x_s \triangleq \text{col}\{x_{s,t}\}_{t=1}^{T}, \quad u_s \triangleq \text{col}\{u_{s,t}\}_{t=0}^{T-1} \tag{6.4}$$

then, by iterating (6.3), it can be verified that [155]:

$$x_s = G'_{ss} u_s + \sum_{k \in \mathcal{N}'_s} (F'_{sk} x_k + G'_{sk} u_k) + b_s \tag{6.5}$$

for matrices $\{G'_{sk}, F'_{sk}\}$ constructed from $\{F_s, F_{sk}, G_{sk}\}$ and some vector b_s constructed from F_s, F_{sk} and the initial condition $x_{s,0}$. If we let $w_s \triangleq \text{col}\{x_s, u_s\}$, and introduce:

$$B_{s,s} \triangleq \text{blkrow}\{I, -G'_{ss}\}, \quad B_{s,k} \triangleq -\text{blkrow}\{F'_{sk}, G'_{sk}\},$$

then we can formulate the following problem:

$$\begin{aligned} \underset{w_1, w_2, \cdots, w_K}{\text{minimize}} \quad & \sum_{k=1}^{K} \left(w_k^\mathsf{T} R_k w_k + r_k^\mathsf{T} w_k \right) \\ \text{subject to} \quad & \sum_{k \in \mathcal{N}_s} B_{s,k} w_k = b_s, \quad \forall\ s = 1, \cdots, K \end{aligned} \tag{6.6}$$

where $\mathcal{N}_s$ denotes the neighborhood of agent s, including agent s. □

Example 6.2. (General exchange in smart-grids) In this example, we will describe an example where unlike the previous examples, the number of constraints is arbitrary, and independent of the number of agents K. For simplicity, we describe the resource management (or economic dispatch) problem in smart grids [156] with minimum notation. To begin with, let P_{G_k} and P_{L_k} be the power generation supply and power load demand at node k. Moreover, let $P_k = \text{col}\{P_{G_k}, P_{L_k}\}$ be a 2×1 vector formed by stacking P_{G_k} and P_{L_k}. Then, the resource management problem over a power network consisting of K nodes is [157]:

$$\min_{\{P_k\}} \sum_{k=1}^{K} \big(J_k(P_k) + R_k(P_k) \big), \text{ s.t. } \sum_{k=1}^{K} \left(P_{G_k} - P_{L_k} \right) = 0, \tag{6.7}$$

where the non-differentiable term $R_k(P_k)$ is the indicator function of some capacity constraints such as positive powers and the maximum power generation. This problem fits into (6.1) and couples all nodes in a single constraint. The cost function typically used by power engineers is quadratic [153, 157]. In this formulation, it is assumed that each node is associated with one generator or load with P_k denoting the power generation or demand at that node. Assume now that each node k has multiple generators and/or loads. For example, each generator (or load) can be divided into sub-generators (or sub-loads). Moreover, assume that the power network is divided into K nodes that provide power to E sub-areas. Let $\mathcal{C}_e$ denote the collection of nodes providing power to sub-area e. Let P_{e,G_k} and P_{e,L_k} denote the power supply and power load at node k in area e – see Figure 6.1. In this figure, there are six nodes (agents) and three sub-areas (sub-networks). Each node associates different generators or loads to different sub-areas.

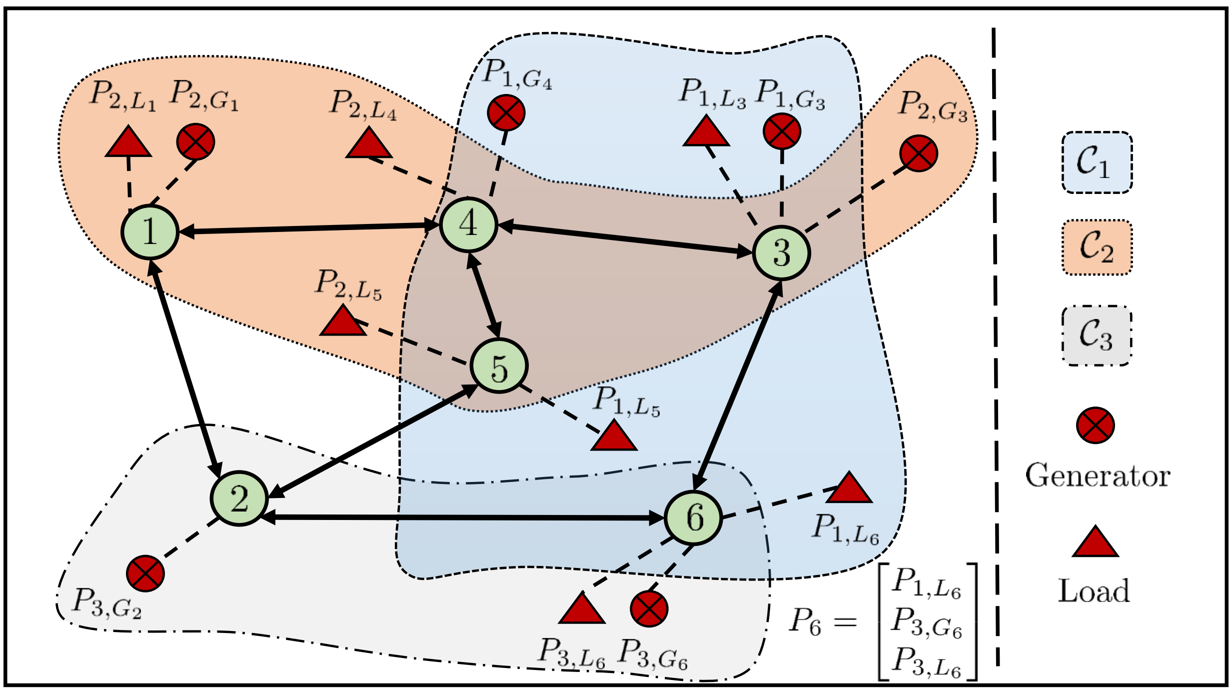


Figure 6.1: An illustration for Example 6.2. In this illustration, there are $E = 3$ areas and $K = 6$ agents.

If we let $\mathcal{C}_e$ denote the nodes that are involved in area e and P_k to be the augmented vector $P_k = \text{col}\{P_{e,G_k}, P_{e,L_k}\}_{e:k\in\mathcal{C}_e}$, which collects all local variables $\{P_{e,G_k}, P_{e,L_k}\}$ over all areas that agent k belongs to. Then, we formulate the following more general problem:

$$\begin{aligned} \underset{\{P_k\}}{\text{minimize}} \quad & \sum_{k=1}^{K} \big(J_k(P_k) + R_k(P_k)\big) \\ \text{subject to} \quad & \sum_{k \in \mathcal{C}_e} (P_{e,G_k} - P_{e,L_k}) = 0, \quad \forall\, e = 1, \cdots, E \end{aligned} \tag{6.8}$$

This formulation fits into the problem of dynamic energy exchange in smart grids applications [158] where each area is a connected sub-network. It can also be motivated as follows. Assume each sub-area represents some city. Then, problem (6.8) is useful when the transmission losses are costly in some parts of an area, which may require power generation from neighboring power networks. It is also useful when there are maintenance to some generators or lines causing high demands in some areas, which requires the need of extra generators from adjacent power networks. □

6.2 Problem Setup

Consider a network of K agents and assume that the agents are coupled through E affine equality constraint sets. For each constraint set e, we let $\mathcal{C}_e$ denote the *sub-network* of agents involved in this particular constraint(s). We then formulate the following optimization problem:

$$\begin{aligned} \underset{w_1, \cdots, w_K}{\text{minimize}} \quad & \sum_{k=1}^{K} \big(J_k(w_k) + R_k(w_k)\big) \\ \text{subject to} \quad & \sum_{k \in \mathcal{C}_e} (B_{e,k} w_k - b_{e,k}) = 0, \quad \forall\, e = 1, \cdots, E, \end{aligned} \tag{6.9}$$

where $B_{e,k} \in \mathbb{R}^{S_e \times Q_k}$ and $b_{e,k} \in \mathbb{R}^{S_e}$. The function $J_k(.) : \mathbb{R}^{Q_k} \to \mathbb{R}$ is a smooth convex function, while $R_k(.) : \mathbb{R}^{Q_k} \to \mathbb{R}$ is possibly a non-smooth convex function. For example, $R_k(.)$ could be an indicator function of some local constraints (e.g., $w_k \geq 0$). These functions are assumed to satisfy the conditions in Assumption 6.1 further ahead. It is also assumed that agent $k \in \mathcal{C}_e$ is only aware of $B_{e,k}$ and $b_{e,k}$. Note that for the special case $E = 1$ and

$\mathcal{C}_1 = \{1, \cdots, K\}$, problem (6.9) reduces to (6.1).

Assumption 6.1. (Cost function): *It is assumed that the aggregate function,* $\mathcal{J}(\scriptstyle\mathcal{W}\displaystyle) = \sum_{k=1}^{K} J_k(w_k)$ *where* $\mathcal{W} \triangleq \text{col}\{w_k\}_{k=1}^{K}$, *is a convex differentiable function with Lipschitz continuous gradient:*

$$\|\nabla_{\mathcal{W}}\mathcal{J}(\mathcal{W}) - \nabla_{\mathcal{W}}\mathcal{J}(z)\| \leq \delta\|\mathcal{W} - z\| \tag{6.10}$$

Moreover, $\mathcal{J}(\mathcal{W})$ *is also strongly convex, namely, it satisfies:*

$$(\mathcal{W} - z)^{\mathsf{T}}\nabla_{\mathcal{W}}\mathcal{J}(\mathcal{W}) \geq \mathcal{J}(\mathcal{W}) - \mathcal{J}(z) + \frac{\nu}{2}\|\mathcal{W} - z\|^2 \tag{6.11}$$

where $\{\delta, \nu\}$ *are strictly positive scalars with* $\delta > \nu$. *The regularization functions* $\{R_k(.)\}$ *are assumed to be closed convex functions, which implies that at least one subgradient exists at every point.* $\square$

These assumptions are widely employed in the literature and they are encountered in many practical applications such as distributed model predictive control [142], power systems [153], and data regression problems [53].

Assumption 6.2. (Sub-networks): *The network of* K *agents is undirected (i.e., agents can interact in both directions over the edges linking them) and each sub-network* $\mathcal{C}_e$ *is connected.* $\square$

This assumption means that there exists an undirected path between any two agents in each sub-network. This is automatically satisfied in various applications due to the physical nature of the problem. This is because coupling between agents often occurs for agents that are located close to each other. Applications where this assumption holds include, network flow optimization [138], optimal power flow [153, 154], and distributed model predictive control [142] problems. As explained in the introduction, in these problems, the constraints have the form given in equation (6.2). In this case, each constraint involves only the neighborhood of an agent, so that $\mathcal{C}_e = \mathcal{N}_s$ (for $s = e$) and neighborhoods are naturally connected. Now, more generally, even if some chosen sub-network happen to be disconnected, we can always construct a larger connected sub-network as long as the *entire* network is connected – an explanation of this construction procedure can be found in [90]. The problem of finding this construction is the well known *Steiner tree problem* [159] and, fortunately, distributed algorithms and heuristics exist to solve it [160, 161]. We now provide one motivational physical application that also satisfies the two previous assumptions.

6.2.1 Related Works

Many distributed algorithms have been developed for constraints of the form (6.2), but for special cases and/or under a different settings from what is considered in this work [153–155, 162, 163]. For example, the algorithms developed in [153–155, 162, 163] require the sharing of *primal* variables among neighboring agents and, moreover, the $s-$th constraint is of the form (6.2), which is limited to agents in the neighborhood of agent s. An augmented Lagrangian solution is pursued in [164], which further requires two hop communications. All these methods are not directly applicable for the case when the s-th constraint involves agents *beyond* the neighborhood of agent s. Extending these methods to this case would require multi-hop communication, which is costly. Moreover, the settings in these methods are different from this work. In these methods, the parameters of the s-th constraint $\{B_{s,k}, b_s\}_{k\in\mathcal{N}_s}$ are known solely by agent s. In this work, we consider a broader setting with arbitrary number of constraints, and each constraint may involve any subset of agents – see Section 6.2. Moreover, each agent s is only aware of the constraints matrices multiplying its own vector w_s.

The setting in this work is closer to the one considered in [53, 165–168]. However, these works focused on problems with a single coupling constraint of type (6.1), which ignores any sparsity structure. Problem (6.1) is solved in these references by using dual decomposition methods, which require each agent to maintain a dual variable associated with the constraint. Ignoring any sparsity structure means that each agent will be involved in the entire constraint. By doing so, each agent will maintain a long dual vector to reflect the *whole* constraint, and *all* agents in the network will have to reach consensus on a longer dual vector. The work [169] studied problem (6.1) for smooth functions with resource constraints (i.e., $\underline{w}_k \leq w_k \leq \overline{w}_k$) and focused on handling the useful case of dynamic and directed graphs. The matrix B_k in that work has a specific structure; moreover, the solution employed also shares the whole dual variable and neglects any sparsity structure. In other resource allocation problems [78,170–172] all agents are involved in a single constraint of the form (6.1) with $B_k = I$.

Different from the previously mentioned works, we consider a broader class of coupled affine constraints, where there exist multiple affine constraints and each constraint may involve any subset of agents. Our solution requires sharing dual variables only and does not directly share any sensitive primal information, e.g., it does not share the local variables $\{w_k\}$. Unlike the works [53, 165–169], which solve problem (6.1) and do not consider the sparsity structure in the constraint, this work exploits the constraint structure. In this way, each agent will only need to maintain the dual variables corresponding to its part of the constraints and not the *whole* constraint. Thus, only the agents involved in one particular part will need to agree on the associated dual variables. An algorithm that ignores the sparsity structure scales badly (in terms of communications and memory) as the number of constraints or agents increases. Moreover, it is theoretically shown in this work that the sparsity in the constraint set influences the performance of the algorithm in terms of convergence rate. Therefore, for large scale networks, it is important to design a scalable algorithm that exploits any sparsity in the constraint.

In [173], a multi-agent optimization problem is considered with stochastic quadratic costs and constraints similar to what is considered in this chapter albeit with substantially different settings. First, the work [173] considers quadratic costs only, does not handle non-differentiable terms, and their solution solves an approximate penalized problem instead of the original problem. Second, it is assumed that every agent knows all the matrices multiplying the vectors of all other agents involved in the same constraint. For example, for the constraint (6.2), agent s knows $\{B_{k',k}\}$ for all $k \in \mathcal{N}_s$ or $k' \in \mathcal{N}_s$. Lastly, the solution method requires every agent to maintain and receive delayed estimates of primal variables

w_k from all agents involved in the same constraint through a multi-hop relay protocol. This solution method suffers from high memory and communication burden; thus, it is impractical for large scale networks. In network utility maximization problems, a similar formulation appears, albeit with a different distributed framework; it is assumed that the agents (called sources) involved in a constraint are connected through a centralized unit (called link) that handles the constraint coupling these agents – see [79] and references therein.

6.2.2 Main Contributions

Given the above, we now state the main contributions of this work. A novel low computational distributed algorithm is developed that exploits the sparsity in the constraints. The developed algorithm handles non-differentiable terms and is shown to converge to the optimal solution for *constant* step-sizes. Furthermore, linear convergence is shown in the absence of non-differentiable terms and an explicit upper bound on the rate of convergence is given. This bound shows the importance of exploiting any constraint sparsity and why not doing so degrades the performance of the designed algorithm.

Table 6.1: A listing of repeatedly used symbols in this chapter.

Symbol	Description
$\mathcal{C}_e$	Sub-network of nodes involved in constraint e.
N_e	The cardinality of the set $\mathcal{C}_e$.
$\mathcal{E}_k$	The set of equality constraints indices involving agent k.
$\mathcal{W}$	The vector formed by stacking $\{w_k\}$ over all agents.
$\mathcal{J}(\mathcal{W})$	The sum of all smooth functions, $\mathcal{J}(\mathcal{W}) = \sum_{k=1}^{K} J_k(w_k)$.
v^e	Dual variable for equality constraint(s) e.
$\{v^e\}_{e\in\mathcal{E}_k}$	Collection of all dual variables v^e related to agent k.
v_k^e	Local copy of v^e at agent $k \in \mathcal{C}_e$.
$\mathcal{Y}^e$	Collection of v_k^e over all $k \in \mathcal{C}_e$.
$\mathcal{Y}$	Collection of $\mathcal{Y}^e$ over all e.

6.3 Algorithm Development

In this section, we will derive our algorithm and introduce some important symbols, which are necessary for algorithm description and later analysis. To do so, we start by introducing the Lagrangian function of (6.9):

$$\mathcal{L}\big(\scriptstyle\mathcal{W}\displaystyle, \{v^e\}\big) = \sum_{k=1}^{K} J_k'(w_k) + \sum_{e=1}^{E} (v^e)^\mathsf{T} \Bigg(\sum_{k \in \mathcal{C}_e} B_{e,k} w_k - b_{e,k} \Bigg) \tag{6.12}$$

where $J_k'(w_k) \stackrel{\Delta}{=} J_k(w_k) + R_k(w_k)$, and $v^e \in \mathbb{R}^{S_e}$ denotes the dual variable associated with the e-th constraint. To facilitate the development of the algorithm we rewrite (6.12) as a sum of local Lagrangian terms. To do so, we need to introduce the set $\mathcal{E}_k$, which denotes the set of equality constraints that agent k is involved in (e.g., if agent k is involved in equality constraints one and three, then $\mathcal{E}_k = \{1, 3\}$). From the definition of $\mathcal{E}_k$ and $\mathcal{C}_e$, we have

$$\mathcal{C}_e = \{k \mid e \in \mathcal{E}_k\}, \quad \mathcal{E}_k = \{e \mid k \in \mathcal{C}_e\} \tag{6.13}$$

Using this notation, the second term on the right hand side of (6.12) can be rewritten as a sum over all agents as follows: let $\overline{B}_{e,k} = B_{e,k}$ if $k \in \mathcal{C}_e$ (or $e \in \mathcal{E}_k$) and zero otherwise and, likewise, for $\bar{b}_{e,k}$. then it holds that:

$$\begin{aligned}
\sum_{e=1}^{E} \sum_{k \in \mathcal{C}_e} (v^e)^\mathsf{T} \left(B_{e,k} w_k - b_{e,k} \right) &= \sum_{e=1}^{E} \sum_{k=1}^{K} (v^e)^\mathsf{T} \left(\overline{B}_{e,k} w_k - \bar{b}_{e,k} \right) \\
&= \sum_{k=1}^{K} \sum_{e \in \mathcal{E}_k} (v^e)^\mathsf{T} \left(B_{e,k} w_k - b_{e,k} \right)
\end{aligned}$$

where in the last step we switched the order of summation and used the fact that $k \in \mathcal{C}_e$ if, and only, if $e \in \mathcal{E}_k$. Therefore, if we let $\{v^e\}_{e \in \mathcal{E}_k}$ denote the collection of dual variables related to agent k, then using the previous equation we can rewrite (6.12) as a sum of local terms as follows:

$$\mathcal{L}\big(\mathcal{W}, \{v^e\}\big) = \sum_{k=1}^{K} L_k\big(w_k, \{v^e\}_{e\in\mathcal{E}_k}\big) \tag{6.14}$$

where

$$L_k\big(w_k, \{v^e\}_{e\in\mathcal{E}_k}\big) \;\stackrel{\Delta}{=}\; J_k'(w_k) + \sum_{e\in\mathcal{E}_k} (v^e)^{\mathsf{T}}\big(B_{e,k}w_k - b_{e,k}\big) \tag{6.15}$$

is the local term for agent k. We are therefore interested in finding the minimizer of (6.9) through the equivalent solution of the saddle point problem [94]:

$$\min_{\mathcal{W}} \max_{\{v^e\}} \mathcal{L}(\mathcal{W}, \{v^e\}) \tag{6.16}$$

Assumption 6.3. (Strong duality) *A solution exists for problem* (6.16) *and strong duality holds.* □

Since our problem (6.9) is convex with affine constraints only, then Slater's condition is satisfied and strong duality holds [94, Section 5.2.3], which ensures that the solution of (6.16) coincides with the solution of (6.9). We denote an optimal solution pair of (6.16) by $\mathcal{W}^\star = \text{col}\{w_k^\star\}_{k=1}^K$ and $\{v^{e,\star}\}$. From assumption 6.1, $\mathcal{W}^\star$ is unique, but $\{v^{e,\star}\}$ are not necessarily unique. To derive our algorithm, which solves the saddle point problem (6.16), we will now relate the dual problem to the one considered in our previous work [90] and explain how the dual variables are partially shared across the agents, which is important for our derivation.

6.3.1 Dual Problem

Note that the Lagrangian (6.14) is separable in the variables $\{w_k\}$. Thus, the dual problem is [94](we are reversing the min and max operations by negating the function):

$$\underset{v^1,\cdots,v^E}{\text{minimize}} \quad -\sum_{k=1}^{K} f_k\big(\{v^e\}_{e\in\mathcal{E}_k}\big) \tag{6.17}$$

where[1]

$$f_k\big(\{v^e\}_{e\in\mathcal{E}_k}\big) \;\overset{\Delta}{=}\; \min_{w_k} L_k\big(w_k, \{v^e\}_{e\in\mathcal{E}_k}\big) \tag{6.18}$$

Figure 6.2 illustrates how the dual variables $\{v^e\}$ are shared across agents participating in the same constraint. For example, agent $k = 4$ in Figure 6.2 is part of two sub-networks, $\mathcal{C}_1$ and $\mathcal{C}_2$; it is therefore part of two equality constraints and will be influenced by their respective dual variables, denoted by v^1 and v^2. Similarly, for the other agents in the network. Problem (6.17) is of the form considered in [90]: it involves minimizing the aggregate sum of cost functions $f_k\big(\{v^e\}_{e\in\mathcal{E}_k}\big)$ where the arguments $\{v^e\}_{e\in\mathcal{E}_k}$ among different agents can share block entries as illustrated in Fig. 6.2 . The main difference here, however, is that the costs $f_k\big(\{v^e\}_{e\in\mathcal{E}_k}\big)$ do not admit a closed form expression in general and are instead defined by (6.18), i.e., in this work we are actually dealing with the more challenging saddle point problem and not with a minimization problem as was the case in [90]. Thus, more is needed to arrive at the solution of (6.16), as we explain later.

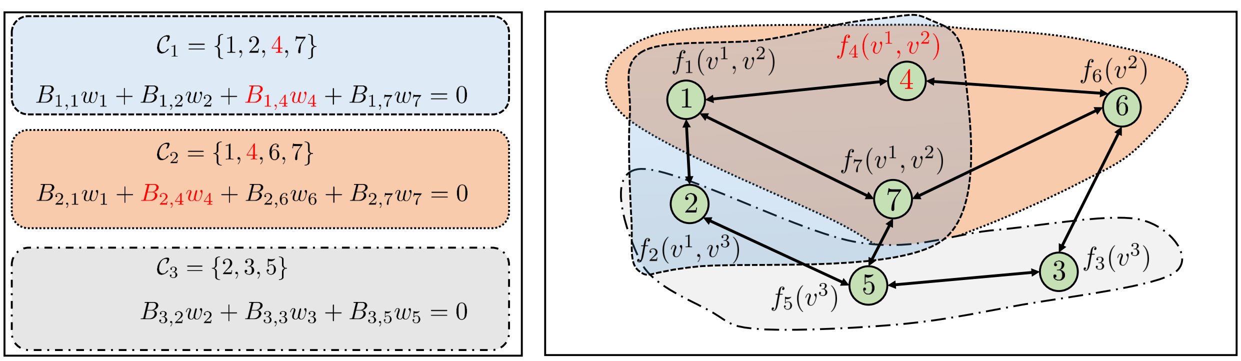


Figure 6.2: An example to illustrate the dual problem (6.17) for agent $k = 4$. In this example we have three sub-networks and agent 4 is involved in the equality constraints for sub-networks $\mathcal{C}_1$ and $\mathcal{C}_2$.

[1]Technically inf is used instead of min in (6.18), however, to avoid confusion we use min.

6.3.2 Combination Coefficients

To proceed from here and for the algorithm description, we introduce combination coefficients for the edges in $\mathcal{C}_e$ denoted by $\{a_{e,sk}\}_{s,k\in\mathcal{C}_e}$; $a_{e,sk}$ refers to the coefficient used to scale data moving from agent s to agent k in subnetwork $\mathcal{C}_e$ with $a_{e,sk} = 0$ if $s \notin \mathcal{N}_k \cap \mathcal{C}_e$. We collect these coefficients into the combination matrix

$$A_e \overset{\Delta}{=} [a_{e,sk}]_{s,k\in\mathcal{C}_e} \in \mathbb{R}^{N_e\times N_e} \tag{6.19}$$

where N_e denotes the number of agents involved in equality e. The matrix A_e is assumed to be symmetric and doubly-stochastic. We also require A_e to be primitive, meaning that there exists an integer j such that the entries of the matrix A_e^j are all positive. One way to meet these conditions is to choose weights satisfying

$$\begin{cases} \displaystyle\sum_{s\in\mathcal{C}_e} a_{e,sk} = 1, \qquad \sum_{k\in\mathcal{C}_e} a_{e,sk} = 1 & \text{(6.20a)} \\ a_{e,sk} > 0 \;\; \text{for} \;\; s \in \mathcal{N}_k \cap \mathcal{C}_e & \text{(6.20b)} \end{cases}$$

with $a_{e,sk} = 0$ if $s \notin \mathcal{N}_k \cap \mathcal{C}_e$. Under Assumption (6.2) many rules exists to choose such weights in a distributed way – see [8, Ch. 14]. We are now ready to derive our algorithm.

6.3.3 Dual Coupled Diffusion

Using the combination matrix A_e, it was shown in [90] that problem (6.17) can be solved by using the following coupled diffusion algorithm. Set $v^e_{k,-1} = \psi^e_{k,-1}$ to arbitrary values. For each k and $e \in \mathcal{E}_k$ repeat for $i \geq 1$:

$$\psi^e_{k,i} = v^e_{k,i-1} + \mu_v \nabla_{v^e} f_k\big(\{v^e_{k,i-1}\}_{e\in\mathcal{E}_k}\big) \tag{6.21a}$$

$$\phi^e_{k,i} = \psi^e_{k,i} + v^e_{k,i-1} - \psi^e_{k,i-1} \tag{6.21b}$$

$$v^e_{k,i} = \sum_{s\in\mathcal{N}_k\cap\mathcal{C}_e} \bar{a}_{e,sk}\phi^e_{s,i} \tag{6.21c}$$

where $v^e_{k,i}$ is the estimate for v^e at agent k, $\mu_v > 0$ is a step-size parameter, and $\{\psi^e_{k,i}, \phi^e_{k,i}\}$ are auxiliary vectors used to find $v^e_{k,i}$. The coefficients $\{\bar{a}_{e,sk}\}$ are the entries of the matrix $\bar{A}_e$ defined as follows:

$$\bar{A}_e = [\bar{a}_{e,sk}]_{s,k\in\mathcal{C}_e} \triangleq 0.5(I_{N_e} + A_e) \tag{6.22}$$

If the functions $\{f_k(.)\}$ are known and are differentiable, then each agent could run (6.21a)–(6.21c) to converge to its corresponding optimal dual variable, which in turn could be used to find the local minimizer $w^\star_k$ by solving $\min_{w_k} L_k\big(w_k, \{v^{e,\star}\}_{e\in\mathcal{E}_k}\big)$. However, this approach is not always possible because the local dual function $f_k\big(\{v^e\}_{e\in\mathcal{E}_k}\big)$ does not generally admit a closed form expression nor is guaranteed to be differentiable. Moreover, this method involves two time scales: one for finding the dual and the other for finding the primal. Therefore, to solve (6.16) we propose to employ a distributed version of the centralized dual-ascent construction [2] combined with a proximal gradient descent step. Specifically, recall first that the dual-ascent method updates the primal variable w_k at each iteration i as follows:

$$w_{k,i} = \arg\min_{w_k} L_k(w_k, \{v^e_{i-1}\}_{e\in\mathcal{E}_k}), \ \forall\ k \tag{6.23}$$

Note that this minimization step, which need to be solved at each iteration, can be costly in terms of computation unless a closed form solution exists, which is not the case in general. Therefore, we approximate (6.23) by a proximal gradient descent step to arrive at what we shall refer to as the *dual coupled* diffusion algorithm (6.24). At each time instant i, each agent k first performs a proximal gradient descent step (6.24a) for the primal variable. Then, for each dual-ascent step, the coupled diffusion (6.24b)–(6.24d) are applied where step (6.24b) is obtained by using $\nabla_{v^e} L_k(w_{k,i}, \{v^e_k\}_{e\in\mathcal{E}_k})$ to approximate the gradient at the minimum value in (6.18). Note that only step (6.24d) requires sharing dual variables with the neighbors that are involved in similar constraints.

To analyze algorithm (6.24) and show that it converges to an optimal solution of (6.16), we will rewrite it in a compact network form, which facilitates its analysis.

Algorithm (Dual Coupled Diffusion)

Setting: Let $v^e_{k,-1} = \psi^e_{k,-1}$ and $w_{k,-1}$ arbitrary.

For every agent k, repeat for $i \geq 0$:

$$w_{k,i} = \underset{\mu_w R_k}{\text{prox}}\Big(w_{k,i-1} - \mu_w \nabla_{w_k} J_k(w_{k,i-1}) - \mu_w \sum_{e\in\mathcal{E}_k} B^{\mathsf{T}}_{e,k} v^e_{k,i-1}\Big) \tag{6.24a}$$

For all $e \in \mathcal{E}_k$:

$$\psi^e_{k,i} = v^e_{k,i-1} + \mu_v \left(B_{e,k} w_{k,i} - b_{e,k} \right) \tag{6.24b}$$

$$\phi^e_{k,i} = \psi^e_{k,i} + v^e_{k,i-1} - \psi^e_{k,i-1} \tag{6.24c}$$

$$v^e_{k,i} = \sum_{s\in\mathcal{N}_k\cap\mathcal{C}_e} \bar{a}_{e,sk} \phi^e_{s,i} \tag{6.24d}$$

6.4 Network Recursion

We start by stacking the dual estimates within each cluster and then stacking over all the clusters. This will allow us to rewrite the dual steps (6.24b)–(6.24d) in a form that enables us to see the affect of each sub-network in our analysis. Thus, we introduce the sub-network vector that collects the dual estimates $v^e_{k,i}$ over the agents in $\mathcal{C}_e$:

$$\mathcal{Y}^e_i \;\overset{\Delta}{=}\; \text{col}\{v^e_{k,i}\}_{k\in\mathcal{C}_e} \in \mathbb{R}^{N_e S_e}, \tag{6.25}$$

and the global network vector that collects $\mathcal{Y}^e_i$ over all e:

$$\mathcal{Y}_i \;\overset{\Delta}{=}\; \text{col}\{\mathcal{Y}^e_i\}_{e=1}^{E} \tag{6.26}$$

We also repeat a similar construction for the quantities:

$$b_e \;\overset{\Delta}{=}\; \text{col}\{b_{e,k}\}_{k\in\mathcal{C}_e}, \qquad b \;\overset{\Delta}{=}\; \text{col}\,\{b_e\}_{e=1}^{E} \tag{6.27}$$

$$\bar{\mathcal{A}}_e \;\overset{\Delta}{=}\; \bar{A}_e \otimes I_{S_e}, \qquad \bar{\mathcal{A}} \;\overset{\Delta}{=}\; \text{blkdiag}\{\bar{\mathcal{A}}_e\}_{e=1}^{E} \tag{6.28}$$

where $\bar{A}_e = \frac{1}{2}(I_{N_e} + A_e)$. For the networked representation of the primal update (6.24a), we introduce the network quantities:

$$\mathcal{W}_i \triangleq \text{col}\{w_{k,i}\}_{k=1}^{K} \tag{6.29}$$

$$\mathcal{R}(\mathcal{W}) \triangleq \sum_{k=1}^{K} R_k(w_k) \tag{6.30}$$

$$\nabla \mathcal{J}(\mathcal{W}_i) \triangleq \text{col}\{\nabla J_k(w_{k,i})\}_{k=1}^{K} \tag{6.31}$$

We also need to represent the term $\sum_{e \in \mathcal{E}_k} B_{e,k}^{\mathsf{T}} v_{k,i-1}^e$ in terms of the network quantity $\mathcal{Y}_{i-1}$ defined in (6.26). To do that we first rewrite each term $B_{e,k}^{\mathsf{T}} v_{k,i-1}^e$ in terms of the sub-network vector $\mathcal{Y}_{i-1}^e$. This can be simply done by introducing the $1 \times N_e$ block row matrix $\mathcal{B}_{ek}^{\mathsf{T}}$ of similar block structure as $\mathcal{Y}_{i-1}^e$ such that $\mathcal{B}_{ek}^{\mathsf{T}} \mathcal{Y}_{i-1}^e = B_{e,k}^{\mathsf{T}} y_{k,i-1}^e$ if $k \in \mathcal{C}_e$ and zero otherwise – Figure 6.3 illustrates this construction. This construction can be represented by:

$$\mathcal{B}_{ek}^{\mathsf{T}} = \text{blkrow}\{B_{e,kk'}^{\mathsf{T}}\}_{k' \in \mathcal{C}_e} \tag{6.32a}$$

$$B_{e,kk'}^{\mathsf{T}} \triangleq \begin{cases} B_{e,k}^{\mathsf{T}}, & \text{if } \; k \in \mathcal{C}_e \;, k = k' \\ 0_{Q_k, S_e}, & \text{otherwise} \end{cases} \tag{6.32b}$$

Thus, we have $\sum_{e \in \mathcal{E}_k} B_{e,k}^{\mathsf{T}} v_{k,i-1}^e = \sum_{e=1}^{E} \mathcal{B}_{ek}^{\mathsf{T}} \mathcal{Y}_{i-1}^e$.

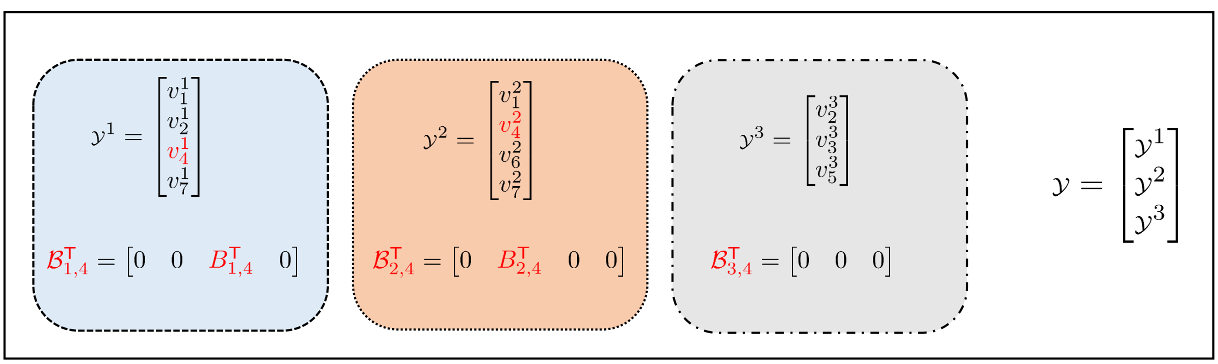


Figure 6.3: An illustration of constructions (6.25) and (6.26) for the network in Figure 6.2 as well as construction (6.32) for agent $k = 4$ in that network.

If we let

$$\mathcal{B} \triangleq \begin{bmatrix} \mathcal{B}_{11} & \cdots & \mathcal{B}_{1K} \\ \vdots & & \vdots \\ \mathcal{B}_{E1} & \cdots & \mathcal{B}_{EK} \end{bmatrix} \tag{6.33}$$

then algorithm (6.24) can be rewritten compactly as follows:

$$\mathcal{W}_i = \underset{\mu_w \mathcal{R}}{\text{prox}}\big(\mathcal{W}_{i-1} - \mu_w \nabla \mathcal{J}(\mathcal{W}_{i-1}) - \mu_w \mathcal{B}^{\mathsf{T}} \mathcal{Y}_{i-1}\big) \tag{6.34a}$$

$$\mathcal{Y}_i = \bar{\mathcal{A}}\Big(2\mathcal{Y}_{i-1} - \mathcal{Y}_{i-2} + \mu_v \mathcal{B}(\mathcal{W}_i - \mathcal{W}_{i-1})\Big) \tag{6.34b}$$

for $i \geq 1$ with initialization:

$$\mathcal{Y}_0 = \mathcal{Y}_{-1} + \mu_v(\mathcal{B}\mathcal{W}_0 - b) \tag{6.35}$$

Notice that step (6.34b) depends on the two previous estimates; thus it is tedious to analyze directly. Therefore, to facilitate our analysis we will rewrite it in an equivalent form. To do that, we let:

$$\mathcal{A} = \text{blkdiag}\{A_e \otimes I_{S_e}\}_{e=1}^{E} \tag{6.36}$$

and introduce the singular value (or eigenvalue for symmetric matrices) decomposition [?]:

$$0.5(I_N - \mathcal{A}) = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{U}_1^{\mathsf{T}} \\ \mathcal{U}_2^{\mathsf{T}} \end{bmatrix} = \mathcal{U}_1 \Sigma \mathcal{U}_1^{\mathsf{T}} \tag{6.37}$$

where $N = \sum_{e=1}^{E} N_e S_e$, $\mathcal{U}_1 \in \mathbb{R}^{N \times r}$, $\mathcal{U}_2 \in \mathbb{R}^{N \times (N-r)}$, and $\Sigma = \text{diag}\{\lambda_j\}_{j=1}^{r}$ with $\lambda_r \leq \cdots \leq \lambda_1$ denoting the non-zero eigenvalues of the matrix $0.5(I - \mathcal{A})$. Using an approach similar to the one used in [56], we can rewrite (6.34b) equivalently as follows — see Appendix 6.A:

$$\mathcal{X}_i = \mathcal{X}_{i-1} - \frac{1}{\mu_v}\mathcal{U}_1^{\mathsf{T}}\big(\mathcal{Y}_{i-1} + \mu_v(\mathcal{B}\mathcal{W}_i - b) + \mu_v\mathcal{U}_1\Sigma\mathcal{X}_{i-1}\big) \tag{6.38a}$$

$$\mathcal{Y}_i = \mathcal{Y}_{i-1} + \mu_v\big(\mathcal{B}\mathcal{W}_i - b\big) + \mu_v\mathcal{U}_1\Sigma\mathcal{X}_i \tag{6.38b}$$

for $i \geq 1$, where we introduced a new sequence $\mathcal{X}_i$ with $\mathcal{X}_0 = 0$. Intuitively, step (6.38b) can be regarded as a corrected gradient ascent step. Note that from the conditions in (6.20), it holds that the eigenvalues of each matrix A_e are in $(-1, 1]$ – see [8, Lemma F.4]. Thus, from the block structure of $\mathcal{A}$ in (6.36), the eigenvalues of the matrix $0.5(I - \mathcal{A})$ are in $[0, 1)$. Therefore, the non-zero eigenvalues are positive and satisfy:

$$0 < \lambda_r \leq \cdots \leq \lambda_1 < 1 \tag{6.39}$$

This property is important to show convergence.

6.5 Convergence Results

In this section, we give the Lemmas leading to the main convergence results.

Lemma 6.1. (Optimality condition) *If there exists a point $(\mathcal{W}^\star, \mathcal{Y}^\star, \mathcal{X}^\star)$ and a subgradient $g^\star \in \partial_{\mathcal{W}}\mathcal{R}(\mathcal{W}^\star)$ such that:*

$$\nabla\mathcal{J}(\mathcal{W}^\star) + g^\star + \mathcal{B}^{\mathsf{T}}\mathcal{Y}^\star = 0 \tag{6.40a}$$

$$\mathcal{U}_1^{\mathsf{T}}\mathcal{Y}^\star = 0 \tag{6.40b}$$

$$(\mathcal{B}\mathcal{W}^\star - b) + \mathcal{U}_1\Sigma\mathcal{X}^\star = 0 \tag{6.40c}$$

Then, it holds that $v_k^{e,\star} = v^{e,\star}\ \forall\ k \in \mathcal{C}_e$ where $(\mathcal{W}^\star, v^{1,\star}, \cdots, v^{E,\star})$ is a saddle point for the Lagrangian (6.12)*.*

Proof. Using the block structure of $\nabla\mathcal{J}(.)$ and $\mathcal{B}$ in (6.31) and (6.32)–(6.33), we can expand

(6.40a) into its components to get:

$$\nabla_{w_k} J_k(w_k^\star) + g_k^\star + \sum_{e \in \mathcal{E}_k} B_{e,k}^\mathsf{T} v_k^{e,\star} = 0, \quad \forall\, k \tag{6.41}$$

where $g_k^\star \in \partial_{w_k} R_k(w_k^\star)$. From the fact $\mathcal{U}_1^\mathsf{T}\mathcal{U}_1 = I$ and $\Sigma > 0$, condition (6.40b) is equivalent to:

$$\mathcal{U}_1^\mathsf{T} \mathcal{Y}^\star = 0 \iff \mathcal{U}_1 \Sigma \mathcal{U}_1^\mathsf{T} \mathcal{Y}^\star = 0 \iff \frac{1}{2}(I - \mathcal{A})\mathcal{Y}^\star = 0 \tag{6.42}$$

Therefore, from (1.5), and the block structure of $\mathcal{A}$ in (6.36), condition (6.40b) gives:

$$v_k^{e,\star} = v_s^{e,\star} = v^{e,\star}, \quad \forall\, k, s \in \mathcal{C}_e \tag{6.43}$$

for some $v^{e,\star}$. Hence, condition (6.41) satisfies the first optimality condition for problem (6.9) – see [94]. Now, let $\mathcal{Z} = \text{blkdiag}\{\mathbb{1}_{N_e} \otimes I_{S_e}\}_{e=1}^{E}$. Multiplying equation (6.40c) on the left by $\mathcal{Z}^\mathsf{T}$ gives:

$$0 = \mathcal{Z}^\mathsf{T}(\mathcal{B}\mathcal{W}^\star - b) + \underbrace{\mathcal{Z}^\mathsf{T}\mathcal{U}_1}_{=0} \Sigma \mathcal{X}^\star \overset{(a)}{=} \mathcal{Z}^\mathsf{T}(\mathcal{B}\mathcal{W}^\star - b) \tag{6.44}$$

where step (a) holds because because from (1.5), $\mathcal{Z}$ is in the nullspace of $I - \mathcal{A}$ and thus also in the nullspace of $\mathcal{U}_1^\mathsf{T}$ [88, Equation (51)]. Using the block structure of $\mathcal{B}$ and b in (6.32)–(6.33) and (6.27), we can also expand (6.44) into its components to get:

$$(\mathbb{1}_{N_e}^\mathsf{T} \otimes I_{S_e})\left(\sum_{k=1}^{K} (\mathcal{B}_{ek} w_k^\star) - b_e\right) = \sum_{k \in \mathcal{C}_e} (B_{e,k} w_k^\star - b_{e,k}) = 0 \tag{6.45}$$

for all e since

$$\mathcal{B}_{ek}^\mathsf{T}(\mathbb{1}_{N_e} \otimes I_{S_e}) = \sum_{k' \in \mathcal{C}_e} B_{e,kk'}^\mathsf{T} = \begin{cases} B_{e,k}^\mathsf{T}, & \text{if } k \in \mathcal{C}_e \\ 0, & \text{otherwise} \end{cases} \tag{6.46}$$

Equation (6.45) is the second optimality condition for problem (6.9) and, thus, $(\mathcal{W}^\star, v^{1,\star}, \cdots, v^{E,\star})$ is an optimal point for (6.16) – [94]. $\square$

Remark 6.1 (EXISTENCE AND UNIQUENESS)**.** Note that there exists a point $(\mathcal{W}^\star, \mathcal{Y}^\star, \mathcal{X}^\star)$ that satisfies the optimality conditions (6.40). Specifically, if $\mathcal{W}^\star = \text{col}\{w_k^\star\}$ and $\mathcal{Y}^\star = \text{col}\{\mathbb{1} \otimes v^{e,\star}\}_{e=1}^E$, where $(\mathcal{W}^\star, v^{1,\star}, \cdots, v^{E,\star})$ is an optimal solution of the saddle point problem (6.16), then, it can be easily verified that conditions (6.40a)–(6.40b) are satisfied. Now, by following an argument similar to the one used in [93, Lemma 3], it can be shown that there exists an $\mathcal{X}^\star$ such that (6.40c) holds; moreover, there exists a unique $\mathcal{X}^\star$ in the range space of $\mathcal{U}_1^\mathsf{T}$. Now, we know from strong convexity that $\mathcal{W}^\star$ is unique. Thus, from (6.40a), the dual point $\mathcal{Y}^\star$ is unique if the matrix $\mathcal{B}$ has full row rank. Under this condition and in the absence of non-smooth terms, we will show that our algorithm converges linearly to this unique point – see Theorem 6.2 . $\square$

We will now show that the equivalent network recursions (6.34a) and (6.38a)–(6.38b) of the proposed algorithm converge to a point that satisfies the optimality conditions given in Lemma 6.1. To give the convergence results, we introduce the error vectors:

$$\widetilde{\mathcal{W}}_i \overset{\Delta}{=} \mathcal{W}^\star - \mathcal{W}_i, \quad \widetilde{\mathcal{X}}_i \overset{\Delta}{=} \mathcal{X}^\star - \mathcal{X}_i \quad \widetilde{\mathcal{Y}}_i \overset{\Delta}{=} \mathcal{Y}^\star - \mathcal{Y}_i \tag{6.47}$$

and the diagonal matrix:

$$\mathcal{D} \overset{\Delta}{=} \mu_v(\Sigma - \Sigma^2) > 0 \tag{6.48}$$

where Σ was introduced in (6.37). Note that $\mathcal{D}$ is positive definite because of (6.39).

Lemma 6.2. (Primal-dual bound)*: Suppose Assumptions 6.1-6.3 hold, then:*

$$\begin{aligned}\|\widetilde{\mathcal{W}}_i\|^2 - \|\widetilde{\mathcal{W}}_{i-1}\|^2 \leq &-\big(1 - \mu_w(2\delta - \nu)\big)\|\mathcal{W}_i - \mathcal{W}_{i-1}\|^2 - \mu_w\nu\left(\|\widetilde{\mathcal{W}}_{i-1}\|^2 + \|\widetilde{\mathcal{W}}_i\|^2\right)\\ &- 2\mu_w(\mathcal{Y}_{i-1} - \mathcal{Y}^\star)^\mathsf{T}\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star)\end{aligned} \tag{6.49}$$

and

$$\|\widetilde{\mathcal{Y}}_i\|^2_{\mu_v^{-1}} + \|\widetilde{\mathcal{X}}_i\|^2_{\mathcal{D}} - \|\widetilde{\mathcal{Y}}_{i-1}\|^2_{\mu_v^{-1}} - \|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mathcal{D}} = -\|\mathcal{X}_i - \mathcal{X}_{i-1}\|^2_{\mathcal{D}} - \|\Sigma\widetilde{\mathcal{X}}_i\|^2_{\mu_v} + \|\mathcal{B}\widetilde{\mathcal{W}}_i\|^2_{\mu_v}$$
$$+ 2(\mathcal{Y}_{i-1} - \mathcal{Y}^\star)^\mathsf{T}\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star) \tag{6.50}$$

where $(\mathcal{W}^\star, \mathcal{Y}^\star, \mathcal{X}^\star)$ *satisfy the optimality conditions given in Lemma 6.1.*

Proof. See Appendices 6.B and 6.C. □

The previous Lemma is used to establish the following theorem.

Theorem 6.1. (Convergence)*: Suppose Assumptions 6.1–6.3 hold, then for positive constant step-sizes satisfying:*

$$\begin{cases} \mu_w < \dfrac{1}{(2\delta - \nu)} & (6.51a) \\ \mu_v < \dfrac{\nu}{\lambda_{\max}(\mathcal{B}^\mathsf{T}\mathcal{B})} & (6.51b) \end{cases}$$

recursions (6.34a) *and* (6.38a)–(6.38b) *converge and it holds that* $\mathcal{W}_i$ *converges to the optimal solution of* (6.9).

Proof. See Appendix 6.D □

At this point we showed that the dual coupled diffusion strategy, which handles non-smooth terms, converges to the optimal point. However, it is still unclear how the sparsity of the constraints affects the convergence behavior. Apart from saving communication and memory, the next result reveals the advantage of exploiting the constraint structure.

Theorem 6.2. (Linear convergence)*: Suppose Assumptions 6.1–6.3 hold, and, furthermore, assume that each* $R_k(w_k) = 0$ *and each matrix* $\text{blkcol}\{B_{e,k}\}_{e\in\mathcal{E}_k}$ *has full row rank. If the step sizes satisfy* (6.51)*, then it holds that:*

$$\|\widetilde{\mathcal{W}}_i\|^2_{C_w} + \|\widetilde{\mathcal{Y}}_i\|^2_{\frac{\mu_w}{\mu_v}} + \|\widetilde{\mathcal{X}}_i\|^2_{\mu_w\mu_v\Sigma} \le \gamma^i C_0 \tag{6.52}$$

where $C_w = I - \mu_w\mu_v\mathcal{B}^\mathsf{T}\mathcal{B} > 0$, $C_0 \;\overset{\Delta}{=}\; \|\widetilde{\scriptstyle\mathcal{W}}_0\|^2 + \|\widetilde{\scriptstyle\mathcal{Y}}_0\|^2_{\frac{\mu_w}{\mu_v}} + \|\widetilde{\scriptstyle\mathcal{X}}_0\|^2_{\mu_w\mu_v\Sigma}$ *and*

$$\gamma \;\overset{\Delta}{=}\; \max\left\{1 - \mu_w\nu(1-\delta\mu_w), 1 - \mu_w\mu_v\lambda_{\min}(\mathcal{B}\mathcal{B}^\mathsf{T}), 1 - \lambda_r\right\} < 1$$

with λ_r *denoting the smallest non-zero eigenvalue of* $0.5(I - \mathcal{A})$.

Proof. See Appendix 6.E. □

The above result shows why solving (6.9) directly is important for at least two reasons. First, by using model (6.9), we are able to prove linear convergence under the assumption that each $\text{blkcol}\{B_{e,k}\}_{e\in\mathcal{E}_k}$ has full row rank. If instead, we were to rewrite problem (6.9) into the form (6.1) by embedding zeros into the matrices B_k, then *our analysis* would require B_k to be full row rank for linear convergence. This will not be satisfied if some agent is not involved in some constraint since in that case B_k will have zero rows and, thus, B_k is row rank deficient even if $\text{blkcol}\{B_{e,k}\}_{e\in\mathcal{E}_k}$ has full row rank.

The second more important reason is that the convergence rate depends on the connectivity of the sub-networks $\mathcal{C}_e$ and not on the connectivity of the entire network, as we illustrate now. Note from the block structure of (6.36) that the smallest non-negative eigenvalue of $0.5(I - \mathcal{A})$ has the form $\lambda_r = \min_e \sigma_e$ where σ_e denotes the smallest non-zero eigenvalue of the matrix $0.5(I - A_e)$. Since

$$I - 0.5(I - A_e) = 0.5(I + A_e) = \bar{A}_e \tag{6.53}$$

it holds that $1 - \sigma_e = \bar{\lambda}_e$, where $\bar{\lambda}_e$ denotes the second largest eigenvalue of $\bar{A}_e$ (the largest eigenvalue is equal to one). Therefore,

$$1 - \lambda_r = 1 - \min_e \sigma_e = \max_e(1 - \sigma_e) = \max_e \bar{\lambda}_e \tag{6.54}$$

Thus, assuming $1 - \lambda_r$ is dominating the convergence rate, then the smaller $\max_e \bar{\lambda}_e$ is, the faster the algorithm is. We see that this depends on the second largest eigenvalue of the matrices $\{\bar{A}_e\}$, which depends on the sub-networks connectivity and not the whole network. This observation reveals the importance of the algorithm for sparse networks and under sparsely coupled constraints. Since in that case the small sub-networks are much well connected than the whole network. This observation will be illustrated in the simulation section next.

Remark 6.2 (Condition Number). By using the the upper bound (6.51), we conclude from Theorem 6.2 that the number of iterations needed to reach ϵ accuracy is on the order of

$$O\big(\max\{\kappa_J\kappa_B, 1/\lambda_r\}\big)\log\frac{1}{\epsilon}$$

where $\kappa_J = \delta/\nu$ and $\kappa_B = \lambda_{\max}(\mathcal{B}\mathcal{B}^{\mathsf{T}})/\lambda_{\min}(\mathcal{B}\mathcal{B}^{\mathsf{T}})$ are the condition numbers of the cost $\mathcal{J}(.)$ and the matrix $\mathcal{B}\mathcal{B}^{\mathsf{T}}$, respectively. $\square$

6.6 Numerical Simulation

In this section, we test the performance of the proposed algorithm with two numerical experiments.

- (Distributed Linear Regression) The first set-up considers a linear regression problem with costs:

$$J_k(w_k) = \frac{1}{2T_k}\sum_{t=1}^{T_k} \|u_{k,t}^{\mathsf{T}} w_k - p_k(t)\|^2 + \eta_1\|w_k\|_1$$

  where $u_{k,t} \in \mathbb{R}^{Q_k}$ is the regressor vector for data sample t and $p_k(t) \in \mathbb{R}$.

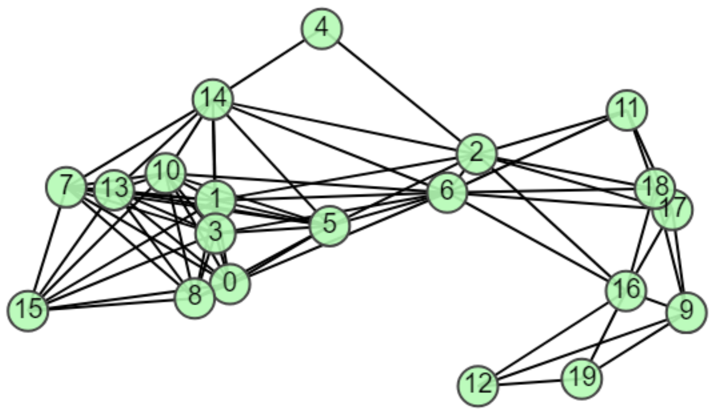
- (Distributed Logistic Regression) The second set-up considers a logistic regression problem with costs:

$$J_k(w_k) = \frac{1}{T_k}\sum_{t=1}^{T_k} \ln\big(1 + \exp(-x_k(t)h_{k,t}^{\mathsf{T}} w_k)\big) + \eta_2\|w_k\|_1$$

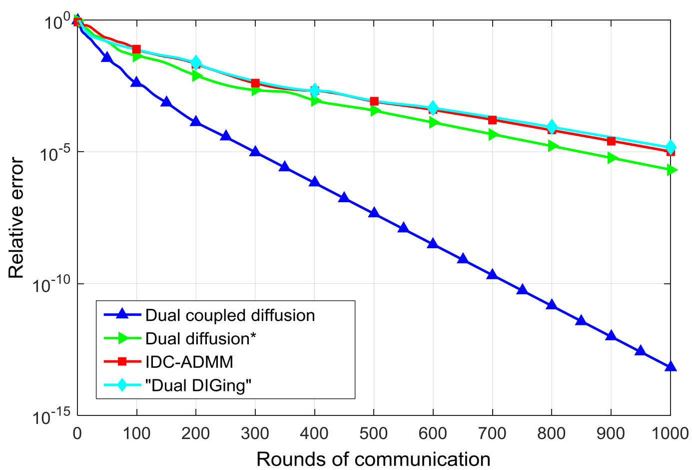
with additional 2-norm regularizers $R'_k(w_k) = 0.5\eta_3\|w_k\|^2$. The vector $h_{k,t} \in \mathbb{R}^{Q_k}$ is the regressor vector for data sample t, and $x_k(t)$ is the label for that data sample, which is either $+1$ or -1. In both costs, T_k denotes the amount of data for agent k.

In both experiments, the network used is shown in Fig. 6.4a with $K = 20$ agents. The positions (x-axis and y-axis) of the agents are randomly generated in $([0,1],[0,1])$, and two agents are connected if the distance between them is less than or equal $d = 0.3$. As for the constraints, we assume $E = K = 20$, and each constraint e (or k) (where $e \in \{1, \cdots, 20\}$) is associated with a subnetwork involving agent e (or k) and all its neighbors as described in equation (6.2). Each element in $B_{e,k}$ is generated according to the standard Gaussian distribution $\mathcal{N}(0,1)$. Each $b_{e,k}$ is also randomly generated and we guarantee that there exists a feasible solution to (6.9). All the combination matrices are generated according to the Metropolis rule.

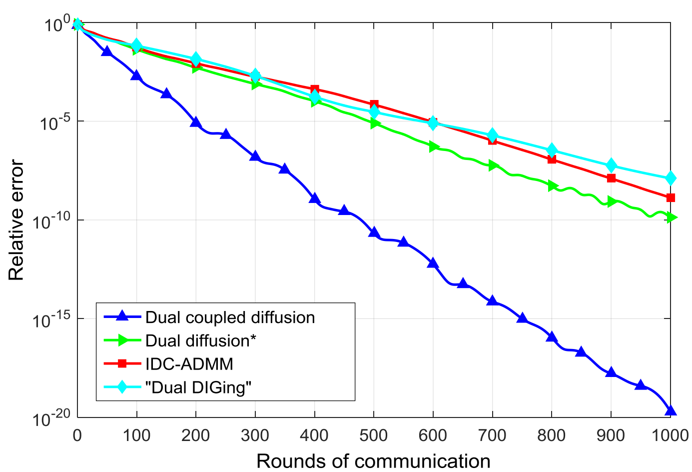
In the first simulation, we set $T_k = 1000$ for all k and each regressor $u_{k,t}$ is generated according to the Gaussian distribution $\mathcal{N}(0,1)$. To generate the associated $p_k(t)$, we first generate a vector $w_{k,0} \in \mathbb{R}^{Q_k}$ randomly from $\mathcal{N}(0,1)$. We let 20% of the entries of $w_{0,k}$ to be 0. With such sparse $w_{k,0}$, we generate $p_k(t)$ as $p_k(t) = u_{k,t}^{\mathsf{T}} w_{k,0} + n_k$ where $n_k \sim \mathcal{N}(0, 0.1)$ is some Gaussian noise. In this experiment, we set $Q_k = 10$ for $k = 1, \cdots, K$. We also set $\eta_1 = 0.3$ and $B_{e,k} \in \mathbb{R}^{3\times 10}$ to be an under-determined coefficient matrix. In the second set-up, each $T_k = 1000$. Among all local data samples, half of them are generated by the Gaussian distribution $\mathcal{N}(1,1)$ and their corresponding labels $\{x_k(t)\}$ are $+1$'s. The other half are generated by $\mathcal{N}(-1,1)$ and their corresponding labels $\{x_k(t)\}$ are -1's. We set $Q_k = 10$ for $k = 1, \cdots, K$ and $\eta_2 = \eta_3 = 0.1$. We let $B_{e,k} \in \mathbb{R}^{3\times 10}$ to be an under-determined coefficient matrix.



(a) The network topology used in simulations.



(b) Least squares results.



(c) Logistic regression results.

Figure 6.4: Simulation results. *Dual diffusion refers to (6.24) applied on the same problem reformulated into (6.1), which ignores the sparsity structure. Similarly, both IDC-ADMM [53] and "dual DIGing" [174] are designed for problem (6.1) and ignore the sparsity structure.

To illustrate the effect of the constraint structure, we consider two approaches to solve problem (6.9). The first approach is to use the dual coupled diffusion (6.24) while considering the structure of the problem (6.9), i.e., run (6.24) with $E = K, \mathcal{C}_e = \mathcal{N}_e$. The second approach is to ignore the special structure of the problem and reformulate it into the form of problem (6.1) and also run the dual coupled diffusion (6.24) with $E = 1, \mathcal{C}_1 = \{1, \cdots, K\}$, which we call dual diffusion. To compare with other related methods that only share dual variables, we simulate the inexact distributed consensus ADMM (IDC-ADMM) from [53] and a modified proximal version of the one in [174] in which the dual iterates are updated similar to the DIGing algorithm in [62], which we call "Dual DIGing". Both of these algorithm are designed for problem (6.1) and ignores any structure. The step-sizes are chosen manually to get the best possible performance for each algorithm. In the first linear regression setup, the parameters used are ($\mu_w = 0.28,\ \mu_v = 0.28$) for the dual coupled diffusion, ($\mu_w = 0.28,\ \mu_v = 0.28$) for the dual diffusion, ($c = 0.25,\ \mu_w = 0.05$) for the IDC-ADMM [53], and the step-sizes are set to 0.45 for the dual DIGing method. In the second logistic regression set-up, they are set to ($\mu_w = 0.2,\ \mu_v = 0.2$) for the dual coupled diffusion, ($\mu_w = 0.2,\ \mu_v = 0.2$) for the dual diffusion, ($c = 0.45,\ \mu_w = 0.2$) for the IDC-ADMM [53], and the step-sizes are set to 0.18 for the dual DIGing method. Figure 6.4 shows the relative error $\frac{1}{K}\sum_{k=1}^{K} \|w_{k,i} - w_k^\star\|^2/\|w_k^\star\|^2$ for each of the previous algorithms for both set-ups. Note that the dual DIGing algorithm requires two rounds of communication per iteration. Therefore, in the x-axis we use rounds of communication for a fair comparison. It is observed that dual diffusion, the IDC-ADMM, and the dual DIGing algorithms have a close performance (all ignores any structure), while the dual coupled diffusion clearly outperforms them. This means that, apart from requiring less amount of data to be exchanged per round of communication, our algorithm is also able to reach an ϵ accuracy (where ϵ is arbitrarily small) with much less time compared to these other algorithms. As explained before, this superiority is due to the sub-networks being better connected compared to the whole network and the dual coupled diffusion takes advantage of that. In this simulation, we have $1 - \lambda_r = 0.911$ for the dual coupled diffusion and $1 - \lambda = 0.973$ for the dual diffusion (we dropped the sub-index since we have one network combination matrix in this case), which backs up our theoretical findings. To further illustrate the effect of the sub-networks connectivity on the convergence rate, we simulate the dual coupled diffusion (exploits sparsity) and dual diffusion (which does not exploit the sparsity) with the same logistic regression set-up from before but for the three different networks shown in top half of Fig. 6.5. The step sizes used in this simulation are adjusted to get the best possible results, which are shown on the bottom of Figure 6.5. Note that the network on the left has less connections compared to the network on the right, and thus, the sub-networks on the left are more sparse than the one on the right. Note further that for the constraints settings used (6.2), the more connections the network has, the closer the sub-networks are to the entire network. It is seen that dual coupled diffusion performs significantly better under sparser networks since in that case the sub-networks are much better connected than the whole network. On the other hand, when we add more connections, the sub-networks connectivity becomes closer to the network connectivity and, thus, the performance of the two algorithms become closer and closer. The performance will become identical when all agents are involved in all the constraint.

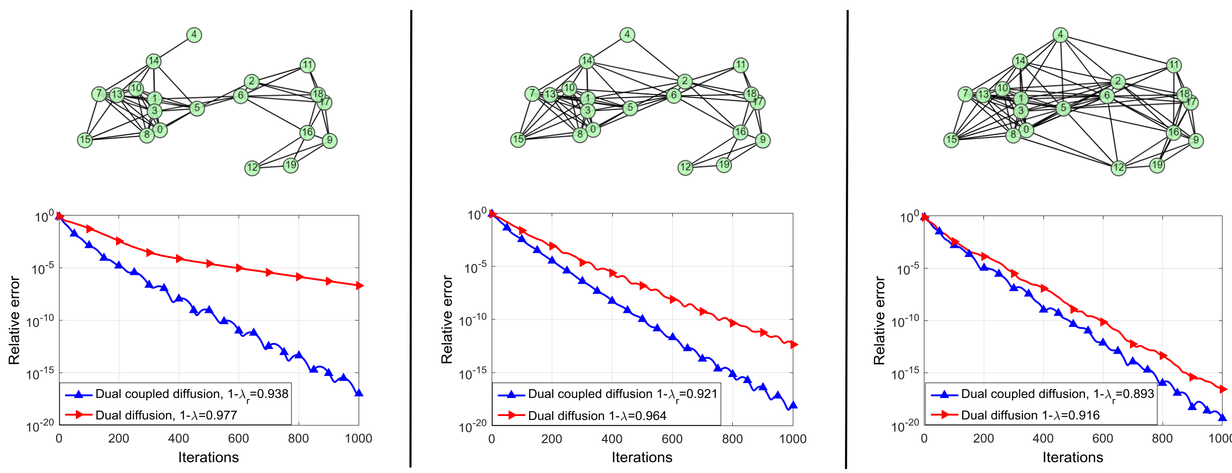


Figure 6.5: A comparison of algorithm (6.24) for different network connectivity and under two implementations: dual coupled diffusion exploits structure while dual diffusion ignores the structure.

Appendices

6.A Equivalent Representation

In this appendix, we show that (6.38a)–(6.38b) is equivalent to (6.34b). Multiplying equation (6.38a) by $\mathcal{U}_1\Sigma$ and then collecting the term $\mathcal{U}_1\Sigma\mathcal{X}_{i-1}$ we get:

$$\mathcal{U}_1\Sigma\mathcal{X}_i = (I - \mathcal{U}_1\Sigma\mathcal{U}_1^{\mathsf{T}})\mathcal{U}_1\Sigma\mathcal{X}_{i-1} - \mu_v^{-1}\left(\mathcal{U}_1\Sigma\mathcal{U}_1^{\mathsf{T}}\right)\Big(\mathcal{Y}_{i-1} + \mu_v\big(\mathcal{B}\mathcal{W}_i - b\big)\Big) \tag{6.55}$$

Let $\bar{\mathcal{X}}_i \triangleq \mathcal{U}_1\Sigma\mathcal{X}_i$. Using (6.37) and collecting the term $\mathcal{X}_{i-1}$ on the right hand side of the last equation, we get:

$$\bar{\mathcal{X}}_i = \bar{\mathcal{A}}\bar{\mathcal{X}}_{i-1} - \mu_v^{-1}\frac{1}{2}\left(I - \mathcal{A}\right)\big(\mathcal{Y}_{i-1} + \mu_v\big(\mathcal{B}\mathcal{W}_i - b\big)\big) \tag{6.56}$$

Multiplying (6.38b) by $\bar{\mathcal{A}}$ on the left and using the definition $\bar{\mathcal{X}}_i \triangleq \mathcal{U}_1\Sigma\mathcal{X}_i$ we have:

$$\bar{\mathcal{A}}\mathcal{Y}_{i-1} = \bar{\mathcal{A}}\mathcal{Y}_{i-2} + \mu_v\bar{\mathcal{A}}\big(\mathcal{B}\mathcal{W}_{i-1} - b\big) + \mu_v\bar{\mathcal{A}}\bar{\mathcal{X}}_{i-1} \tag{6.57}$$

Now, subtracting (6.57) from (6.38b) we get:

$$\begin{aligned}\mathcal{Y}_i - \bar{\mathcal{A}}\mathcal{Y}_{i-1} = {} & \mathcal{Y}_{i-1} - \bar{\mathcal{A}}\mathcal{Y}_{i-2} + \mu_v\big(\mathcal{B}\mathcal{W}_i - b\big) - \mu_v\bar{\mathcal{A}}\big(\mathcal{B}\mathcal{W}_{i-1} - b\big)\\ & + \mu_v(\bar{\mathcal{X}}_i - \bar{\mathcal{A}}\bar{\mathcal{X}}_{i-1})\end{aligned} \tag{6.58}$$

Using (6.56) we can remove the term $\mu_v(\bar{\mathcal{X}}_i - \bar{\mathcal{A}}\bar{\mathcal{X}}_{i-1})$ from the previous expression to get:

$$\begin{aligned}\mathcal{Y}_i - \bar{\mathcal{A}}\mathcal{Y}_{i-1} &= \mathcal{Y}_{i-1} - \bar{\mathcal{A}}\mathcal{Y}_{i-2} + \mu_v\big(\mathcal{B}\mathcal{W}_i - b\big) - \mu_v\bar{\mathcal{A}}\big(\mathcal{B}\mathcal{W}_{i-1} - b\big)\\ &\quad - \frac{1}{2}\left(I - \mathcal{A}\right)\big(\mathcal{Y}_{i-1} + \mu_v\big(\mathcal{B}\mathcal{W}_i - b\big)\big)\\ &= \bar{\mathcal{A}}\mathcal{Y}_{i-1} - \bar{\mathcal{A}}\mathcal{Y}_{i-2} + \mu_v\bar{\mathcal{A}}\big(\mathcal{B}\mathcal{W}_i - b\big) - \mu_v\bar{\mathcal{A}}\big(\mathcal{B}\mathcal{W}_{i-1} - b\big)\end{aligned} \tag{6.59}$$

Rearranging the last expression gives (6.34b).

6.B Primal Error Bound

In this appendix, we derive inequality (6.49). From the optimality condition of (6.34a), we have:

$$\mathcal{w}_i = \mathcal{w}_{i-1} - \mu_w \nabla \mathcal{J}(\mathcal{w}_{i-1}) - \mu_w \mathcal{B}^\mathsf{T} \mathcal{y}_{i-1} - \mu_w g_i \tag{6.60}$$

for some $g_i \in \partial_w \mathcal{R}(\mathcal{w}_i)$. Rearranging the last equation and using the optimality condition (6.40a) we get:

$$\mathcal{w}_{i-1} - \mathcal{w}_i = \mu_w \big(\nabla \mathcal{J}(\mathcal{w}_{i-1}) - \nabla \mathcal{J}(\mathcal{w}^\star)\big) + \mu_w (g_i - g^\star) + \mu_w \mathcal{B}^\mathsf{T} (\mathcal{y}_{i-1} - \mathcal{y}^\star) \tag{6.61}$$

Multiplying $(\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T}$ to both sides of the previous equation, we get:

$$\begin{aligned}(\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T}(\mathcal{w}_{i-1} - \mathcal{w}_i) = \mu_w (\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T} \big(\nabla \mathcal{J}(\mathcal{w}_{i-1}) - \nabla \mathcal{J}(\mathcal{w}^\star)\big) \\ + \mu_w (\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T} (g_i - g^\star) \\ + \mu_w (\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T} \mathcal{B}^\mathsf{T} (\mathcal{y}_{i-1} - \mathcal{y}^\star)\end{aligned} \tag{6.62}$$

From the conditions on $R_k(w_k)$ in Assumption 6.1, there exists at least one subgradient at every point. And from the subgradient property (1.9) we have $g_x^\mathsf{T}(y - x) \le f(y) - f(x)$ and $g_y^\mathsf{T}(x - y) \le f(x) - f(y)$. Summing the two inequalities with $y = \mathcal{w}^\star$ and $x = \mathcal{w}_i$, we get:

$$(\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T}(g_i - g^\star) \le 0 \tag{6.63}$$

Using this bound in (6.62) we get:

$$\begin{aligned}(\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T}(\mathcal{w}_{i-1} - \mathcal{w}_i) \le \mu_w (\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T} \big(\nabla \mathcal{J}(\mathcal{w}_{i-1}) - \nabla \mathcal{J}(\mathcal{w}^\star)\big) \\ + \mu_w (\mathcal{w}^\star - \mathcal{w}_i)^\mathsf{T} \mathcal{B}^\mathsf{T} (\mathcal{y}_{i-1} - \mathcal{y}^\star)\end{aligned} \tag{6.64}$$

Note that:

$$
\begin{aligned}
2(w^\star - w_i)^\mathsf{T}(w_{i-1} - w_i) &= -\|w^\star - w_i - (w_{i-1} - w_i)\|^2 + \|w^\star - w_i\|^2 + \|w_{i-1} - w_i\|^2 \\
&= -\|\widetilde{w}_{i-1}\|^2 + \|\widetilde{w}_i\|^2 + \|w_{i-1} - w_i\|^2 \qquad (6.65)
\end{aligned}
$$

Substituting the last equation into (6.64) and rearranging terms gives:

$$
\begin{aligned}
\|\widetilde{w}_i\|^2 - \|\widetilde{w}_{i-1}\|^2 \leq &-\|w_{i-1} - w_i\|^2 - 2\mu_w(w_i - w^\star)^\mathsf{T}\mathcal{B}^\mathsf{T}(y_{i-1} - y^\star) \\
&- 2\mu_w(w_i - w^\star)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) \qquad (6.66)
\end{aligned}
$$

Using Assumption 6.1 we can bound the inner product:

$$
\begin{aligned}
(w_i - w^\star)^\mathsf{T}\nabla\mathcal{J}(w_{i-1}) &= (w_i - w_{i-1} + w_{i-1} - w^\star)^\mathsf{T}\nabla\mathcal{J}(w_{i-1}) \\
&\overset{(6.11)}{\geq} (w_i - w_{i-1})^\mathsf{T}\nabla\mathcal{J}(w_{i-1}) \\
&\quad + \mathcal{J}(w_{i-1}) - \mathcal{J}(w^\star) + \frac{\nu}{2}\|\widetilde{w}_{i-1}\|^2 \\
&= (w_i - w_{i-1})^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w_i) + \nabla\mathcal{J}(w_i)\big) \\
&\quad + \mathcal{J}(w_{i-1}) - \mathcal{J}(w^\star) + \frac{\nu}{2}\|\widetilde{w}_{i-1}\|^2 \qquad (6.67)
\end{aligned}
$$

We again use (6.11) in the last expression to get:

$$
\begin{aligned}
(w_i - w^\star)^\mathsf{T}\nabla\mathcal{J}(w_{i-1}) \geq\ &(w_i - w_{i-1})^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w_i)\big) \\
&+ \mathcal{J}(w_i) - \mathcal{J}(w_{i-1}) + \frac{\nu}{2}\|w_i - w_{i-1}\|^2 \\
&+ \mathcal{J}(w_{i-1}) - \mathcal{J}(w^\star) + \frac{\nu}{2}\|\widetilde{w}_{i-1}\|^2 \\
=\ &-(w_{i-1} - w_i)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w_i)\big) \\
&+ \frac{\nu}{2}\|w_i - w_{i-1}\|^2 + \mathcal{J}(w_i) - \mathcal{J}(w^\star) + \frac{\nu}{2}\|\widetilde{w}_{i-1}\|^2 \qquad (6.68)
\end{aligned}
$$

From (6.11) it holds that:

$$(w_i - w^\star)^\mathsf{T} \nabla \mathcal{J}(w^\star) \le \mathcal{J}(w_i) - \mathcal{J}(w^\star) - \frac{\nu}{2}\|\widetilde{w}_i\|^2 \tag{6.69}$$

Therefore, the last inner product in (6.66) can be bounded as follows:

$$\begin{aligned}
&- 2\mu_w (w_i - w^\star)^\mathsf{T} \big(\nabla \mathcal{J}(w_{i-1}) - \nabla \mathcal{J}(w^\star)\big) \\
&= -2\mu_w (w_i - w^\star)^\mathsf{T} \nabla \mathcal{J}(w_{i-1}) + 2\mu_w (w_i - w^\star)^\mathsf{T} \nabla \mathcal{J}(w^\star) \\
&\le 2\mu_w (w_{i-1} - w_i)^\mathsf{T} \big(\nabla \mathcal{J}(w_{i-1}) - \nabla \mathcal{J}(w_i)\big) \\
&\quad - \mu_w \nu \|w_i - w_{i-1}\|^2 - \mu_w \nu \|\widetilde{w}_{i-1}\|^2 - \mu_w \nu \|\widetilde{w}_i\|^2 \\
&\le \mu_w (2\delta - \nu) \|w_i - w_{i-1}\|^2 - \mu_w \nu (\|\widetilde{w}_{i-1}\|^2 + \|\widetilde{w}_i\|^2)
\end{aligned} \tag{6.70}$$

where the last step holds because

$$(w - z)^\mathsf{T} \big(\nabla_w \mathcal{J}(w) - \nabla_w \mathcal{J}(z)\big) \le \delta \|w - z\|^2 \tag{6.71}$$

holds by using the Cauchy-Schwartz inequality and (6.10). Substituting (6.70) into (6.66) gives (6.49).

6.C Dual Error Bound

In this appendix, we derive equality (6.50). It holds that:

$$\begin{aligned}
\|\widetilde{x}_{i-1}\|^2_{\mathcal{D}} + \|\widetilde{y}_{i-1}\|^2_{\mu_v^{-1}} =& \|x^\star - x_i + x_i - x_{i-1}\|^2_{\mathcal{D}} + \|y^\star - y_i + y_i - y_{i-1}\|^2_{\mu_v^{-1}} \\
=& \|x^\star - x_i\|^2_{\mathcal{D}} + \|y^\star - y_i\|^2_{\mu_v^{-1}} + \|x_i - x_{i-1}\|^2_{\mathcal{D}} \\
&+ \|y_i - y_{i-1}\|^2_{\mu_v^{-1}} + 2(x_i - x_{i-1})^\mathsf{T} \mathcal{D} (x^\star - x_i) \\
&+ 2(y_i - y_{i-1})^\mathsf{T} \mu_v^{-1} (y^\star - y_i)
\end{aligned} \tag{6.72}$$

Rearranging the last equality we have:

$$
\begin{aligned}
&\|\widetilde{y}_i\|^2_{\mu_v^{-1}} + \|\widetilde{x}_i\|^2_{\mathcal{D}} - \|\widetilde{y}_{i-1}\|^2_{\mu_v^{-1}} - \|\widetilde{x}_{i-1}\|^2_{\mathcal{D}} \\
&= -\|x_i - x_{i-1}\|^2_{\mathcal{D}} - \|y_i - y_{i-1}\|^2_{\mu_v^{-1}} + 2(x_{i-1} - x_i)^\mathsf{T}\mathcal{D}(x^\star - x_i) - 2(y_{i-1} - y_i)^\mathsf{T}\mu_v^{-1}(y_i - y^\star)
\end{aligned}
\tag{6.73}
$$

Note that:

$$
\begin{aligned}
(y_i - y^\star)^\mathsf{T}\mathcal{U}_1\Sigma(x_i - x^\star) &\overset{(a)}{=} (y_i - y_{i-1} + y_{i-1} - y^\star)^\mathsf{T}\mathcal{U}_1\Sigma(x_i - x^\star) \\
&\overset{(b)}{=} (\mathcal{U}_1^\mathsf{T}(y_i - y_{i-1}) + \mathcal{U}_1^\mathsf{T}y_{i-1})^\mathsf{T}\Sigma(x_i - x^\star) \\
&\overset{(6.38b)}{=} \big(\mathcal{U}_1^\mathsf{T}(\mu_v(\mathcal{B}w_i - b) + \mu_v\mathcal{U}_1\Sigma x_i) + \mathcal{U}_1^\mathsf{T}y_{i-1}\big)^\mathsf{T}\Sigma(x_i - x^\star) \\
&= \Bigg(\mathcal{U}_1^\mathsf{T}\big(y_{i-1} + \mu_v(\mathcal{B}w_i - b) + \mu_v\mathcal{U}_1\Sigma x_{i-1}\big) \\
&\quad + \mu_v\Sigma(x_i - x_{i-1})\Bigg)^\mathsf{T}\Sigma(x_i - x^\star) \\
&\overset{(6.38a)}{=} \Bigg(\mu_v(x_{i-1} - x_i) + \mu_v\Sigma(x_i - x_{i-1})\Bigg)^\mathsf{T}\Sigma(x_i - x^\star) \\
&= -\big(\mu_v(I - \Sigma)(x_i - x_{i-1})\big)^\mathsf{T}\Sigma(x_i - x^\star) \\
&\overset{(6.48)}{=} -(x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star)
\end{aligned}
\tag{6.74}
$$

where in step (b) we took $\mathcal{U}_1$ inside the first bracket and used $\mathcal{U}_1^\mathsf{T}y^\star = 0$ from (6.40b). From step (a) and the last step we get:

$$
(y_{i-1} - y^\star)^\mathsf{T}\mathcal{U}_1\Sigma(x_i - x^\star) = -(y_i - y_{i-1})^\mathsf{T}\mathcal{U}_1\Sigma(x_i - x^\star) - (x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star) \tag{6.75}
$$

Furthermore, note that:

$$
\begin{aligned}
&(y_{i-1} - y^\star)^\mathsf{T}(\mathcal{B}w_i - b - \mathcal{B}w^\star + b) \\
&\overset{(6.38b)}{=} (y_{i-1} - y^\star)^\mathsf{T}\left(-\mu_v^{-1}(y_{i-1} - y_i) - \mathcal{U}_1\Sigma x_i - \mathcal{B}w^\star + b\right) \\
&\overset{(6.40c)}{=} (y_{i-1} - y^\star)^\mathsf{T}\left(-\mu_v^{-1}(y_{i-1} - y_i) - \mathcal{U}_1\Sigma x_i + \mathcal{U}_1\Sigma x^\star\right) \\
&= -(y_{i-1} - y^\star)^\mathsf{T}\mu_v^{-1}(y_{i-1} - y_i) - (y_{i-1} - y^\star)^\mathsf{T}\mathcal{U}_1\Sigma(x_i - x^\star)
\end{aligned} \tag{6.76}
$$

Substituting (6.75) into (6.76), we have

$$
\begin{aligned}
&(y_{i-1} - y^\star)^\mathsf{T}(\mathcal{B}w_i - b - \mathcal{B}w^\star + b) \\
&= -(y_{i-1} - y^\star)^\mathsf{T}\mu_v^{-1}(y_{i-1} - y_i) \\
&\quad + (y_i - y_{i-1})^\mathsf{T}\mathcal{U}_1\Sigma(x_i - x^\star) + (x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star) \\
&= \left(-\mu_v^{-1}(y_{i-1} - y^\star) - \mathcal{U}_1\Sigma(x_i - x^\star)\right)^\mathsf{T}(y_{i-1} - y_i) + (x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star) \\
&\overset{(a)}{=} \Bigg(- \mu_v^{-1}(y_{i-1} - y^\star) + \mu_v^{-1}(y_{i-1} - y_i) + (\mathcal{B}w_i - b) - \mathcal{B}w^\star + b\Bigg)^\mathsf{T}(y_{i-1} - y_i) \\
&\qquad + (x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star) \\
&= \left(-\mu_v^{-1}(y_i - y^\star) + \mathcal{B}(w_i - w^\star)\right)^\mathsf{T}(y_{i-1} - y_i) + (x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star) \\
&= -(y_i - y^\star)^\mathsf{T}\mu_v^{-1}(y_{i-1} - y_i) + (w_i - w^\star)^\mathsf{T}\mathcal{B}^\mathsf{T}(y_{i-1} - y_i) + (x_i - x_{i-1})^\mathsf{T}\mathcal{D}(x_i - x^\star)
\end{aligned} \tag{6.77}
$$

where in step (a) we used (6.38b) and the optimality condition (6.40c). Re-arranging the last equation (6.77), we get

$$
\begin{aligned}
&- (y_{i-1} - y_i)^\mathsf{T}\mu_v^{-1}(y_i - y^\star) + (x_{i-1} - x_i)^\mathsf{T}\mathcal{D}(x^\star - x_i) \\
&= (y_{i-1} - y^\star)^\mathsf{T}\mathcal{B}(w_i - w^\star) - (y_{i-1} - y_i)^\mathsf{T}\mathcal{B}(w_i - w^\star)
\end{aligned} \tag{6.78}
$$

Substituting (6.78) into (6.73), we get,

$$\begin{aligned}\|\widetilde{\mathcal{Y}}_i\|^2_{\mu_v^{-1}} + \|\widetilde{\mathcal{X}}_i\|^2_{\mathcal{D}} - \|\widetilde{\mathcal{Y}}_{i-1}\|^2_{\mu_v^{-1}} - \|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mathcal{D}} &= -\|\mathcal{X}_i - \mathcal{X}_{i-1}\|^2_{\mathcal{D}} - \|\mathcal{Y}_i - \mathcal{Y}_{i-1}\|^2_{\mu_v^{-1}} \\ &\quad + 2(\mathcal{Y}_{i-1} - \mathcal{Y}^\star)^\mathsf{T}\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star) \\ &\quad - 2(\mathcal{Y}_{i-1} - \mathcal{Y}_i)^\mathsf{T}\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star) \end{aligned} \tag{6.79}$$

The last term of (6.79) can be rewritten as:

$$\begin{aligned}-2(\mathcal{Y}_{i-1} - \mathcal{Y}_i)^\mathsf{T}\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star) &= -\|\mathcal{Y}_{i-1} - \mathcal{Y}_i + \mu_v\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star)\|^2_{\mu_v^{-1}} \\ &\quad + \|\mathcal{Y}_{i-1} - \mathcal{Y}_i\|^2_{\mu_v^{-1}} + \|\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star)\|^2_{\mu_v} \\ &= -\|\Sigma(\mathcal{X}^\star - \mathcal{X}_i)\|^2_{\mu_v} + \|\mathcal{Y}_{i-1} - \mathcal{Y}_i\|^2_{\mu_v^{-1}} + \|\mathcal{B}(\mathcal{W}_i - \mathcal{W}^\star)\|^2_{\mu_v} \end{aligned} \tag{6.80}$$

where in the last step we used (6.38b), (6.40c), and $\mathcal{U}_1^\mathsf{T}\mathcal{U}_1 = I$. Substituting the last equality into (6.79), we get (6.50).

6.D Proof of Theorem 6.1

Let us introduce the energy function:

$$V(\widetilde{\mathcal{W}}_i, \widetilde{\mathcal{Y}}_i, \widetilde{\mathcal{X}}_i) = \|\widetilde{\mathcal{W}}_i\|^2 + \mu_w\|\widetilde{\mathcal{Y}}_i\|^2_{\mu_v^{-1}} + \|\widetilde{\mathcal{X}}_i\|^2_{\mathcal{D}} \tag{6.81}$$

Using (6.49) and (6.50) we have:

$$\begin{aligned}V(\widetilde{\mathcal{W}}_i, \widetilde{\mathcal{Y}}_i, \widetilde{\mathcal{X}}_i) - V(\widetilde{\mathcal{W}}_{i-1}, \widetilde{\mathcal{Y}}_{i-1}, \widetilde{\mathcal{X}}_{i-1}) &\leq -(1 + \mu_w\nu - 2\mu_w\delta)\|\mathcal{W}_i - \mathcal{W}_{i-1}\|^2 \\ &\quad - \mu_w\nu\left(\|\widetilde{\mathcal{W}}_{i-1}\|^2 + \|\widetilde{\mathcal{W}}_i\|^2\right) + \mu_w\|\mathcal{B}\widetilde{\mathcal{W}}_i\|^2_{\mu_v} \\ &\quad - \mu_w\|\mathcal{X}_i - \mathcal{X}_{i-1}\|^2_{\mathcal{D}} - \mu_w\|\Sigma\widetilde{\mathcal{X}}_i\|^2_{\mu_v} \end{aligned} \tag{6.82}$$

Note that:

$$\|\mathcal{B}\widetilde{w}_i\|^2_{\mu_v} \leq \mu_v \lambda_{\max}(\mathcal{B}^\mathsf{T}\mathcal{B})\|\widetilde{w}_i\|^2$$

Therefore, under condition (6.51), it holds that:

$$\begin{aligned}
&V(\widetilde{w}_i, \widetilde{y}_i, \widetilde{x}_i) - V(\widetilde{w}_{i-1}, \widetilde{y}_{i-1}, \widetilde{x}_{i-1}) \\
&\leq -\underbrace{(1 + \mu_w \nu - 2\mu_w \delta)}_{>0}\|w_i - w_{i-1}\|^2 - \mu_w \nu \|\widetilde{w}_{i-1}\|^2 \\
&\quad - \mu_w \underbrace{\left(\nu - \mu_v \lambda_{\max}(\mathcal{B}^\mathsf{T}\mathcal{B})\right)}_{>0}\|\widetilde{w}_i\|^2 - \mu_w \|x_i - x_{i-1}\|^2_{\mathcal{D}} - \mu_w \|\Sigma \widetilde{x}_i\|^2_{\mu_v} \leq 0 \qquad (6.83)
\end{aligned}$$

Since $V(\widetilde{w}_i, \widetilde{y}_i, \widetilde{x}_i)$ is non-negative, we conclude that the norm of the error is non-increasing and bounded. Since $V(\widetilde{w}_i, \widetilde{y}_i, \widetilde{x}_i)$ is non-negative, we conclude that the norm of the error is non-increasing and bounded. Iterating the above inequality we have:

$$\begin{aligned}
V(\widetilde{w}_i, \widetilde{y}_i, \widetilde{x}_i) \leq V(\widetilde{w}_0, \widetilde{y}_0, \widetilde{x}_0) - \sum_{j=1}^{i}\Bigg(&(1 + \mu_w \nu - 2\mu_w \delta)\|w_j - w_{j-1}\|^2 + \mu_w \nu \|\widetilde{w}_{j-1}\|^2 \\
&+ \mu_w \left(\nu - \mu_v \lambda_{\max}(\mathcal{B}^\mathsf{T}\mathcal{B})\right)\|\widetilde{w}_j\|^2 + \mu_w \|x_j - x_{j-1}\|^2_{\mathcal{D}} + \mu_w \|\Sigma \widetilde{x}_j\|^2_{\mu_v}\Bigg) \qquad (6.84)
\end{aligned}$$

and thus

$$\begin{aligned}
\sum_{j=1}^{\infty}\Bigg(&(1 + \mu_w \nu - 2\mu_w \delta)\|w_j - w_{j-1}\|^2 + \mu_w \nu \|\widetilde{w}_{j-1}\|^2 + \mu_w \left(\nu - \mu_v \lambda_{\max}(\mathcal{B}^\mathsf{T}\mathcal{B})\right)\|\widetilde{w}_j\|^2 \\
&+ \mu_w \|x_j - x_{j-1}\|^2_{\mathcal{D}} + \mu_w \|\Sigma \widetilde{x}_j\|^2_{\mu_v}\Bigg) \leq V(\widetilde{w}_0, \widetilde{y}_0, \widetilde{x}_0) \qquad (6.85)
\end{aligned}$$

Since the sum of the infinite positive terms is upper bounded by a constant, it holds that each term $(w_i - w_{i-1}), \widetilde{w}_{i-1}, \widetilde{w}_i, (x_i - x_{i-1})$, and $\Sigma \widetilde{x}_i$ must converge to zero.

6.E Proof of Theorem 6.2

In this Appendix, we show that the dual coupled diffusion strategy converges linearly for smooth cost functions and under the additional assumption that the matrix $\mathcal{B}$ has full row rank. From the structure of $\mathcal{B}$ in (6.33), it can be confirmed that $\mathcal{B}$ having full row rank is equivalent to assuming that each matrix $\text{blkcol}\{B_{e,k}\}_{e\in\mathcal{E}_k}$ has full row rank. This is illustrated in Fig. 6.6. Because two different agents belonging to the same cluster are located differently in $\mathcal{Y}^e$, it holds that the block rows of $\mathcal{B}$ are zeros except at one location. Recall that $B_{ek} \in \mathbb{R}^{S_e \times Q_k}$. Therefore, an equivalent statement is to say that $\text{blkcol}\{B_{e,k}\}_{e\in\mathcal{E}_k}$ has full row rank.

$$
\mathcal{B} = \begin{bmatrix}
\begin{bmatrix} B_{1,1} \\ 0 \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} 0 \\ B_{1,2} \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} 0 \\ 0 \\ B_{1,4} \\ 0 \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} 0 \\ 0 \\ 0 \\ B_{1,7} \end{bmatrix} \\
\begin{bmatrix} B_{2,1} \\ 0 \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} 0 \\ B_{2,4} \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} 0 \\ 0 \\ B_{2,6} \\ 0 \end{bmatrix} & \begin{bmatrix} 0 \\ 0 \\ 0 \\ B_{2,7} \end{bmatrix} \\
\begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} B_{3,2} \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} 0 \\ B_{3,3} \\ 0 \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} 0 \\ 0 \\ B_{3,5} \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix} & \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix}
\end{bmatrix}
$$

Figure 6.6: An illustration of the construction $\mathcal{B}$ for the network in Figure 6.2.

Recall from (6.66) that

$$
\begin{aligned}
\|\widetilde{\mathcal{W}}_i\|^2 - \|\widetilde{\mathcal{W}}_{i-1}\|^2 \leq & -\|\mathcal{W}_{i-1} - \mathcal{W}_i\|^2 - 2\mu_w(\mathcal{W}_i - \mathcal{W}^\star)^\mathsf{T}\mathcal{B}^\mathsf{T}(\mathcal{Y}_{i-1} - \mathcal{Y}^\star) \\
& - 2\mu_w(\mathcal{W}_i - \mathcal{W}^\star)^\mathsf{T}\big(\nabla\mathcal{J}(\mathcal{W}_{i-1}) - \nabla\mathcal{J}(\mathcal{W}^\star)\big) \qquad (6.86)
\end{aligned}
$$

It holds that

$$-2\mu_w(w_i - w^\star)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) = -2\mu_w(w_{i-1} - w^\star)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) + 2\mu_w(w_{i-1} - w_i)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) \tag{6.87}$$

The last term can be upper bounded by

$$\begin{aligned} &2\mu_w(w_{i-1} - w_i)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) \\ &= -\|w_{i-1} - w_i - \mu_w\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big)\|^2 + \|w_{i-1} - w_i\|^2 + \mu_w^2\|\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\|^2 \\ &\overset{(a)}{=} -\mu_w^2\|\mathcal{B}^\mathsf{T}\widetilde{y}_{i-1}\|^2 + \|w_{i-1} - w_i\|^2 + \mu_w^2\|\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\|^2 \\ &\le -\mu_w^2\|\mathcal{B}^\mathsf{T}\widetilde{y}_{i-1}\|^2 + \|w_{i-1} - w_i\|^2 + \mu_w^2\delta(w_{i-1} - w^\star)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) \end{aligned} \tag{6.88}$$

where in step (a) we used (6.34a) and (6.40a) with $\mathcal{R}(w) = 0$. The last inequality holds from (1.8) since $\mathcal{J}(w)$ has δ-Lipschitz gradients. Combining the last two equations we have

$$\begin{aligned} &-2\mu_w(w_i - w^\star)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) \\ &\le -\mu_w^2\|\mathcal{B}^\mathsf{T}\widetilde{y}_{i-1}\|^2 + \|w_{i-1} - w_i\|^2 - \mu_w(2 - \delta\mu_w)(w_{i-1} - w^\star)^\mathsf{T}\big(\nabla\mathcal{J}(w_{i-1}) - \nabla\mathcal{J}(w^\star)\big) \\ &\le -\mu_w^2\|\mathcal{B}^\mathsf{T}\widetilde{y}_{i-1}\|^2 + \|w_{i-1} - w_i\|^2 - \mu_w\nu(2 - \delta\mu_w)\|\widetilde{w}_{i-1}\|^2 \end{aligned}$$

where the last step holds from the strong-convexity condition (1.6) and $\mu_w < 2/\delta$. Substituting into (6.86) we get:

$$\|\widetilde{w}_i\|^2 \le \big(1 - \mu_w\nu(2 - \delta\mu_w)\big)\|\widetilde{w}_{i-1}\|^2 - \mu_w^2\|\mathcal{B}^\mathsf{T}\widetilde{y}_{i-1}\|^2 - 2\mu_w(w_i - w^\star)^\mathsf{T}\mathcal{B}^\mathsf{T}(y_{i-1} - y^\star) \tag{6.89}$$

Note that $-\mu_w\|x_i - x_{i-1}\|^2_{\mathcal{D}} \le 0$. Thus, multiplying (6.50) by μ_w, using $\mathcal{D} = \mu_v(\Sigma - \Sigma^2)$, and rearranging terms we get:

$$\begin{aligned}&\mu_w\|\widetilde{\mathcal{Y}}_i\|^2_{\mu_v^{-1}}+\mu_w\|\widetilde{\mathcal{X}}_i\|^2_{\mu_v\Sigma}=\mu_w\|\widetilde{\mathcal{Y}}_i\|^2_{\mu_v^{-1}}+\mu_w\|\widetilde{\mathcal{X}}_i\|^2_{\mathcal{D}+\mu_v\Sigma^2}\\&\leq\mu_w\|\mathcal{B}\widetilde{\mathcal{W}}_i\|^2_{\mu_v}+\mu_w\|\widetilde{\mathcal{Y}}_{i-1}\|^2_{\mu_v^{-1}}+\mu_w\|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mathcal{D}}+2\mu_w(\mathcal{Y}_{i-1}-\mathcal{Y}^\star)^{\mathsf{T}}\mathcal{B}(\mathcal{W}_i-\mathcal{W}^\star)\end{aligned}\tag{6.90}$$

Since $\mathcal{B}$ is full row rank, it holds that $\|\mathcal{B}^{\mathsf{T}}\widetilde{\mathcal{Y}}_{i-1}\|^2\geq\lambda_{\min}(\mathcal{B}\mathcal{B}^{\mathsf{T}})\|\widetilde{\mathcal{Y}}_{i-1}\|^2$. Using this bound and combining (6.89) and (6.90), we get:

$$\begin{aligned}&\|\widetilde{\mathcal{W}}_i\|^2+\|\widetilde{\mathcal{Y}}_i\|^2_{\frac{\mu_w}{\mu_v}}+\|\widetilde{\mathcal{X}}_i\|^2_{\mu_w\mu_v\Sigma}\leq\|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mu_w\mu_v(\Sigma-\Sigma^2)}\\&+\big(1-\mu_w\nu(2-\delta\mu_w)\big)\|\widetilde{\mathcal{W}}_{i-1}\|^2+\mu_w\mu_v\|\mathcal{B}\widetilde{\mathcal{W}}_i\|^2+\big(1-\mu_w\mu_v\lambda_{\min}(\mathcal{B}\mathcal{B}^{\mathsf{T}})\big)\|\widetilde{\mathcal{Y}}_{i-1}\|^2_{\frac{\mu_w}{\mu_v}}\end{aligned}\tag{6.91}$$

Since $\Sigma>0$ we have $-\|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mu_w\mu_v\Sigma^2}\leq-\lambda_r\|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mu_w\mu_v\Sigma}$. Substituting this bound into (6.91) and rearranging, we arrive at the following inequality:

$$\begin{aligned}\|\widetilde{\mathcal{W}}_i\|^2_{C_w}+\|\widetilde{\mathcal{Y}}_i\|^2_{\frac{\mu_w}{\mu_v}}+\|\widetilde{\mathcal{X}}_i\|^2_{\mu_w\mu_v\Sigma}\leq\big(1-\mu_w\nu(2-\delta\mu_w)\big)\|\widetilde{\mathcal{W}}_{i-1}\|^2+\gamma_2\|\widetilde{\mathcal{Y}}_{i-1}\|^2_{\frac{\mu_w}{\mu_v}}\\+(1-\lambda_r)\|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mu_w\mu_v\Sigma}\end{aligned}\tag{6.92}$$

where $C_w=I-\mu_w\mu_v\mathcal{B}^{\mathsf{T}}\mathcal{B}$ and $\gamma_2=1-\mu_w\mu_v\lambda_{\min}(\mathcal{B}\mathcal{B}^{\mathsf{T}})$. Let $\gamma_1=\big(1-\mu_w\nu(1-\delta\mu_w)\big)$. Note that

$$\begin{aligned}\big(1-\mu_w\nu(2-\delta\mu_w)\big)\|\widetilde{\mathcal{W}}_{i-1}\|^2&=\gamma_1\|\widetilde{\mathcal{W}}_{i-1}\|^2_{C_w}-\mu_w\nu\|\widetilde{\mathcal{W}}_{i-1}\|^2+\gamma_1\mu_w\mu_v\|\widetilde{\mathcal{W}}_{i-1}\|^2_{\mathcal{B}^{\mathsf{T}}\mathcal{B}}\\&\leq\gamma_1\|\widetilde{\mathcal{W}}_{i-1}\|^2_{C_w}-\mu_w\big(\nu-\mu_v\lambda_{\max}(\mathcal{B}^{\mathsf{T}}\mathcal{B})\big)\|\widetilde{\mathcal{W}}_{i-1}\|^2\\&\leq\gamma_1\|\widetilde{\mathcal{W}}_{i-1}\|^2_{C_w}\end{aligned}$$

where the first inequality holds since $\gamma_1<1$ for $\mu_w<\frac{1}{2\delta-\nu}\leq\frac{1}{\delta}$ and the last inequality holds under (6.51). Substituting into (6.92) we get:

$$\|\widetilde{\mathcal{W}}_i\|^2_{C_w}+\|\widetilde{\mathcal{Y}}_i\|^2_{\frac{\mu_w}{\mu_v}}+\|\widetilde{\mathcal{X}}_i\|^2_{\mu_w\mu_v\Sigma}\leq\gamma_1\|\widetilde{\mathcal{W}}_{i-1}\|^2_{C_w}+\gamma_2\|\widetilde{\mathcal{Y}}_{i-1}\|^2_{\frac{\mu_w}{\mu_v}}+(1-\lambda_r)\|\widetilde{\mathcal{X}}_{i-1}\|^2_{\mu_w\mu_v\Sigma}$$

Under condition (6.51), it holds that $\mu_w\mu_v<1/\lambda_{\max}(\mathcal{B}^{\mathsf{T}}\mathcal{B})$; thus, $\gamma_2=1-\mu_w\mu_v\lambda_{\min}(\mathcal{B}\mathcal{B}^{\mathsf{T}})<1$

and $C_w = I - \mu_w \mu_v \mathcal{B}^{\mathsf{T}} \mathcal{B} > 0$. Since $0 < \lambda_r < 1$, we have $1 - \lambda_r < 1$. By iterating the previous inequality we arrive at (6.52).

CHAPTER 7

Conclusion and Future Directions

Conclusion

In this dissertation, we have studied the performance and linear convergence of decentralized multi-agent optimization algorithms. The main conclusions are summarized below.

- We studied the classical incremental primal-dual gradient algorithm (2.3) for the solution of constrained optimization problems. Through an original proof we established its linear convergence. We also related the incremental implementation to the non-incremental Arrow-Hurwicz implementation (2.4) and established its linear convergence as well.
- We proposed a general adapt-then-combine algorithmic framework that captures most existing decentralized gradient based algorithms. We then established its linear convergence and showed that the ATC structure is more stable than the non-ATC structure. We also studied the benefits of the augmented Lagrangian penalty term on the convergence rate of decentralized algorithms and highlighted several benefits of this term and gave cases where it is not that beneficial.
- We established the linear convergence of a proximal decentralized algorithm in the presence of a non-smooth term. With this result, we closed the linear convergence gap between decentralized and centralized proximal gradient algorithms.
- We studied decentralized multi-agent optimization problems under general multiple coupling across the agents. Specifically, motivated by real applications, we considered scenarios where there can exist multiple consensus variables or multiple coupling constraints with only a subset of agents involved in them. We then showed how to

design algorithms to exploit these structures. More importantly, we showed theoretically that algorithms exploiting this structure can greatly improve the convergence rate compared to algorithms that do not exploit such structure.

Future Directions

We now list several future directions that are worth investigating:

- In this dissertation, we considered undirected and static networks. In some applications, the network is directed where agents can communicate only in one direction. In these cases, the combination matrix A is not symmetric any longer. Moreover, the combination matrix can be time varying where the network topology changes with time. Thus, is it possible to extend the ATC algorithm framework to directed and time-varying networks? For example, can the push-pull idea [63, 64] or the push-sum technique [175] be used to handle time-varying digraphs of the proposed framework?

- We showed that any decentralized proximal gradient algorithm cannot achieve *global* linear convergence in the presence of more than one non-smooth term in the worst case. However, *asymptotic* linear convergence have been established under piecewise linear quadratic costs [130]. Can we establish the asymptotic linear convergence of proximal decentralized algorithms under some other more practical conditions on the costs?

- In this work, we studied deterministic and convex problems. One direction is to study the convergence of the proposed framework in stochastic and non-convex settings. While decentralized non-convex problems have been studied widely, the efficient escape from saddle-points have only been established recently for the diffusion algorithm [176]. Recently, it has been shown in [177] that the performance of the diffusion algorithm can degrade over sparse networks compared to exact diffusion, which is a special case of our proposed ATC framework. Therefore, studying the efficient escape from saddle-points for the proposed method in stochastic and non-convex settings is of interest.

REFERENCES

[1] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via alternating direction method of multipliers," *Found. Trends Mach. Lear.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[3] Z. Peng, Y. Xu, M. Yan, and W. Yin, "ARock: an algorithmic framework for asynchronous parallel coordinate updates," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A2851–A2879, 2016.

[4] V. Smith, S. Forte, M. Chenxin, M. Takac, M. I. Jordan, and M. Jaggi, "CoCoA: A general framework for communication-efficient distributed optimization," *Journal of Machine Learning Research*, vol. 18, no. 230, pp. 1–49, 2018.

[5] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *11th Symposium on Operating Systems Design and Implementation (OSDI)*, Broomfield, Denver, Colorado, 2014, pp. 583–598.

[6] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, 2017, pp. 5330–5340.

[7] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," in *International Conference on Machine Learning (ICML)*, Stockholm, Sweden, 2018, pp. 1–10.

[8] A. H. Sayed, "Adaptation, learning, and optimization over neworks." *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[9] G. Inalhan, D. M. Stipanovic, and C. J. Tomlin, "Decentralized optimization, with application to multiple aircraft coordination," in *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 1, Las Vegas, NV, USA, 2002, pp. 1147–1155.

[10] M. Tillerson, G. Inalhan, and J. P. How, "Co-ordination and control of distributed spacecraft systems using convex optimization techniques," *International Journal of Robust and Nonlinear Control*, vol. 12, no. 2-3, pp. 207–242, 2002.

[11] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings*, vol. 4, Salt Lake City, UT, USA, May 2001, pp. 2033–2036.

[12] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[13] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *em Proc. International Symposium on Information Processing in Sensor Networks*. Berkeley, California, USA: ACM, April 2004, pp. 20–27.

[14] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *International Symposium on Information Processing in Sensor Networks*, Boise, ID, USA, 2005, pp. 63–70.

[15] M. Alizadeh, X. Li, Z. Wang, A. Scaglione, and R. Melton, "Demand-side management in the smart grid: Information processing for the power switch," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 55–67, 2012.

[16] D. Chazan and W. Miranker, "Chaotic relaxation," *Linear Algebra and its Applications*, vol. 2, no. 2, pp. 199–222, 1969.

[17] G. M. Baudet, "Asynchronous iterative methods for multiprocessors," *Journal of the ACM (JACM)*, vol. 25, no. 2, pp. 226–244, 1978.

[18] D. Bertsekas, "Distributed dynamic programming," *IEEE Transactions on Automatic Control*, vol. 27, no. 3, pp. 610–616, 1982.

[19] D. P. Bertsekas, "Distributed asynchronous computation of fixed points," *Mathematical Programming*, vol. 27, no. 1, pp. 107–120, 1983.

[20] J. N. Tsitsiklis, "Problems in decentralized decision making and computation." Ph.D. dissertation, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1984.

[21] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, 1986.

[22] V. Kibardin, "Decomposition into functions in the minimization problem," *Avtomatika I Telemekhanika*, no. 9, pp. 66–79, 1979.

[23] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problem," *SIAM J. Optim.*, vol. 7, no. 4, pp. 913–926, Nov. 1997.

[24] M. V. Solodov and S. K. Zavriev, "Error stability properties of generalized gradient-type algorithms," *Journal of Optimization Theory and Applications*, vol. 98, no. 3, pp. 663–680, Sep 1998.

[25] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM J. Optim.*, vol. 12, no. 1, pp. 109–138, Jul. 2001.

[26] A. Nedich, D. P. Bertsekas, and V. S. Borkar, "Distributed asynchronous incremental subgradient methods," *Studies in Computational Mathematics*, vol. 8, no. C, pp. 381–407, 2001.

[27] M. G. Rabbat and R. D. Nowak, "Decentralized source localization and tracking," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Montreal, QC, Canada, 2004, pp. 921–924.

[28] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.

[29] ——, "Distributed processing over adaptive networks," in *Proc. Adaptive Sensor Array Processing Workshop*, MIT Lincoln Laboratory, MA, June 2006, pp. 1–5.

[30] A. H. Sayed and C. Lopes, "Adaptive estimation algorithms over distributed networks," in *Proc. 21st IEICE Signal Processing Symposium*, Kyoto, Japan, Nov. 2006, pp. 1–5.

[31] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 3, Honolulu, HI, USA, April 2007, pp. 917–920.

[32] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "A diffusion RLS scheme for distributed estimation over adaptive networks," in *IEEE Workshop on Signal Processing Advances in Wireless Communications*, Helsinki, Finland, June 2007, pp. 1–5.

[33] A. Nedic and A. Ozdaglar, "On the rate of convergence of distributed subgradient methods for multi-agent optimization," in *46th IEEE Conference on Decision and Control*, New Orleans, LA, USA, Dec. 2007, pp. 4711–4716.

[34] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *47th IEEE Conference on Decision and Control*, Dec. 2008, pp. 4185–4190.

[35] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 5, pp. 1865–1877, 2008.

[36] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, 2008.

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

[38] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, Apr. 2014.

[39] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, “Exact diffusion for distributed optimization and learning-Part I: Algorithm development,” *IEEE Transactions on Signal Processing*, vol. 67, no. 3, pp. 708–723, Feb. 2019.

[40] A. H. Sayed, “Diffusion adaptation over networks,” in *Academic Press Library in Signal Processing*. Elsevier, 2014, vol. 3, pp. 323–453.

[41] J. Chen and A. H. Sayed, “Distributed pareto optimization via diffusion strategies,” *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 2, pp. 205–220, April 2013.

[42] K. Yuan, Q. Ling, and W. Yin, “On the convergence of decentralized gradient descent,” *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835–1854, 2016.

[43] A. I. Chen and A. Ozdaglar, “A fast distributed proximal-gradient method,” in *Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, Oct. 2012, pp. 601–608.

[44] D. Jakovetić, J. Xavier, and J. M. Moura, “Fast distributed gradient methods,” *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, 2014.

[45] J. Wang and N. Elia, “Control approach to distributed optimization,” in *48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Allerton, IL, USA, Sept.-Oct. 2010, pp. 557–561.

[46] H. Terelius, U. Topcu, and R. M. Murray, “Decentralized multi-agent optimization via dual decomposition,” *IFAC proceedings volumes*, vol. 44, no. 1, pp. 11 245–11 251, 2011.

[47] D. Jakovetic, J. Xavier, and J. M. Moura, “Cooperative convex optimization in networked systems: Augmented Lagrangian algorithms with directed gossip communication,” *IEEE Transactions on Signal Processing*, vol. 59, no. 8, pp. 3889–3902, 2011.

[48] J. Wang and N. Elia, “A control perspective for centralized and distributed convex optimization,” in *IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, Dec. 2011, pp. 3800–3805.

[49] E. Wei and A. E. Ozdaglar, “Distributed alternating direction method of multipliers,” in *IEEE Conference on Decision and Control (CDC)*, Maui, HI, USA, Dec. 2012, pp. 5445–5450.

[50] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, “On the linear convergence of the ADMM in decentralized consensus optimization,” *IEEE Trans. Signal Process.*, vol. 62, no. 7, pp. 1750–1761, 2014.

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

[52] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Explicit convergence rate of a distributed alternating direction method of multipliers," *IEEE Transactions on Automatic Control*, vol. 61, no. 4, pp. 892–904, 2016.

[53] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, Jan. 2015.

[54] Q. Ling, W. Shi, G. Wu, and A. Ribeiro, "DLM: Decentralized linearized alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 63, pp. 4051–4064, 2015.

[55] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[56] Z. Li, W. Shi, and M. Yan, "A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates," *IEEE Transactions on Signal Processing*, vol. 67, no. 17, pp. 4494–4506, Sept. 2019.

[57] Y. Sun, G. Scutari, and D. Palomar, "Distributed nonconvex multiagent optimization over time-varying networks," in Proc. *Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA, Nov. 2016, pp. 788–794.

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

[59] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes," in Proc. 54th *IEEE Conference on Decision and Control (CDC)*, Osaka, Japan, 2015, pp. 2055–2060.

[60] A. Nedić, A. Olshevsky, W. Shi, and C. A. Uribe, "Geometrically convergent distributed optimization with uncoordinated step-sizes," in Proc. *American Control Conference (ACC)*, Seattle, WA, USA, May 2017, pp. 3950–3955.

[61] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1245–1260, Sept. 2018.

[62] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

[63] S. Pu, W. Shi, J. Xu, and A. Nedić, "A push-pull gradient method for distributed optimization in networks," in Proc. IEEE *Conference on Decision and Control (CDC)*, Miami Beach, FL, USA, Dec. 2018, pp. 3385–3390.

[64] R. Xin and U. A. Khan, "A linear algorithm for optimization over directed graphs with geometric convergence," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 315–320, 2018.

[65] G. Scutari and Y. Sun, "Distributed nonconvex constrained optimization over time-varying digraphs," *Mathematical Programming*, vol. 176, no. 1-2, pp. 497–544, 2019.

[66] J. Chen, S.-Y. Tu, and A. H. Sayed, "Distributed optimization via diffusion adaptation," in *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, San Juan, Puerto Rico, Dec. 2011, pp. 281–284.

[67] R. Olfati-Saber and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Proceedings of the 44th IEEE Conference on Decision and Control*, Seville, Spain, Dec. 2005, pp. 6698–6703.

[68] M. Zhu and S. Martinez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[69] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process*, vol. 58, no. 3, p. 1035, 2010.

[70] S. Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, 2012.

[71] L. Walras, *Elements Deconomie Politique Pure, ou, Theorie de la Richesse Sociale*. F. Rouge, 1896.

[72] K. J. Arrow and G. Debreu, "Existence of an equilibrium for a competitive economy," *Econometrica: Journal of the Econometric Society*, pp. 265–290, 1954.

[73] H. Uzawa, "Walras' tatonnement in the theory of exchange," *The Review of Economic Studies*, vol. 27, no. 3, pp. 182–194, 1960.

[74] G. M. Heal, "Planning without prices," *The Review of Economic Studies*, vol. 36, no. 3, pp. 347–362, 1969.

[75] L. Hurwicz, "The design of mechanisms for resource allocation," *The American Economic Review*, vol. 63, no. 2, pp. 1–30, 1973.

[76] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches*. MIT Press, 1988.

[77] J. F. Kurose and R. Simha, "A microeconomic approach to optimal resource allocation in distributed computer systems," *IEEE Transactions on computers*, vol. 38, no. 5, pp. 705–717, 1989.

[78] Y. Ho, L. Servi, and R. Suri, "A class of center-free resource allocation algorithms," *IFAC Proceedings Volumes*, vol. 13, no. 6, pp. 475–482, 1980.

[79] D. P. Palomar and M. Chiang, "Alternative distributed algorithms for network utility maximization: Framework and applications," *IEEE Transactions on Automatic Control*, vol. 52, no. 12, pp. 2254–2269, Dec. 2007.

[80] R. Halvgaard, L. Vandenberghe, N. K. Poulsen, H. Madsen, and J. B. Jorgensen, "Distributed model predictive control for smart energy systems," *IEEE Trans. Smart Grid*, vol. 7, no. 3, pp. 1675–1682, April 2016.

[81] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, "Distributed basis pursuit," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1942–1956, 2012.

[82] Z. Shen, J. G. Andrews, and B. L. Evans, "Adaptive resource allocation in multiuser OFDM systems with proportional rate constraints," *IEEE Transactions on Wireless Communications*, vol. 4, no. 6, pp. 2726–2737, 2005.

[83] S. A. Alghunaim and A. H. Sayed, "Linear convergence of primal-dual gradient methods and their performance in distributed optimization," *arXiv preprint:1904.01196*, April 2019.

[84] S. A. Alghunaim, K. Yuan, and A. H. Sayed, "A linearly convergent proximal gradient algorithm for decentralized optimization," in *Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, Dec. 2019, pp. 2844–2854, available on arXiv:1905.07996.

[85] S. A. Alghunaim, E. K. Ryu, K. Yuan, and A. H. Sayed, "Decentralized proximal gradient algorithms with linear convergence rates," *IEEE Transactions on Automatic Control*, 2019.

[86] S. A. Alghunaim, K. Yuan, and A. H. Sayed, "A proximal diffusion strategy for multiagent optimization with sparse affine constraints," *IEEE Transactions on Automatic Control*, 2019, to appear, available on arXiv:1810.02124.

[87] S. A. Alghunaim and A. H. Sayed, "Distributed coupled multi-agent stochastic optimization," *IEEE Transactions on Automatic Control*, vol. 65, no. 1, pp. 175–190, Jan 2020, also available on arXiv:1712.08817.

[88] S. A. Alghunaim, K. Yuan, and A. H. Sayed, "Dual coupled diffusion for distributed optimization with affine constraints," in *Proc. IEEE CDC*, Miami Beach, FL, USA, Dec. 2018, pp. 829–834.

[89] S. A. Alghunaim and A. H. Sayed, "Distributed coupled learning over adaptive networks," in *Proc. IEEE ICASSP*, Calgary, Canada, April 2018, pp. 6353–6357.

[90] S. A. Alghunaim, K. Yuan, and A. H. Sayed, "Decentralized exact coupled optimization," in *Proc. Allerton Conference on Communication, Control, and Computing*, Allerton, IL, USA, October 2017, pp. 338– 345.

[91] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[92] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[93] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning-Part II: Convergence analysis," *IEEE Transactions on Signal Processing*, vol. 67, no. 3, pp. 724–739, Feb. 2019.

[94] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[95] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2013, vol. 87.

[96] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[97] R. Ayadi, I. Kammoun, and M. Siala, "Optimization of the pulse shape of OFDM systems using the Arrow-Hurwicz algorithm," in *International Symposium on Wireless Communication Systems*, Trondheim, Norway, Dec. 2007, pp. 91–95.

[98] J. Chen and V. K. Lau, "Convergence analysis of saddle point problems in time varying wireless systems—control theoretical approach," *IEEE Transactions on Signal Processing*, vol. 60, no. 1, pp. 443–452, 2012.

[99] A. Cherukuri and J. Cortes, "Initialization-free distributed coordination for economic dispatch under varying loads and generator commitment," *Automatica*, vol. 74, pp. 183–193, 2016.

[100] S. V. Macua, J. Chen, S. Zazo, and A. H. Sayed, "Distributed policy evaluation under multiple behavior strategies," *IEEE Transactions on Automatic Control*, vol. 60, no. 5, pp. 1260–1274, 2015.

[101] L. Cassano, S. A. Alghunaim, and A. H. Sayed, "Team policy learning for multi-agent reinforcement learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, United Kingdom, May 2019, pp. 3062–3066.

[102] D. Feijer and F. Paganini, "Stability of primal–dual gradient dynamics and applications to network optimization," *Automatica*, vol. 46, no. 12, pp. 1974–1981, 2010.

[103] T. Kose, "Solutions of saddle value problems by differential equations," *Econometrica, Journal of the Econometric Society*, pp. 59–70, 1956.

[104] K. J. Arrow, L. Hurwicz, and H. Uzawa, *Studies in Linear and Nonlinear Programming*. Stanford University Press, Palo Alto, 1958.

[105] B. Polyak, "Iterative methods using Lagrange multipliers for solving extremal problems with constraints of the equation type," *USSR Computational Mathematics and Mathematical Physics*, vol. 10, no. 5, pp. 42–52, 1970.

[106] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Academic press, 2014.

[107] M. Kallio and C. H. Rosa, "Large-scale convex optimization via saddle point computation," *Operations Research*, vol. 47, no. 1, pp. 93–101, 1999.

[108] A. Nedic and A. Ozdaglar, "Subgradient methods for saddle-point problems," *Journal of Optimization Theory and Applications*, vol. 142, no. 1, pp. 205–228, 2009.

[109] A. Cherukuri, E. Mallada, and J. Cortes, "Asymptotic convergence of constrained primal–dual dynamics," *Systems & Control Letters*, vol. 87, pp. 10–15, 2016.

[110] J. Cortes and S. K. Niederlander, "Distributed coordination for nonsmooth convex optimization via saddle-point dynamics," *Journal of Nonlinear Science*, vol. 29, no. 4, pp. 1247–1272, Aug 2019.

[111] N. K. Dhingra, S. Z. Khong, and M. R. Jovanovic, "The proximal augmented Lagrangian method for nonsmooth composite optimization," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2861–2868, July 2019.

[112] G. Qu and N. Li, "On the exponential stability of primal-dual gradient dynamics," *IEEE Control Systems Letters*, vol. 3, no. 1, pp. 43–48, Jan. 2019.

[113] S. S. Du and W. Hu, "Linear convergence of the primal-dual gradient method for convex-concave saddle point problems without strong convexity," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, Naha, Okinawa, Japan, April 2019, pp. 196–205.

[114] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.

[115] ——, "On the ergodic convergence rates of a first-order primal–dual algorithm," *Mathematical Programming*, vol. 159, no. 1-2, pp. 253–287, Sept. 2016.

[116] O. Fercoq and P. Bianchi, "A coordinate descent primal-dual algorithm with large step size and possibly non separable functions," *SIAM Journal on Optimization*, vol. 29, no. 1, pp. 100–134, 2019.

[117] M. Yan, "A new primal–dual algorithm for minimizing the sum of three functions with a linear operator," *Journal of Scientific Computing*, vol. 76, no. 3, pp. 1698–1717, 2018.

[118] P. L. Combettes, L. Condat, J.-C. Pesquet, and B. Vũ, "A forward-backward view of some primal-dual optimization methods in image recovery," in *IEEE International Conference on Image Processing (ICIP)*, Paris, France, 2014, pp. 4141–4145.

[119] N. Komodakis and J.-C. Pesquet, "Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 31–54, 2015.

[120] A. J. Laub, *Matrix Analysis For Scientists And Engineers.* SIAM, PA, USA, 2004.

[121] A. Sundararajan, B. Van Scoy, and L. Lessard, "A canonical form for first-order distributed optimization algorithms," in Proc. *American Control Conference (ACC)*, Philadelphia, PA, USA, Jul. 2019, pp. 4075–4080, available on arXiv:1809.08709.

[122] D. Jakovetic, "A unification and generalization of exact distributed first-order methods," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 5, no. 1, pp. 31–46, 2019.

[123] A. Sundararajan, B. Van Scoy, and L. Lessard, "Analysis and design of first-order distributed optimization algorithms over time-varying graphs," *arXiv preprint:1907.05448*, July 2019.

[124] Z. J. Towfic and A. H. Sayed, "Stability and performance limits of adaptive primal-dual networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2888–2903, 2015.

[125] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2012.

[126] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, "A decentralized second-order method with exact linear convergence rate for consensus optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 507–522, 2016.

[127] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulie, "Optimal algorithms for smooth and strongly convex distributed optimization in networks," in *International Conference on Machine Learning (ICML)*, Stockholm, Sweden, 2017, pp. 3027–3036.

[128] N. S. Aybat, Z. Wang, T. Lin, and S. Ma, "Distributed linearized alternating direction method of multipliers for composite convex consensus optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 1, pp. 5–20, 2018.

[129] W. Shi, Q. Ling, G. Wu, and W. Yin, "A proximal gradient algorithm for decentralized composite optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 22, pp. 6013–6023, 2015.

[130] P. Latafat, N. M. Freris, and P. Patrinos, "A new randomized block-coordinate primal-dual proximal algorithm for distributed optimization," IEEE *Transactions on Automatic Control*, vol. 64, no. 10, pp. 4050–4065, Oct. 2019.

[131] L. He, A. Bian, and M. Jaggi, "COLA: Decentralized linear learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Montreal, Canada, 2018, pp. 4536–4546.

[132] P. Bianchi, W. Hachem, and F. Iutzeler, “A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization,” *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 2947–2957, 2016.

[133] B. Woodworth and N. Srebro, “Tight complexity bounds for optimizing composite objectives,” in *Advances in Neural Information Processing Systems (NIPS)*, Dec. 2016, pp. 3639–3647.

[134] Y. Arjevani and O. Shamir, “Communication complexity of distributed convex learning and optimization,” in *Advances in Neural Information Processing Systems (NIPS)*, Dec. 2015, pp. 1756–1764.

[135] A. Beck, *First-Order Methods in Optimization.* SIAM, PA, 2017.

[136] J. Chen, L. Tang, J. Liu, and J. Ye, “A convex formulation for learning shared structures from multiple tasks,” in *Proc. ICML*, Montreal, QC, Canada, June 2009, pp. 137–144.

[137] O. Chapelle, P. Shivaswmy, K. Q. Vadrevu, S. Weinberger, Y. Zhang, and B. Tseng, “Multi-task learning for boosting with applications to web search ranking,” in *Proc. ACM SIGKDD*, Washington, DC, USA, Jul. 2010, pp. 1189–1198.

[138] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, NJ, 1993.

[139] J. Plata-Chaves, A. Bertrand, and M. Moonen, “Incremental multiple error filtered-X LMS for node-specific active noise control over wireless acoustic sensor networks,” *in IEEE Sensor Array and Multichannel Signal Processing Workshop*, 2016.

[140] F. Cattivelli and A. H. Sayed, “Distributed nonlinear Kalman filtering with applications to wireless localization,” in *Proc. IEEE ICASSP*, Dallas, TX, USA, Mar. 2010, pp. 3522–3525.

[141] V. Kekatos and G. B. Giannakis, “Distributed robust power system state estimation,” *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1617–1626, May 2013.

[142] I. Necoara, V. Nedelcu, and I. Dumitrache, “Parallel and distributed optimization methods for estimation and control in networks,” *Journal of Process Control*, vol. 21, no. 5, pp. 756–766, Jun 2011.

[143] J. Mota, J. Xavier, P. Aguiar, and M. Puschel, “Distributed optimization with local domains: Application in MPC and network flows,” *IEEE Trans. Autom. Contr.*, vol. 60, no. 7, pp. 2004–2009, July 2015.

[144] T. H. Summers and J. Lygeros, “Distributed model predictive consensus via the alternating directoin method of multipliers,” in *Proceedings of the 50th Annual Allerton Confrence on Communication, Control, and Computing*, Monticello, IL, USA, 2012, pp. 79–84.

[145] T. Erseghe, "A distributed and scalable processing method based upon ADMM," *IEEE Signal Processing Letters*, vol. 19, no. 9, pp. 563–566, 2012.

[146] Y. Pu, M. N. Zeilinger, and C. N. Jones, "Inexact fast alternating minimization algorithm for distributed model predictive control," in *IEEE Conference on Decision and Control (CDC)*, Los Angeles, CA, USA, 2014, pp. 5915–5921.

[147] I. Notarnicola, R. Carli, and G. Notarstefano, "Distributed partitioned big-data optimization via asynchronous dual decomposition," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1910–1919, 2018.

[148] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Diffusion LMS for multitask problems with local linear equality constraints," *IEEE Trans. Signal Process*, vol. 65, no. 19, pp. 4979 – 4993, 2017.

[149] R. Arablouei, S. Werner, Y.-F. Huang, and K. Dougancay, "Distributed least mean-square estimation with partial diffusion," *IEEE Transactions on Signal Processing*, vol. 62, no. 2, pp. 472–484, 2014.

[150] C. Richard, J. Chen, S. K. Ting, and A. H. Sayed, "Group diffusion LMS," in *Proc. IEEE ICASSP*, Shanghai, China, Mar. 2016, pp. 4925–4929.

[151] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks," *IEEE Trans. Signal Process*, vol. 62, no. 16, pp. 4129–4144, August 2014.

[152] J. Zhou, L. Yuan, J. Liu, and J. Ye, "A multi-task learning formulation for predicting disease progression," in *Proc. ACM SIGKDD*, San Diego, CA, USA, Aug. 2011, pp. 814–822.

[153] S. Kar, G. Hug, J. Mohammadi, and J. M. Moura, "Distributed state estimation and energy management in smart grids: A consensus + innovations approach," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 6, pp. 1022–1038, 2014.

[154] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1464–1475, 2013.

[155] P. Giselsson, M. D. Doan, T. Keviczky, B. De Schutter, and A. Rantzer, "Accelerated gradient methods and dual decomposition in distributed model predictive control," *Automatica*, vol. 49, no. 3, pp. 829–833, Mar. 2013.

[156] A. J. Wood and B. F. Wollenberg, *Power Generation, Operation, and Control.* Wiley, NY, 2012.

[157] Y. Xu and Z. Li, "Distributed optimal resource management based on the consensus algorithm in a microgrid," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2584–2592, 2015.

[158] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Dynamic network energy management via proximal message passing," *Foundations and Trends in Optimization*, vol. 1, no. 2, pp. 73–126, 2014.

[159] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem.* Elsevier, 1992, vol. 53.

[160] P. Chalermsook and J. Fakcharoenphol, "Simple distributed algorithms for approximating minimum Steiner trees," in *International Computing and Combinatorics Conference*, NY, USA, August 2005, pp. 380–389.

[161] M. Bezensek and B. Robic, "A survey of parallel and distributed algorithms for the Steiner tree problem," *International Journal of Parallel Programming*, vol. 42, no. 2, pp. 287–319, March 2014.

[162] R. Rostami, G. Costantini, and D. Gorges, "ADMM-based distributed model predictive control: Primal and dual approaches," in *IEEE Conference on Decision and Control (CDC)*, Melbourne, Australia, Dec. 2017.

[163] I. Necoara and V. Nedelcu, "On linear convergence of a distributed dual gradient algorithm for linearly constrained separable convex problems," *Automatica*, vol. 55, pp. 209–216, 2015.

[164] S. Lee, N. Chatzipanagiotis, and M. M. Zavlanos, "A distributed augmented Lagrangian method for model predictive control," in *IEEE 56th Annual Conference on Decision and Control (CDC)*, Melbourne, Australia, Dec. 2017, pp. 2888–2893.

[165] T.-H. Chang, A. Nedić, and A. Scaglione, "Distributed constrained optimization by consensus-based primal-dual perturbation method," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1524–1538, June 2014.

[166] T.-H. Chang, "A proximal dual consensus ADMM method for multi-agent constrained optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 14, pp. 3719–3734, July 2016.

[167] I. Notarnicola and G. Notarstefano, "Constraint-coupled distributed optimization: Relaxation and duality approach," *IEEE Transactions on Control of Network Systems*, June 2019, to appear (early access). Also available on arXiv: 1711.09221, Nov. 2017.

[168] A. Falsone, K. Margellos, S. Garatti, and M. Prandini, "Dual decomposition for multi-agent distributed optimization with coupling constraints," *Automatica*, vol. 84, pp. 149–158, October 2017.

[169] Y. Xu, T. Han, K. Cai, Z. Lin, G. Yan, and M. Fu, "A distributed algorithm for resource allocation over dynamic digraphs," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2600–2612, May 2017.

[170] L. Xiao and S. Boyd, "Optimal scaling of a gradient method for distributed resource allocation," *Journal of Optimization Theory and Applications*, vol. 129, no. 3, pp. 469–488, 2006.

[171] H. Lakshmanan and D. P. De Farias, "Decentralized resource allocation in dynamic networks of agents," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 911–940, 2008.

[172] B. Johansson and M. Johansson, "Distributed non-smooth resource allocation over a network," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with the 28th Chinese Control Conference*, Shanghai, China, 2009, pp. 1678–1683.

[173] F. Hua, R. Nassif, C. Richard, and H. Wang, "Penalty-based multitask estimation with non-local linear equality constraints," in *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, Curacao, Dec. 2017, pp. 433–437.

[174] S. Lee and M. M. Zavlanos, "On the sublinear regret of distributed primal-dual algorithms for online constrained optimization," *arXiv preprint:1705.11128*, May 2017.

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

[176] S. Vlaski and A. H. Sayed, "Distributed learning in non-convex environments–part II: Polynomial escape from saddle-points," *arXiv preprint:1907.01849*, 2019.

[177] K. Yuan, S. A. Alghunaim, B. Ying, and A. H. Sayed, "On the performance of exact diffusion over adaptive networks," in *Proc. IEEE CDC*, Nice, France, Dec. 2019, pp. 1–1, available on arXiv:1903.10956v1.