DISTRIBUTED SOLUTIONS TO COUPLED CONVEX FEASIBILITY AND OPTIMIZATION PROBLEMS ON AGENT NETWORKS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Yingying Xiao

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Jianghai Hu, Chair

School of Electrical and Computer Engineering

Dr. Shreyas Sundaram

School of Electrical and Computer Engineering

Dr. Dengfeng Sun

School of Aeronautics and Astronautics

Dr. Shaoshuai Mou

School of Aeronautics and Astronautics

Approved by:

Dr. Dimitrios Peroulis

Head of the School Graduate Program

Dedicated to my husband, my foster and biological parents.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my Ph.D. advisor, Professor Jianghai Hu. He motivated me to work on the multi-agent network problems, and more importantly, provided constant inspiration along the journey to explore the mystery behind it. He always maintains the highest availability for his students and I was rewarded by numerous helpful ideas from the discussion with him. His research talent as well as his gentle personality makes the five-year experience very much enjoyable.

I also would like to thank Professors Shreyas Sundaram, Dengfeng Sun, and Shaoshuai Mou for serving on my dissertation committee and providing insightful advices on my work. Their courses about large-scale network and convex optimization laid a solid foundation to my research. Besides, I collaborated closely with Professor Ji Liu from Stony Brook University to develop the algorithms for solving convex feasibility problems and he made significant contributions to establish the proofs. I'm also thankful to Professors James Braun, Panagiota Karava from Ray W. Herrick Laboratories, and Jie Cai from University of Oklahoma for helping me apply the distributed algorithms to the coordinated control problems in smart buildings.

Also I want to express thanks to my colleagues, Dr. Xiaodong Hou and Dr. Donghwan Lee, for their inspirational discussions on research and enjoyable collaborations on courses. Many thanks also go to my friends from ECE department and Ray W. Herrick Laboratories for all the happy moments we had together.

Last but not least, thanks go to my parents for raising me up with unconditional love and supporting me to chase my dreams. I also thank my husband Luyao for always being there to share every moment of my life.

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

Figure Page

SYMBOLS

$\mathbb{R}$	set of real numbers
$\overline{\mathbb{R}}$	$\mathbb{R} \cup \{+\infty\}$
$\mathbb{R}_+$	$[0, +\infty)$
$\mathcal{I}_m$	index set $\{1, 2, \ldots, m\}$ for any positive integer m
$(x_i)_{i \in \mathcal{I}_m}$	column stack of x_i's, $i \in \mathcal{I}_m$, with indices in ascending order
$\mathbf{1}$	column vector with all 1 entries of proper dimension
Id/I	identity matrix/operator
e	unit vector with only one entry to be 1 in the proper position and all other entries zero
$\mathrm{diag}(A_1, \ldots, A_m)$	block diagonal matrix
$[A]_{ij}$	entry in the i-th row and j-th column of matrix A
$\lvert\mathcal{A}\rvert$	cardinality of set A
$d_{\mathcal{A}}(x)$	Euclidean distance of x to the set $\mathcal{A}$
$\mathcal{A}^{\mathrm{o}}$	set of the interior points of set $\mathcal{A}$
$\imath_{\mathcal{A}}(x)$	indicator function of set $\mathcal{A}$, i.e., $\imath_{\mathcal{A}}(x) = 0$ if $x \in \mathcal{A}$ and $\imath_{\mathcal{A}}(x) = +\infty$ if otherwise
$\lvert\!\lvert\!\lvert\cdot\rvert\!\rvert\!\rvert$	general norm when the operand is vector, general induced matrix norm when the operand is matrix
$\lVert\cdot\rVert_1$	l_1-norm or corresponding induced matrix norm
$\lVert\cdot\rVert$	l_2-norm or corresponding induced matrix norm
$\lVert\cdot\rVert_P$	P-norm defined by $\lVert x\rVert_P = \sqrt{x^\top P x}$ or corresponding induced matrix norm
$Q \circ P(x)$	composition $Q(P(x))$ for operators P and Q
$\mathrm{Fix}\, P$	set of fixed points of the operator P

$\operatorname{zer} P$	set of zeros of the operator P
$>,\ <,\ \geq,\ \leq$	entry-wise comparison when the operands are vectors
$\succ,\ \prec,\ \succcurlyeq,\ \preccurlyeq$	$A \succ B$ for matrices A and B means the matrix difference $A - B$ is positive definite; define others similarly

ABBREVIATIONS

CFP	convex feasibility problem
ADMM	alternating direction method of multipliers
AOA	angle of arrival
CPP	closed, convex, and proper

ABSTRACT

Xiao, Yingying PhD, Purdue University, December 2019. Distributed Solutions to Coupled Convex Feasibility and Optimization Problems on Agent Networks. Major Professor: Jianghai Hu.

This thesis studies the distributed solutions to the coupled convex feasibility and optimization problems on agent networks, with the aim of reducing the storage and communication requirements for individual agents by exploiting the potential coupling sparsity across agents. We first focus on the convex feasibility problems. Four iterative solutions are proposed, where each agent only maintains its own variable together with its desired values for those neighboring agents whose valuations help determining its feasibility; within each iteration, projection and consensus operations are carried out by agents in parallel based on information from only the relevant neighbors. Then the approach is extended to solve convex optimization problems for a group of agents whose constraints as well as objective functions may depend on neighbors' variables. Similar solution algorithms are developed by replacing the projection with the proximal operator. Finally, we consider the optimization problems subject to global constraints that involve every agent on the network in addition to the local couplings. Distributed algorithms (with or without a coordinator) following the same approach are proposed. Convergence analysis and numerical examples are provided to demonstrate the effectiveness of the proposed algorithms.

1. INTRODUCTION

1.1 Background and Motivations

Multi-agent systems with the agents on a network cooperating (actively or passively) to achieve a common goal arise in many applications, especially for tasks that are impossible for a single agent to complete due to its limited power, computation ability, sensing range, etc. Examples include mobile sensing networks [1–6], robot teams [7–11], microsatellite formation in space missions [12, 13], opinion in social networks [14–16], electricity grids [17–19], data centers [20, 21], to name a few. Besides the above areas where the multi-agent formulation naturally arises, some applications can be easily transformed to multi-agent systems due to the need to tackle high computation complexity, such as image recovery [22–24], model predictive control [25, 26] and so on.

Depending on the common goal of multi-agent systems, many problems on agent-networks can be formulated as a convex optimization problem, or its special case, a convex feasibility problem (CFP), also called the convex intersection problem or constrained consensus problem [27, 28], which is to find a common point that belongs to a family of nonempty closed convex sets.

There has been a tremendous amount of existing literature on the solutions of networked optimization problems (or CFP). A majority of existing approaches, especially the earlier ones, are centralized in that a central solver updates a guess of the solution iteratively to approach an optimal point (or satisfy the convex constraints in the case of CFP). A particular popular class of such approaches is the alternative projection method and its variants (e.g. [29–31]). Centralized solution algorithms have the advantages of easy implementation and guaranteed convergence. On the other hand, they often scale poorly as the network's size increases.

For problems on large-scale networks, a natural idea is to distribute the computation among agents, where each agent completes a (relatively) small task and exchanges information with its neighbors (possibly multi-hop away) to cooperatively achieve the common goal. This leads to the distributed approaches on agent networks. In many applications, the distributed solutions are naturally desired since the aggregation of information at a single place is almost impossible, e.g., the machine learning problems involving massive data from multiple data centers. Furthermore, in many cases, the constraints and objective functions relevant to an agent often involve the private information of the agent and its neighbors. Without the need to pass these information to the central solver, distributed solutions can better maintain privacy.

A class of distributed optimization problems that receives the most attention is the consensus optimization problem, where a group of m agents with local variables $x_1, \ldots, x_m$ tries to minimize the (separable) objective function $f_1(x_1) + \cdots + f_m(x_m)$ while simultaneously achieving consensus $x_1 = \cdots = x_m$ through local information exchanges. Representative algorithms developed for their solutions include, e.g., subgradient-based methods [27, 32], proximal gradient descent methods [34], dual-decomposition methods [33], and general first order algorithms [35–37], to name a few. Many of the methods developed for consensus optimization problems, e.g., subgradient-based methods, require variable step sizes to be carefully tuned to achieve the best convergence, which is not practical in many applications. For CFP, some noteworthy effort toward this direction includes the distributed algorithms to solve linear equations proposed in [38–43], which are subsequently extended to solve nonlinear equations [44–46], e.g., paracontractions and strongly quasi-nonexpansive maps, and the projected consensus algorithms for constrained consensus problems [27, 28, 47–50] and approximate projections [51]. Some of the earlier works on general CFP along this direction can be found in [29, 52–54].

In these distributed algorithms, each agent maintains a local guess of the solution. These local guesses are updated by individual agents to optimize their local objective

functions as well as satisfy local constraints, and are exchanged between neighboring agents to reach consensus via averaging.

On the other hand, some practical distributed optimization problems arising in agent networks, e.g., the formation control problem [55] and the network localization problem [10], often have locally coupled objective functions and constraints for individual agents. In theory these problems can still be formulated as a consensus optimization problem with each agent maintaining copies of all other agents' variables. However, doing so may result in excessive memory and communication overhead, especially when the number of agents is large. For example, Fig. 1.1 shows a planar network localization problem using relative orientation measurements, which consists of two anchors with known locations and 28 free agents whose locations need to be identified. Using the existing algorithms above, each agent will need to maintain and at each round broadcast 56 variables. However, take the agent labeled by 0 as an instance. Its localization constraint only relates to the five neighbors within its measurement range, i.e., only 12 out of the 56 variables are relevant to the location feasibility of the agent 0. With a larger problem size, this disadvantage of the existing algorithms becomes even more serious.

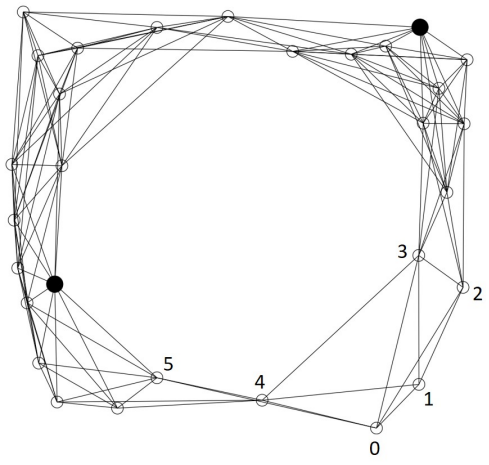


Fig. 1.1. Network localization with 2 anchors (solid dots) and 28 free agents (small circles).

One way to mitigate this issue is to partition not only the objective functions or constraints but also the solution vector into different parts and assign them to individual agents. Along this direction, one approach [56] to solve linear algebraic equations is that each agent still holds a copy of the solution but partitions it into multiple blocks and then broadcasts periodically or randomly only one of them to its neighbors. This approach reduces the communication load at the sacrifice of convergence rate that heavily depends on how frequently the local copies are broadcast. Another method toward this direction for solving linear equations $Ax = b$ is proposed in [57] , which also intends to exploit the sparsity of matrix A. Given the partition of solution x, each agent keeps and broadcasts only the blocks relevant to its own constraints, which often has a much reduced dimension than x. However, in this proposed algorithm, the mappings from the indices of local blocks held by one agent into the index of the global variable x are required to be known not only to this agent but also to its neighbors whose constraints involve all or some of these local blocks. That could lead to a large setup overload. This algorithm is also not applicable to those applications where these mappings are private information that agents do not want to share.

1.2 Preview of Main Contributions

Motivated by the discussions in previous section, we study the convex feasibility and optimization problems on agent networks with local and/or global couplings. The local couplings across agents are modeled by a directed *dependency graph* while the global couplings involving every agent on networks will be dealt with by a coordinator or implicitly by a consensus graph. Our goal is to develop distributed solution algorithms which are able to reduce the storage and communication requirements for individual agents by exploiting the potential sparsity of local couplings. At the same time, the algorithms are expected to have the following properties: constant step size, general convex objectives and constraints, general dependency graph, guaran-

teed convergence to optimality, capability of dealing with asynchrony due to network uncertainty and inhomogeneity, and ability to preserve privacy when necessary. Towards this goal, the focuses of three main chapters are listed in Fig. 1.2 and Chapter 2 summarizes the techniques to be utilized for establishing the convergence proofs of the proposed algorithms.

	Feasibility problems	Optimization problems
Local couplings	Chapter 3	Chapter 4
Local + Global couplings		Chapter 5

Fig. 1.2. Main chapters' organization

Chapter 3 focuses on the convex feasibility problems with locally coupled constraints. In the proposed algorithms, each agent only maintains its own variable together with its desired values for those neighboring agents whose valuations help determining its feasibility; within each iteration, projection and consensus operations are carried out by agents in parallel based on information from only the relevant neighbors. Such an approach significantly reduces the amount of storage and communication required for individual agents in the case of sparse local couplings. Four algorithms in this framework are proposed. Algorithm 1 is synchronous in the way that at each round one of the two operations, projection and consensus, must be completed by all agents before carrying out the other one, which converges exponentially fast under some further assumptions. Algorithm 2 extends Algorithm 1 to be asynchronous by allowing agents to independently choose their operations in each iteration. The convergence proofs of Algorithms 1 and 2 are established on the basis of paracontractions. Algorithm 3 generalizes Algorithm 1 by utilizing general, time-varying consensus operations and allowing individual agents to decide if they would like to perform the projection at each round. Algorithm 4, the most general version, combines the relaxations of Algorithms 2 and 3. Algorithms 3 and 4 accommodate the practical situation of temporary communication blackout and accelerate

the consensus process by weighting the desired values from different neighbors, whose convergences, however, are established by utilizing the strongly quasi-non-expansive maps since the weighted consensus operation is not paracontractions.

Chapter 4 studies the locally coupled optimization problems on agent networks where the local constraints as well as objective functions may depend on neighbors' variables. By adopting the same concept of desired values for neighbors' variables in Chapter 3, the overall constrained optimization problem can be transformed to finding a point $\mathbf{x}^* \in \operatorname{zer}(T_1 + T_2)$ with T_1 and T_2 being two maximal monotone operators. Then the Douglas-Rachford operator splitting method [58,59] is utilized to derive the synchronous distributed algorithm (Algorithm 5), as well as its randomized version (Algorithm 8) via the random coordinate descent method [60–62]. When the same idea is applied to the corresponding dual problem, Algorithms 6 and 9 are obtained. Algorithm 7 shows the distributed solution of applying ADMM. At any round of our proposed algorithms, an agent communicates only with those neighbors whose variables affect its objective and constraints as in CFP, and expensive operations (e.g., solving local optimization problems) are carried out only once by each agent with the rest being simple linear vector operations.

Asynchronous algorithms have been proposed for solving the special class of consensus optimization problem [60, 63, 64], with the algorithm in [60] requiring the activation of at least two agents in each round and the algorithms in [62,65] requiring the knowledge of activation probabilities. The AD-ADMM algorithm in [66] is applicable only to the star topology (albeit with possible network delays). Our proposed asynchronous algorithms in Chapter 4 have no such limitations.

In Chapter 5, we consider the optimization problems subject to global constraints that involve every agent on the network in addition to the local couplings. By introducing at each agent a local copy of the dual variable corresponding to the global constraints, the problem can be decomposed in the same way as that in Chapters 3 and 4 with the addition of the consensus requirement on dual variables. When the consensus task is dealt with by an extra agent (coordinator), similar approach (Al-

gorithm (5.6)) to that of Chapter 4 is developed, which, however, suffers the vulnerability to single-point failures. Instead, when the task is distributed among agents through a consensus graph (which can be constructed from the dependency graph by adding edges between strongly connected groups if such groups exist), a fully distributed solution (Algorithm 10) is proposed based on the preconditioned Douglas-Rachford splitting in [67], which preserves the properties of efficient communication, privacy protection, constant step size, etc., as the other algorithms in Chapters 3 and 4.

Chapter 6 shows the simulation results of applying Algorithms 1, 3, 5 and 8 to solving the linear programs/equations and the network localization problems as well as the comparison with ADMM and the well known projection consensus based algorithm in [28].

2. PRELIMINARIES

In this chapter, we will introduce some techniques that will be used later to establish the convergence proofs and analyze the properties of the algorithms proposed in Chapters 3, 4 and 5.

2.1 Maximal Monotone Operators

An (set-valued) **operator** T on $\mathbb{R}^n$, also called a multi-valued function or relation, is defined to be a subset of $\mathbb{R}^n \times \mathbb{R}^n$: $\{(x, y) \,|\, y \in T(x)\}$. When $T(x)$ is singleton for each x, the operator T becomes a (single-valued) function. A well-known example is the subdifferential operator ∂f of a function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ defined as $\partial f = \{(x, g) \,|\, x \in \mathbf{dom}\, f, f(y) \geq f(x) + g^\top (y - x), \forall y \in \mathbf{dom}\, f\}$. Some useful notions for operators are listed as follows.

a. $\mathbf{dom}\, T = \{x \,|\, T(x) \neq \emptyset\}$.

b. For two operators T and S on $\mathbb{R}^n$, their **composition** $T \circ S = \{(x, z) \,|\, \exists y \in \mathbb{R}^n,\ (x, y) \in S,\ (y, z) \in T\}$ and **sum** $T+S = \{(x, y+z) \,|\, (x, y) \in S,\ (y, z) \in T\}$. Other simple operations such as scalar multiplication are defined similarly to functions.

c. The **inverse** of T is $T^{-1} = \{(y, x) \,|\, (x, y) \in T\}$, which always exists. Note that in general $R^{-1} \circ R \neq I$ with I being the identity operator.

d. The **zero set** of T is $\operatorname{zer} T = \{x \,|\, 0 \in T(x)\}$, which equals to $T^{-1}(0)$.

e. The set of **fixed points** of T is $\operatorname{Fix} T = \{x \,|\, x \in T(x)\}$.

Definition 2.1.1 *An operator T is **monotone** if it satisfies*

$$(q-p)^\top (y-x) \geq 0,\ \forall p \in T(x),\ \forall q \in T(y),\ \forall x,\ y.$$

The monotone operator is the generalization of monotonicity for functions.

Definition 2.1.2 *An operator T is **maximal monotone** if T is monotone and its graph is not properly contained in the graph of another monotone operator.*

In other words, for a maximal monotone operator T, there exists no $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$ such that the operator $\{(x, y)\} \cup T$ is monotone.

The subdifferential operator ∂f defined above is monotone for any $f : \mathbb{R}^n \to \overline{\mathbb{R}}$, and is maximal monotone when f is closed convex proper (CCP), i.e., f is lower semi-continuous, convex and $f(x) \not\equiv +\infty$. Later in this section, we will define the resolvent and the Cayley operator associated with each monotone operator which, when applied to the subdifferential operator, are very useful in deriving the distributed algorithms on networks.

Following are two instances of maximal monotone operators.

a. An **affine** function $f(x) = Ax+b$ is maximal monotone if and only if the matrix A satisfies $A + A^\top \succcurlyeq 0$.

b. For a set $\mathcal{A} \subset \mathbb{R}^n$, its **indicator function** $\imath_{\mathcal{A}}$ is defined as

$$\imath_{\mathcal{A}}(x) = \begin{cases} 0, & \text{if } x \in \mathcal{A}, \\ +\infty, & \text{otherwise.} \end{cases}$$

When the set $\mathcal{A}$ is closed convex, its subdifferential $\partial \imath_{\mathcal{A}}$ can be obtained through the **normal cone operator** $N_{\mathcal{A}}$ defined as follows

$$\partial \imath_{\mathcal{A}}(x) = N_{\mathcal{A}}(x) := \begin{cases} \{q \,|\, q^\top (y-x) \leq 0,\ \forall y \in \mathcal{A}\}, & \text{if } x \in \mathcal{A}, \\ \emptyset, & \text{otherwise.} \end{cases}$$

Obviously, if $x \in \mathbf{int}\mathcal{A}$ (an interior point of the set $\mathcal{A}$), then we have $N_{\mathcal{A}}(x) = \{0\}$. When $\mathcal{A}$ is additionally nonempty, i.e., $\imath_{\mathcal{A}}$ is a CCP function, the operator $N_{\mathcal{A}}$ is maximal monotone. As we will see later, the resolvent of normal cone operator is the projection operator $\Pi_{\mathcal{A}}(x)$ defined later in (2.2), i.e., $(I + \rho N_{\mathcal{A}})^{-1} = \Pi_{\mathcal{A}}(x)$ for some constant $\rho > 0$.

c. The saddle subdifferential operator to be defined later in Section 2.5 is maximal monotone if its saddle function is closed and proper (see Theorem 2.5.1).

2.2 Non-Expansive Mappings

If an operator T satisfies

$$\|q - p\| \leq L\|y - x\|, \ \forall p \in T(x), \ \forall q \in T(y), \ \forall x, \ y,$$

for some constant $L \geq 0$, L is called a Lipschitz constant of T. Obviously, $p = q$ whenever $y = x$, implying that such an operator T degrades to a single-valued function.

Definition 2.2.1 *For any $x, y \in \mathbf{dom}\, T$, the operator T with Lipschitz constant L is*

a. ***non-expansive*** *if $L \leq 1$, or equivalently $\|T(y) - T(x)\| \leq \|y - x\|$;*

b. ***contractive*** *if $L < 1$.*

According to the definitions above, contractive implies non-expansive but not vice versa. For two operators T and $\tilde{T}$ with Lipschitz constants L and $\tilde{L}$ respectively, some properties of function operations on T and $\tilde{T}$ are listed as follows:

a. The **composition** $T \circ \tilde{T}$ has a Lipschitz constant $L\tilde{L}$. This implies that the composition operation preserves the non-expansive (resp. contractive) property when its two operands are non-expansive (resp. contractive).

b. The **weighted average** $(1-\alpha)T + \alpha\tilde{T}$ with $\alpha \in [0,1]$ has a Lipschitz constant $(1-\alpha)L + \alpha\tilde{L}$, thus preserving the non-expansive/contractive property. In addition, if one of T and $\tilde{T}$ is contractive and $\alpha \in (0,1)$, the weighted average is contractive.

For a contractive operator T, there are two important properties below:

a. its fixed point is unique if there exists one, i.e., $\operatorname{Fix} T$ is singleton,

b. the iteration $x^{t+1} = T(x^t)$ will converge to the unique fixed point as $t \to \infty$.

For a non-expansive mapping T, the set of its fixed points $\operatorname{Fix} T$ can be shown to be closed and convex (see [59] for proof details), however, its iteration $x^{t+1} = T(x^t)$ in general does not converge even though $\operatorname{Fix} T$ is nonempty. This issue can be avoided by adopting the following notion of averaged operators (mappings).

Definition 2.2.2 *An mapping T is α-**averaged** if $T = (1-\alpha)I + \alpha S$, for $\alpha \in (0,1)$ and some non-expansive function S.*

The fact that an averaged operator is non-expansive directly follows from the property of weighted average operation above. The two useful results about averaged operators (or mappings) are listed below without proof:

a. The set of fixed points $\operatorname{Fix} T = \operatorname{Fix} S$.

b. The iteration $x^{t+1} = T(x^t)$ will converge to a point $x^* \in \operatorname{Fix} T$ as $t \to \infty$ if $\operatorname{Fix} T$ is nonempty.

These conclusions will be used heavily later on to establish the convergence of the proposed algorithms. An equivalent characterization of α-averaged function [68] is

$$\|T(y) - T(x)\|^2 \leq \|y - x\|^2 - \frac{1-\alpha}{\alpha}\|(T(y) - y) - (T(x) - x)\|^2, \ \forall x, y, \ \alpha \in (0,1). \tag{2.1}$$

As shown in [68], averaged operators are closed under relaxations, convex combinations, and compositions.

Theorem 2.2.1 ([68] Lemma 2.2) *For $i \in \mathcal{I}_m$, let $\alpha_i \in (0,1)$, T_i be an α_i-averaged operator and $w_i \in (0,1]$ such that $\sum_{i\in\mathcal{I}_m} w_i = 1$. Then the following holds:*

a. $\forall i \in \mathcal{I}_m$, $\forall \lambda \in (0, 1/\alpha_i)$, the relaxation $I+\lambda(T_i-I)$ is a $(\lambda\alpha_i)$-averaged operator;

b. the convex combination $\sum_{i\in\mathcal{I}_m} w_i T_i$ is α-averaged, with $\alpha := \max_{i\in\mathcal{I}_m} \alpha_i$;

c. $T_1 \circ \cdots \circ T_m$ is α-averaged with $\alpha := \frac{m}{m-1+1/\max_{i\in\mathcal{I}_m}\alpha_i}$.

The following result, a special case of [61, Thm. 3], will be used later in Chapter 4 for the asynchronous design of distributed algorithms.

Theorem 2.2.2 *Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be an α-averaged operator with $\mathrm{Fix}(T) \neq \emptyset$. Partition x into $(x_1, \ldots, x_m)$ and Tx into $(T_1x, \ldots, T_mx)$ where $x_i, T_ix \in \mathbb{R}^{n_i}$ for $i \in \mathcal{I}_m$. Consider the following iteration. At each step $t = 0, 1, \ldots$, first an index $i^t \in \mathcal{I}_m$ is chosen randomly and independently with the probabilities $\mathbb{P}(i^t = j) = p_j \geq \varepsilon$, $j \in \mathcal{I}_m$, for some positive ε; then x^t is updated to x^{t+1} where $x_{i^t}^{t+1} = T_{i^t}x^t$ and $x_\ell^{t+1} = x_\ell^t$ for $\ell \neq i^t$. Then, x^t converges almost surely to some $x^* \in \mathrm{Fix}(T)$ as $t \to \infty$.*

Here are two examples of non-expansive mappings.

Example 2.2.1 (Differentiable functions) *For a differentiable function $T : \mathbb{R}^n \to \mathbb{R}^n$, its Lipschitz constant is $L = \sup_x \|DT(x)\|$ with $DT(x)$ being the Jacobian matrix of T at point x and $\|\cdot\|$ being the l_2 induced matrix norm defined as $\|A\| := \sup_{x\in\mathbb{R}^n, x\neq 0} \|Ax\|/\|x\|$ for $A \in \mathbb{R}^{n\times n}$. The function T is non-expansive and contractive when $L \leq 1$ and $L < 1$, respectively.*

Example 2.2.2 (Projection) *For a point $x \in \mathbb{R}^n$, denote by $\Pi_{\mathcal{F}}(x)$ the (unique) orthogonal projection of x onto the nonempty closed convex set $\mathcal{F} \subset \mathbb{R}^n$, i.e.*

$$\Pi_{\mathcal{F}}(x) = \mathrm{argmin}_{y\in\mathcal{F}} \|y - x\|, \tag{2.2}$$

with $\|\cdot\|$ being the l_2-norm. The orthogonal projection operator $\Pi_{\mathcal{F}}$ enjoys the following properties with proof omitted: $\forall x, y \in \mathbb{R}^n$,

a. $x \in \mathcal{F} \iff x = \Pi_{\mathcal{F}}(x)$, *i.e.,* $\mathcal{F} = \operatorname{Fix} \Pi_{\mathcal{F}}$;

b. $\langle x - \Pi_{\mathcal{F}}(x), y - \Pi_{\mathcal{F}}(x) \rangle \leq 0$;

c. $\|x - \Pi_{\mathcal{F}}(x)\|^2 + \|y - \Pi_{\mathcal{F}}(x)\|^2 \leq \|x - y\|^2$;

d. $\Pi_{\mathcal{F}}$ *is non-expansive (see proof from [59]).*

In the rest of this section, we will introduce several concepts related to non-expansive mappings: firmly non-expansive mappings, paracontractions, strongly quasi-non-expansive maps.

2.2.1 Firmly Non-Expansive Mappings

Definition 2.2.3 *A mapping T is* ***firmly non-expansive*** *if* $\forall x, y \in \mathbf{dom}\, T$,

$$\|T(y) - T(x)\|^2 \leq \langle y - x, T(y) - T(x) \rangle, \tag{2.3}$$

or equivalently,

$$\|T(y) - T(x)\|^2 \leq \|y - x\|^2 - \|(T(y) - y) - (T(x) - x)\|^2. \tag{2.4}$$

By definition, firmly non-expansive implies non-expansive but not vice versa. Comparing (2.4) with the equivalent form (2.1) of the α-averaged function, firmly non-expansive mappings are special cases of averaged operators with $\alpha = 1/2$. An interesting result about the relationship between firmly non-expansive mappings and monotone operators from [58] is stated below.

Theorem 2.2.3 ([58] Propositions 23.8 and 23.9) *Let $T : \mathbb{R}^n \to \mathbb{R}^n$. Then T is firmly non-expansive if and only if there is a maximal monotone operator A : such that T is the resolvent (to be defined later in Section 2.3) of A, i.e., $T = (I + A)^{-1}$.*

2.2.2 Paracontractions

Definition 2.2.4 ([69]) *A continuous mapping $T : \mathbb{R}^n \to \mathbb{R}^n$ is called a **paracontraction** w.r.t. a norm $|||\cdot|||$ on $\mathbb{R}^n$ if $|||T(x) - y||| < |||x - y|||$ for any $x \notin \mathrm{Fix}T$ and $y \in \mathrm{Fix}\, T$.*

As shown in [69–73], many well-known operators are paracontracions. Here are several important examples of paracontractions w.r.t. the l_2-norm:

a. the orthogonal projection $\Pi_{\mathcal{F}}$ onto a nonempty closed convex set $\mathcal{F}$;

b. for a nonempty closed convex set $\mathcal{F}$, the α-relaxed projection T onto $\mathcal{F}$ given by

$$T = (1 - \alpha) \cdot \mathrm{Id} + \alpha \cdot \Pi_{\mathcal{F}},\ \alpha \in (0, 2); \tag{2.5}$$

c. the gradient descent map $g(x) = x - \alpha \nabla f(x)$, where $f(x)$ is a convex and differentiable function, its gradient ∇f is Lipschitz continuous with the constant L and $\alpha \in (0, 2/L)$;

d. the proximal operator $\mathbf{prox}_f : \mathbb{R}^n \to \mathbb{R}^n$ associated with the CCP function $f : \mathbb{R}^n \to \overline{\mathbb{R}}^n$ to be defined later on in (2.6);

e. the α-averaged operator $T = (1-\alpha)I+\alpha S$ when the operator S is non-expansive.

A key result proved in [69] is restated in the following theorem, which will be the fundamental tool to establish the convergence of the algorithms in Chapter 3.

Theorem 2.2.4 *Suppose $T_i : \mathbb{R}^n \to \mathbb{R}^n$, $i = 1, \ldots, \ell$, are paracontractions w.r.t. the same norm $|||\cdot|||$ and $\cap_{i=1}^{\ell} \mathrm{Fix}\, T_i \neq \emptyset$. Starting from any $x^0 \in \mathbb{R}^n$ and for any sequence $\sigma^0, \sigma^1, \ldots \in \{1, \ldots, \ell\}$ so that each index i appears infinitely often, the iteration*

$$x^{t+1} = T_{\sigma^t}(x^t), \quad \forall t = 0, 1, \ldots,$$

will converge to a point $x^* = \lim_{t\to\infty} x^t \in \cap_{i=1}^{\ell} \mathrm{Fix}\, T_i$.

Now we present two useful results for the convergence proofs in Chapter 3.

Lemma 2.2.1 *Suppose $T_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$ is a paracontractions w.r.t. the norm $\|\|\cdot\|\|_i$ for $i = 1, \ldots, m$. Then $T = T_1 \times \cdots \times T_m : \mathbb{R}^n \to \mathbb{R}^n$ where $n = \sum_{i=1}^m n_i$ is a paracontraction w.r.t. the norm $\|\|x\|\| := (\sum_{i=1}^m \|\|x_i\|\|_i^p)^{1/p}$ for $x = (x_1, \ldots, x_m) \in \mathbb{R}^n$ and $p \geq 1$.*

Proof For arbitrary $x, y \in \mathbb{R}^n$, suppose y is a fixed point of T, i.e., $T_i(y_i) = y_i$ for all i; while x is not. As a result, for i in a nonempty subset of $\{1, \ldots, m\}$ we have $T_i(x_i) \neq x_i$ and, therefore $\|\|T_i(x_i) - y_i\|\|_i < \|\|x_i - y_i\|\|_i$; while for all other i, equality holds. This implies that $\|\|T(x) - y\|\| < \|\|x - y\|\|$. ■

Lemma 2.2.2 *Suppose $\tilde{T} : \mathbb{R}^n \to \mathbb{R}^n$ is a paracontractions w.r.t. the l_2-, l_1-norm or any norm that is invariant under permutations, denoted by $\|\cdot\|$, and $\tilde{T}$ has the fixed point set $\tilde{\mathcal{A}}$. Then the operator $T : \mathbb{R}^n \to \mathbb{R}^n$ defined by $T = P^\top \tilde{T} P$, where P is a permutation matrix, is still a paracontraction w.r.t. the same norm with the fixed point set $\mathcal{A} = \{P^\top \tilde{y} | \tilde{y} \in \tilde{\mathcal{A}}\}$.*

Proof Let $y \in \mathbb{R}^n$ be a fixed point of T. Then by definition there exist $\tilde{y} \in \tilde{\mathcal{A}}$ such that $y = P^\top \tilde{y}$. Therefore for any $x \in \mathbb{R}^n$ and $x \notin \mathcal{A}$,

$$\|T(x) - y\| = \|P^\top \tilde{T} P x - P^\top \tilde{y}\| = \|\tilde{T} P x - \tilde{y}\| < \|Px - \tilde{y}\| = \|Px - Py\| = \|x - y\|,$$

where the second and last equalities follow from the property that permutation does not change the norm of a vector. This completes the proof. ■

2.2.3 Stongly Quasi-Non-Expansive Maps

Definition 2.2.5 ([74]) *Let $\beta > 0$. A mapping $P : \mathbb{R}^n \to \mathbb{R}^n$ is β-**strongly quasi-non-expansive** w.r.t. a norm $\|\|\cdot\|\|$ on $\mathbb{R}^n$ if*

$$\|\|P(x) - y\|\|^2 \leq \|\|x - y\|\|^2 - \beta \|\|P(x) - x\|\|^2$$

for any $x \in \mathbb{R}^n$ and $y \in \mathrm{Fix}P$.

By the definitions above, a β-strongly quasi-non-expansive map must be a paracontraction. In other words, the class of β-strongly quasi-non-expansive maps is a subclass of paracontractions. However, the converse is not true.

Lemma 2.2.3 *For a nonempty closed convex set $\mathcal{F}$ and constant $\alpha \in (0, 2)$, the α-relaxed projection T defined in* (2.5) *and $\Pi_{\mathcal{F}}$ have the same set of fixed points.*

Proof Suppose x is a fixed point of $\Pi_{\mathcal{F}}$, i.e. $x = \Pi_{\mathcal{F}}(x)$. It follows that

$$T(x) = (1-\alpha)x + \alpha\Pi_{\mathcal{F}}(x) = (1-\alpha)x + \alpha x = x,$$

i.e. x is also a fixed point of T.

On the other hand, suppose x is a fixed point of T, i.e.

$$x = T(x) = (1-\alpha)x + \alpha\Pi_{\mathcal{F}}(x)$$

which directly leads to $x = \Pi_{\mathcal{F}}(x)$. ■

Proposition 2.2.1 ([74]) [1] *For a nonempty closed convex set $\mathcal{F}$ and $\alpha \in (0, 2)$, the α-relaxation map T defined in (2.5) is $\frac{2-\alpha}{\alpha}$-strongly quasi-non-expansive.*

Proof Let $x \in \mathbb{R}^n$ be arbitrary and $y \in \mathbb{R}^n$ be a fixed point of T. Then,

$$\begin{aligned}
\|T(x) - y\|^2 &= \|(1-\alpha)x + \alpha\Pi_{\mathcal{F}}(x) - y\|^2 \\
&= \|x - y\|^2 + \alpha^2\|\Pi_{\mathcal{F}}(x) - x\|^2 + 2\alpha\langle x - y, \Pi_{\mathcal{F}}(x) - x\rangle \\
&= \|x - y\|^2 - \frac{2-\alpha}{\alpha}\|T(x) - x\|^2 \\
&\quad + \frac{2-\alpha}{\alpha}\|T(x) - x\|^2 + \alpha^2\|\Pi_{\mathcal{F}}(x) - x\|^2 + 2\alpha\langle x - y, \Pi_{\mathcal{F}}(x) - x\rangle
\end{aligned}$$

[1] This proof is based on private communication with Professor Ji Liu from Stony Brook University.

By plugging the definition of $T(x)$ in (2.5), into the last line and combining similar terms, we have

$$\begin{aligned}
&\|T(x) - y\|^2 \\
&\quad = \|x - y\|^2 - \frac{2-\alpha}{\alpha}\|T(x) - x\|^2 + 2\alpha\|\Pi_{\mathcal{F}}(x) - x\|^2 + 2\alpha\langle x - y, \Pi_{\mathcal{F}}(x) - x\rangle \\
&\quad = \|x - y\|^2 - \frac{2-\alpha}{\alpha}\|T(x) - x\|^2 + 2\alpha\langle \Pi_{\mathcal{F}}(x) - y, \Pi_{\mathcal{F}}(x) - x\rangle \\
&\quad \leq \|x - y\|^2 - \frac{2-\alpha}{\alpha}\|T(x) - x\|^2,
\end{aligned}$$

where the last inequality follows from the property (c.) of the orthogonal projection. ■

2.3 Resolvent and Cayley Operators

In this section, we introduce the resolvent and reflected resolvent (i.e., Cayley operator) associated with monotone operators.

Definition 2.3.1 *For a monotone operator T and a constant $\rho > 0$, its* ***resolvent*** *$J_{\rho T}$ and* ***reflected resolvent (Cayley operator)*** *$R_{\rho T}$ are*

$$J_{\rho T} = (I + \rho T)^{-1},\ R_{\rho T} = 2J_{\rho T} - I.$$

As shown in Chapter 6 of [59] and Chapter 23 of [58], the resolvent and reflected solvent enjoy the following properties.

Theorem 2.3.1 ([59], [58]) *For an operator T and constant $\rho > 0$, the following holds:*

a. $0 \in T(x)$ if and only if $x = J_{\rho T}(x) = R_{\rho T}(x)$.

b. If T is monotone, $R_{\rho T}$ is non-expansive and thus $J_{\rho T}$ is $1/2$-averaged;

c. If T is maximal monotone, $\mathbf{dom}\, J_{\rho T} = \mathbf{dom}\, R_{\rho T} = \mathbb{R}^n$.

When the function $f(x)$ on $\mathbb{R}^n$ is CCP, its subdifferential $\partial f(x)$ is a maximal monotone operator, whose resolvent turns out to be its **proximal operator** given as follows:

$$J_{\rho\partial f}(x) = \mathrm{prox}_{\rho\partial f}(x) := \operatorname*{argmin}_z \left(f(z) + \frac{1}{2\rho}\|z - x\|^2 \right), \ \forall x \in \mathbf{dom}\, f. \tag{2.6}$$

As a special case, when $f(x) = \imath_{\mathcal{A}}(x)$ for a nonempty convex closed set $\mathcal{A} \in \mathbb{R}^n$, the resolvent of its subdifferential $\partial \imath_{\mathcal{A}}$ (i.e., the normal cone $N_{\mathcal{A}}$) is the projection operator, specifically,

$$J_{\rho\partial \imath_{\mathcal{A}}}(x) = \Pi_{\mathcal{A}}(x), \ \forall x \in \mathbb{R}^n.$$

Note that the evaluation of $J_{\rho\partial \imath_{\mathcal{A}}}(x)$ does not depend on the parameter ρ.

Next we introduce the generalized (or preconditioned) resolvent from [58, 67], whose properties are summarized in the following Theorem 2.3.2, an analogue of Theorem 2.3.1.

Definition 2.3.2 *For a monotone operator T and constant matrix $P \succ 0$, the **generalized resolvent** J_T^P and reflected resolvent R_T^P (preconditioned by P) are defined as*

$$J_T^P = (I + P^{-1}T)^{-1}, \ R_T^P = 2J_T^P - I. \tag{2.7}$$

Theorem 2.3.2 ([58, 67]) *For a monotone operator T and a matrix $P \succ 0$, the following holds:*

a. $0 \in T(x)$ if and only if $x = J_T^P(x) = R_T^P(x)$;

b. If T is monotone, R_T^P is non-expansive and thus J_T^P is $1/2$-averaged function w.r.t. the P-norm defined as $\|x\|_P = \sqrt{x^\top P x}$.

The P-preconditioned resolvent enjoys the following nice property.

Lemma 2.3.1 *Suppose the CCP function $f(x) : \mathbb{R}^n \to \overline{\mathbb{R}}$ is block separable as $f(x) = \sum_{i\in\mathcal{I}_m} f_i(x_i)$, where for each $i \in \mathcal{I}_m$, $f_i(x_i) : \mathbb{R}^{n_i} \to \overline{\mathbb{R}}$ is CCP, and $\sum_{i\in\mathcal{I}_m} n_i = n$. Then for the preconditioned matrix*

$$P = diag\{P_1, \ldots, P_m\},\ P_i \succ 0,\ i \in \mathcal{I}_m,$$

the P-preconditioned resolvent of ∂f can be evaluated as

$$J^P_{\partial f}(x) = \left(J^{P_i}_{\partial f_i}(x_i)\right)_{i\in\mathcal{I}_m}.$$

Proof The P-preconditioned resolvent $x' = J^P_{\partial f}(x)$ is equivalent to $x \in x' + P^{-1}\partial f(x')$. Combined with the fact that $\partial f(x') = \left(\partial f_i(x'_i)\right)_{i\in\mathcal{I}_m}$ resulted from the assumption $f(x) = \sum_{i\in\mathcal{I}_m} f_i(x_i)$, we have $\forall i \in \mathcal{I}_m$, $x_i \in x'_i + P_i^{-1}\partial f(x'_i)$, i.e., $x'_i = J^{P_i}_{\partial f_i}(x_i)$. This completes the proof. ■

Remark 2.3.1 *Here are two special cases of Lemma 2.3.1: (a) when $P = (1/\rho)I_n$, the P-preconditioned resolvent degrades to the canonical case $J_{\rho\partial f}$ and $J_{\rho\partial f}(x) = (J_{\rho\partial f_i}(x_i))_{i\in\mathcal{I}_m}$ (or $\mathbf{prox}_{\rho f}(x) = \left(\mathbf{prox}_{\rho f_i}(x_i)\right)_{i\in\mathcal{I}_m}$); (b) for the indicator function $\imath_C(x) = \sum_{i\in\mathcal{I}_m} \imath_{C_i}(x_i)$ and $P_i = (1/\rho_i)I_{n_i}, \rho_i > 0, \forall i \in \mathcal{I}_m$, we have $J^P_{\partial \imath_C}(x) = \Pi_C(x)$ for any $x \in \mathbb{R}^n$.*

2.4 Douglas-Rachford Splitting

As will be seen later in Chapters 4 and 5, many optimization problems can be eventually formulated as finding a point x such that

$$0 \in T(x)$$

for some maximal monotone operator T, which is equivalent to finding a fixed point of its resolvent $J_{\rho T}$ for some constant $\rho > 0$. Then a solution can be obtained by adopting the well known fixed point iteration

$$x^{t+1} = J_{\rho T}(x^t),$$

when $J_{\rho T}$ is averaged with its fixed point set being the same as the zero set of T. However, as the complexity of the operator T increases, it becomes difficult or even impossible to evaluate its resolvent $J_{\rho T}(x)$ directly.

In many practical problems, the maximal monotone operator T can be written as the sum of (relatively) simpler operators in the following form:

$$T = T_1 + T_2, \quad \text{or} \quad T = T_1 + T_2 + T_3,$$

where the computations of $J_{\rho T_1}, J_{\rho T_2}, J_{\rho T_3}$ are much cheaper than the direct computation of $J_{\rho T}$. In this section, we introduce the two-operator splitting methods which utilize the cheaper $J_{\rho T_1}, J_{\rho T_2}$ to compute $J_{\rho T}$. For the three-operator splitting method, interested readers can refer to [75] for details.

Suppose a maximal monotone operator T admits the splitting $T = T_1 + T_2$ for some maximal monotone operators T_1 and T_2. Then, for $\rho > 0$ and $\alpha \in (0, 1)$, the following holds [76–78]

$$\begin{aligned} 0 \in (T_1 + T_2)(x) &\iff x = J_{\rho T_2}(z),\ z = (2J_{\rho T_1} - I)(2J_{\rho T_2} - I)(z) \\ &\iff x = J_{\rho T_2}(z),\ z = \Big((1-\alpha)I + \alpha(2J_{\rho T_1} - I)(2J_{\rho T_2} - I)\Big)(z), \end{aligned}$$

where the latter equivalence results from the property of α-averaged operators. When the last splitting setup is used for the fixed point iteration, the convergence result is stated in the following Theorem 2.4.1.

Theorem 2.4.1 (Douglas-Rachford splitting) *Let T_1, T_2 be maximal monotone operators and constant scalars $\alpha \in (0, 1)$ and $\rho > 0$. Then starting from any initial points z^0, with the iterations being*

$$x^{t+1} = J_{\rho T_2}(z^t), \tag{2.8a}$$

$$z^{t+1} = z^t + 2\alpha\Big(J_{\rho T_1}(2x^{t+1} - z^t) - x^{t+1}\Big), \tag{2.8b}$$

the sequence $\{x^t\}$ generated by (2.8) will converge to a point $x^* \in \operatorname{zer}(T_1 + T_2)$.

As shown in Proposition 4.2.1 in [67], when $J_{\rho T_1}$ and $J_{\rho T_2}$ are replaced by their generalized resolvents $J^P_{T_1}$ and $J^P_{T_2}$, respectively, the convergence result stays valid, as restated in the following Theorem 2.4.2.

Theorem 2.4.2 (Preconditioned Douglas-Rachford splitting) *Let T_1, T_2 be maximal monotone operators, constant scalar $\alpha \in (0,1)$ and matrix $P \succ 0$. Then starting from any initial points z^0, with the iterations being*

$$x^{t+1} = J^P_{T_2}(z^t), \tag{2.9a}$$

$$z^{t+1} = z^t + 2\alpha\Big(J^P_{T_1}(2x^{t+1} - z^t) - x^{t+1}\Big), \tag{2.9b}$$

the sequence $\{x^t\}$ generated by (2.9) will converge to some $x^* \in \mathrm{zer}(T_1 + T_2)$.

The Douglas-Rachford splitting method and its preconditioned version will be used in Chapters 4 and 5 to derive the distributed solutions to coupled optimization problems on agent networks.

2.5 Saddle Functions and Saddle Points

For the constrained optimization problems to be studied in Chapters 4 and 5, when strong duality holds, the minimizer set has a one-to-one correspondence with the saddle point set of their corresponding Lagrange functions, which are instances of saddle functions. In this section, we review some basic concepts and properties about saddle functions and saddle points.

2.5.1 Saddle Functions

Let $K : X \times Y \to \overline{\mathbb{R}}$ be an extended-real-valued function defined on the product of two Euclidean spaces X and Y.

Definition 2.5.1 ([79]) *K is called a **saddle function** on $X \times Y$ if $K(x,y)$ is a convex function of x for each fixed y and a concave function of y for each fixed x. The saddle function K is* closed *if $K(x,y)$ is lower semicontinuous in x for each fixed y and upper semicontinuous in y for each fixed x. The effective domain of K is defined as*

$$\mathbf{dom}\, K := \{(x,y) \in X \times Y \mid K(x,y') < +\infty,\ \forall y' \in Y \text{ and } K(x',y) > -\infty,\ \forall x' \in X\},$$

and K is called proper *if* $\mathbf{dom}\, K \neq \emptyset$.

Followed are several common instances of saddle functions.

a. $K(x,y) = f(x) - g(y) + x^T Ay$ is a saddle function where $f \in \overline{\mathbb{R}}$ and $g \in \mathbb{R}$ are convex functions and $x^T Ay$ is bilinear for some matrix A. K is closed and proper if both f and g are CCP functions.

b. For the optimization problem $\min_{x\in\mathbb{R}^n}\{f(x) \,|\, g(x) \leq 0\}$ where $f \in \mathbb{R}$ and $g \in \mathbb{R}^m$ are convex functions, the Lagrange function $L(x,y) = f(x) + \langle y, g(x)\rangle - \imath_{\mathbb{R}_+}(y)$ with the set $\mathbb{R}_+ = \{y \,|\, y \geq 0\}$ is a saddle function. It is closed and proper if both f and g are CCP functions.

c. Let $C \subset X$ and $D \subset Y$ be convex subsets. Define

$$\mu_{C\times D}(x,y) := \imath_C(x) - \imath_{(C\times D^c)^c}(x,y) = \begin{cases} 0 & \text{if } x \in C \text{ and } y \in D \\ -\infty & \text{if } x \in C \text{ and } y \notin D \\ +\infty & \text{if } x \notin C, \end{cases} \tag{2.10}$$

where $(C \times D^c)^c = (X \times Y) \setminus (C \times (Y \setminus D))$. Then $\mu_{C\times D}$ is a saddle function on $X \times Y$ with domain $C \times D$. It is closed and proper if both C and D are closed and nonempty.

2.5.2 Saddle Points

Definition 2.5.2 *For the saddle function K, a point $(x^*, y^*) \in X \times Y$ is called a* ***saddle point*** *if*

$$K(x^*, y) \leq K(x^*, y^*) \leq K(x, y^*), \quad \forall x \in X,\, y \in Y. \tag{2.11}$$

In other words, x^* is a minimizer of $K(\cdot, y^*)$ and y^* is a maximizer of $K(x^*, \cdot)$. In this case, we must have $\sup_y \inf_x K(x,y) = \inf_x \sup_y K(x,y) = K(x^*, y^*)$. If K is the payoff function of a zero-sum two-player game, then its saddle points are exactly the Nash equilibria.

General saddle functions may have no saddle points. A sufficient condition for the existence of saddle points based on the Sion's Minimax Theorem [80] is given below.

Proposition 2.5.1 ([80]) *Suppose K is a real-valued closed saddle function on $X \times Y$. Let $C \subset X$ and $D \subset Y$ be two nonempty compact convex subsets. Then $\min_{x\in C} \max_{y\in D} K(x,y) = \max_{y\in D} \min_{x\in C} K(x,y)$. Moreover, (x^*, y^*) with $x^* = \operatorname{argmin}_{x\in C} \max_{y\in D} K(x,y)$ and $y^* = \operatorname{argmax}_{y\in D} \min_{x\in C} K(x,y)$ is a saddle point of $K + \mu_{C\times D}$ with $\mu_{C\times D}$ defined in* (2.10).

2.5.3 Saddle Subdifferential Operators

For the saddle function K on $X \times Y$, a set-valued operator $T_K : X \times Y \to 2^{X\times Y}$ can be defined by

$$T_K(x,y) = \begin{bmatrix} \partial_x K(x,y) \\ \partial_y(-K)(x,y) \end{bmatrix}, \quad \forall (x,y) \in X \times Y. \tag{2.12}$$

Here, $\partial_x K(x,y)$ denotes the subdifferentials (set of subgradients) of the convex function $K(\cdot, y)$ at the point x; similarly for $\partial_y(-K)(x,y)$. In [59], T_K is referred to as the **saddle subdifferential operator** of K. The domain of T_K is $\mathbf{dom}\, T_K := \{(x,y) \,|\, T_K(x,y) \neq \emptyset\}$. The zero set of T_K is $\operatorname{zer}(T_K) := \{(x,y) \,|\, 0 \in T_K(x,y)\}$. The condition (2.11) for a point (x^*, y^*) to be a saddle point of K is equivalent to $0 \in \partial_x K(x^*, y^*)$ and $0 \in \partial_y(-K)(x^*, y^*)$, i.e., $0 \in T_K(x^*, y^*)$. We thus have the following results.

Proposition 2.5.2 ([86]) *The set of saddle points of K is $\operatorname{zer}(T_K)$.*

Theorem 2.5.1 ([79]) *Let K be a saddle function on $X \times Y$. If K is proper, then T_K is a monotone operator with the domain $\mathbf{dom}\, T_K \subset \mathbf{dom}\, K$. If K is proper and closed, then T_K is a maximally monotone operator.*

By the result in Theorem 2.5.1 above, T_K is maximally monotone for a closed proper saddle function K. Its resolvent with $\rho > 0$, denoted by $J_{\rho T_K}$, is an averaged

operator with the fixed point set $\mathrm{Fix}(J_{\rho T_K}) = \mathrm{zer}(T_K)$ being exactly the set of saddle points of K. The iteration $(x^{k+1}, y^{k+1}) = J_{\rho T_K}(x^k, y^k)$, being the iteration of an averaged operator, will converge to a point in $\mathrm{Fix}(J_{\rho T_K})$ and hence a saddle point of K.

Next we characterize how $J_{\rho T_K}$ can be computed. For any $(x, y) \in X \times Y$, $(p, q) = J_{\rho T_K}(x, y)$ if and only if

$$
\begin{aligned}
& (x, y) \in (I + \rho T_K)(p, q) \\
\Leftrightarrow \quad & (x, y) \in (p + \rho \partial_p K(p, q), q + \rho \partial_q (-K)(p, q)) \\
\Leftrightarrow \quad & \begin{cases} 0 \in \partial_p K(p, q) + (p - x)/\rho \\ 0 \in \partial_q (-K)(p, q) + (q - y)/\rho \end{cases} && \text{(2.13a)} \\
\Leftrightarrow \quad & \begin{cases} p = \mathrm{argmin}_{p \in X} \ K(p, q) + \frac{1}{2\rho}\|p - x\|^2 \\ q = \mathrm{argmax}_{q \in Y} \ K(p, q) - \frac{1}{2\rho}\|q - y\|^2 \end{cases} && \text{(2.13b)} \\
\Leftrightarrow \quad & (p, q) \text{ is a saddle point of } K(p, q) + \frac{1}{2\rho}\left(\|p - x\|^2 - \|q - y\|^2\right). && \text{(2.13c)}
\end{aligned}
$$

The above equivalent conditions can be used to compute $J_{\rho T_K}$ for certain families of saddle functions.

a. Suppose $K(x, y) = f(x) + \langle y, Ax - b \rangle$ is the Lagrange function for the optimization problem of minimizing $f(x)$ subject to $Ax = b$. The corresponding T_K is called the KKT operator [59]. In particular, (2.13a) becomes

$$0 \in (p - x)/\rho + \partial f(p) + A^T q, \ \ (q - y)/\rho = Ap - b.$$

The second equation implies $q = y + \rho(Ap - b)$, which when plugged into the first one yields $0 \in \partial f(p) + (p - x)/\rho + A^T y + \rho A^T (Ap - b)$. In other words,

$$\begin{cases} p &= \mathrm{argmin}_z \, \mathcal{L}_\rho(z) + \frac{1}{2\rho}\|z - x\|^2 \\ q &= y + \rho(Ap - b). \end{cases}$$

Here, $\mathcal{L}_\rho(z) := f(z) + \langle y, Az - b \rangle + \frac{\rho}{2}\|Az - b\|^2$ is the augmented Lagrange function.

b. Consider the saddle function

$$K(x,y) = \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^T \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^T & -\Sigma_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}^T \begin{bmatrix} x \\ y \end{bmatrix}$$

where $\Sigma_1, \Sigma_3 \succeq 0$. Then (2.13a) leads to

$$J_{\rho T_K}(x,y) = (I + \rho\Sigma)^{-1} \begin{bmatrix} x - \rho b_1 \\ y + \rho b_2 \end{bmatrix} \tag{2.14}$$

where $\Sigma = \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ -\Sigma_2^T & \Sigma_3 \end{bmatrix}$. If Σ is nonsingular, there is a unique saddle point $\Sigma^{-1}\begin{bmatrix} -b_1^T & b_2^T \end{bmatrix}^T$. In the special case of $\Sigma_1 = 0$ and $\Sigma_3 = 0$, the result becomes:

$$\begin{cases} p = \left(I + \rho^2\Sigma_2\Sigma_2^T\right)^{-1}(x - \rho b_1 - \rho\Sigma_2 y - \rho^2\Sigma_2 b_2) \\ q = y + \rho b_2 + \rho\Sigma_2^T p. \end{cases} \tag{2.15}$$

c. For the saddle function μ in Example 2.5.1 (iii), its resolvent is $J_{\rho T_\mu} = \Pi_C \times \Pi_D$, i.e., $J_{\rho T_\mu}(x,y) = (\Pi_C(x), \Pi_D(y))$.

The following result will be useful later on. Its proof is straightforward and hence omitted.

Proposition 2.5.3 (Separable K) *Suppose $K(x,y) = K_1(x_1,y_1)+\cdots+K_m(x_m,y_m)$ is separable. Here, $x = (x_1,\ldots,x_m) \in X = X_1 \times \cdots \times X_m$, $y = (y_1,\ldots,y_m) \in Y = Y_1 \times \cdots \times Y_m$, and $K_i(x_i,y_i)$ is a closed proper saddle function on $X_i \times Y_i$ for each i. Then, $(p,q) = J_{\rho T_K}(x,y)$ is given by $p = (p_1,\ldots,p_m)$ and $q = (q_1,\ldots,q_m)$ where $(p_i,q_i) = J_{\rho T_{K_i}}(x_i,y_i)$ for each i.*

2.6 Some Useful Notions

For a stochastic matrix $A \in \mathbb{R}^{m\times m}$, its *associated graph* $\mathcal{G}$ is defined to have the vertex set $\mathcal{I}_m$ and a directed edge (j,i) from vertices j to i whenever the entry in i-th row and j-th column is positive, i.e., $[A]_{ij} > 0$.

A finite sequence of graphs $\mathcal{G}_1, \ldots, \mathcal{G}_T$ with the same vertex set $\mathcal{I}_m$ is **jointly strongly connected** if their union $\mathcal{G}_1 \cup \cdots \cup \mathcal{G}_T$ is strongly connected. Here the union $\mathcal{G}_1 \cup \cdots \cup \mathcal{G}_T$ is the directed graph with the same vertex set $\mathcal{I}_m$ and an edge set that is the union of individual graph's edge set. An infinite sequence of graphs $\{\mathcal{G}_t\}$ is **repeatedly jointly strongly connected** if there exists a length $T > 0$ such that every T successive graphs from $\{\mathcal{G}_t\}$ is jointly strongly connected.

A **vector is stochastic** if all entries are nonnegative and sum to one and a **matrix is stochastic** when all of its row vectors are stochastic.

A matrix $A \in \mathbb{R}^{n\times n}$ is an **M-matrix** [81] if: a) each off-diagonal entry is non-positive; and b) A is invertible and and its inverse A^{-1} has no negative entries.

For $n \geq m$, suppose the rows and columns of a square matrix $A \in \mathbb{R}^{n\times n}$ can be partitioned in the same way into $m \times m$ non-empty blocks and denote by A_{ij} the block in the i-th row and j-th column. Then the matrix A is **block diagonally dominant** if

$$\|A_{ii}^{-1}\|^{-1} \geq \sum_{j\in\mathcal{I}_m, j\neq i} \|A_{ij}\|, \ \forall i \in \mathcal{I}_m,$$

and is **block strictly diagonally dominant** if every strict inequality holds in the expression above.

Theorem 2.6.1 *([82] Theorem 9) If a matrix A is block strictly diagonally dominant and each diagonal block of A is an M-matrix, then all eigenvalues of A has positive real part.*

3. CONVEX FEASIBILITY PROBLEMS WITH LOCALLY COUPLED CONSTRAINTS

In this chapter, we consider the convex feasibility problems on agent networks where each agent's constraint depends on its neighbors' variables in addition to its own variable. The dependence relation is depicted by a directed graph. When the couplings across agents are sparse, implying that the dependence graph is (relatively) simple, the algorithms proposed in this chapter could take advantage of this sparsity to significantly reduce the storage and communication amount required for individual agent.

3.1 Problem Formulation

Consider a set of m agents indexed by $\mathcal{I}_m$. Assume each agent $i \in \mathcal{I}_m$ maintains a (local) variable $x_i \in \mathbb{R}^{n_i}$ of its own, which needs to satisfy a constraint of the following form:

$$x_i \in \mathcal{D}_i\left((x_j)_{j\in\mathcal{N}_i^+}\right). \tag{3.1}$$

Here, $\mathcal{N}_i^+ \subset \mathcal{I}_m \setminus \{i\}$ is a set of agents whose variables are needed to determine the feasible set of x_i; $(x_j)_{j\in\mathcal{N}_i^+}$ is the stacked vector of all the variables of agents in $\mathcal{N}_i^+$; and $\mathcal{D}_i\left((x_j)_{j\in\mathcal{N}_i^+}\right)$ is a subset of $\mathbb{R}^{n_i}$ which may vary with $(x_j)_{j\in\mathcal{N}_i^+}$. Equivalently, the constraint (3.1) can be written as

$$\left(x_i, (x_j)_{j\in\mathcal{N}_i^+}\right) \in \mathcal{F}_i, \tag{3.2}$$

where $\mathcal{F}_i$ is a suitably chosen subset of the product space of x_i and $(x_j)_{j\in\mathcal{N}_i^+}$.

The constraint (3.1) on the variable of agent i is in general non-local as it depends on the variables of other agents in $\mathcal{N}_i^+$. In the case of $\mathcal{N}_i^+ = \emptyset$, the feasible set $\mathcal{D}_i$

becomes a fixed subset of $\mathbb{R}^{n_i}$ and the constraint on x_i becomes local. Due to privacy concern, the constraint $\mathcal{F}_i$ (thus $\mathcal{D}_i$) is assumed to be private to each agent $i \in \mathcal{I}_m$ while the local variable x_i is shared with other neighboring agents.

Example 3.1.1 *Consider the example shown in Fig. 3.1. There are four agents with the local variables x_i and the local constraints $\mathcal{F}_i$, $i \in \mathcal{I}_4$. In Fig. 3.1, the local variables are labeled on the right; the local constraints are labeled on the left; the solid lines represent the constraint couplings across agents. Except for agent 1, the local constraint of every other agent is non-local.*

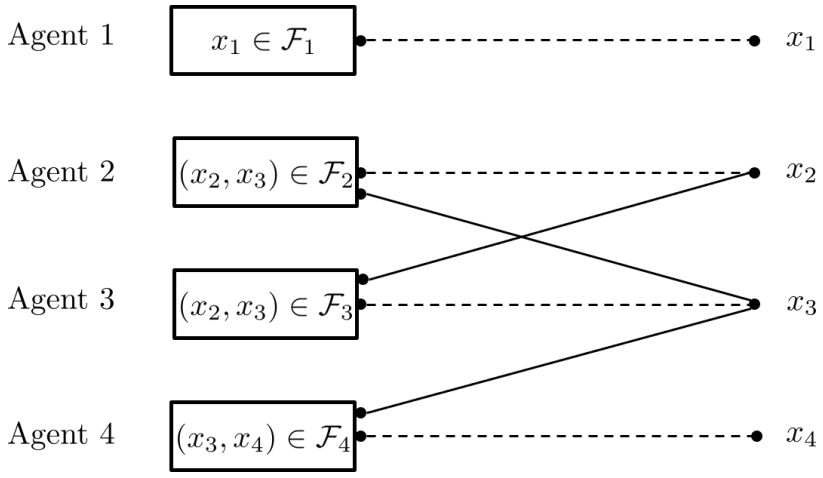


Fig. 3.1. Dependence Illustration of Example 3.1.1.

A directed graph $\mathcal{G}_d$, called the (constraint) *dependency graph*, can be constructed to represent the interdependency of the agents' feasibility: $\mathcal{G}_d$ has the vertex set $\mathcal{I}_m$ and a directed edge from j to i, denoted as (j, i), whenever the feasible set of x_i depends on x_j. Note that there is no self-loop in $\mathcal{G}_d$. See the left of Fig. 3.2 for $\mathcal{G}_d$ of Example 3.1.1. The aforementioned set $\mathcal{N}_i^+$ is exactly the in-neighborhood of vertex i in $\mathcal{G}_d$; thus we call agents indexed by $\mathcal{N}_i^+$ the *in-neighbors* of agent i. Similarly, the out-neighborhood of vertex i in $\mathcal{G}_d$, denoted by $\mathcal{N}_i^- \subset \mathcal{I}_m \setminus \{i\}$, indexes the *out-neighbors* of agent i, namely, agents whose variables' feasibility depends (at least

partially) on the value of x_i. The two neighborhoods $\mathcal{N}_i^+$ and $\mathcal{N}_i^-$ may overlap or even be identical (see Example 3.1.3 below). Denote by $\mathcal{N}_i := \mathcal{N}_i^+ \cup \mathcal{N}_i^-$ the neighbors of agent i.

Since the agents' constraints are coupled, to ensure feasibility they need to communicate with each other to share their local variables (but not their local constraints due to privacy consideration). The allowable communication among agents is represented by the *communication graph* $\mathcal{G}_c$, which is a directed graph with the vertex set $\mathcal{I}_m$ and the edge set such that a directed edge from j to i exists whenever agent i can receive information from agent j via direct communication.

Assumption 3.1.1 (Communicability) *The communication graph $\mathcal{G}_c$ contains the union of $\mathcal{G}_d$ and its transpose $\mathcal{G}_d^\top$* [1].

Assumption 3.1.1 implies that each agent can have two-way communications (i.e. send information to and receive information from) with any of its in-neighbors and out-neighbors. In other words, the communication is bi-directional between two agents whenever one's feasibility depends on the other's variable. See the right of Fig. 3.2 for $\mathcal{G}_c$ of Example 3.1.1. The following Example 3.1.2 demonstrates why the bidirectional communication is necessary.

Fig. 3.2. Dependence graph (left) and communication graph (right) of Example 3.1.1.

[1]The transpose graph $\mathcal{G}_d^T$ is obtained by reversing the direction of every edge of $\mathcal{G}_d$.

Example 3.1.2 *Consider two agents with local variables $x_1, x_2 \in \mathbb{R}$ and local constraints*

$$\mathcal{F}_1 : x_1 = x_2,\ x_1 \leq 5 \quad \text{and} \quad \mathcal{F}_2 : 4 \leq x_2 \leq 7,$$

respectively. The dependence graph $\mathcal{G}_d$ has only one edge $(2,1)$. Assume $x_1(0) = 5$, $x_2(0) = 6$. If agent 2 can not obtain information from agent 1, it will stick to its initial value and never reach consensus with agent 1 on the value of x_2.

Example 3.1.3 *The linear equation $Ax = b$ with*

$$A = \left[\begin{array}{cc|c} 1 & 0 & -1 \\ \hline 1 & 1 & 1 \\ 0 & 1 & 1 \end{array}\right] \text{ and } b = \left[\begin{array}{c} 0 \\ \hline 0 \\ -1 \end{array}\right] \tag{3.3}$$

has a unique solution $x^* = A^{-1}b = (1, -2, 1)$. Partition $x \in \mathbb{R}^3$ into $x = (x_1, x_2)$ where $x_1 \in \mathbb{R}^2$ and $x_2 \in \mathbb{R}$ are the variables of agents 1 and 2, respectively. With the row (constraint) partitions of A and b in (3.3), the private constraint of agent 1 is underdetermined for x_1: $\begin{bmatrix} 1 & 0 \end{bmatrix} x_1 - x_2 = 0$, while the private constraint of agent 2 is overdetermined for x_2:

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_1 + \begin{bmatrix} 1 \\ 1 \end{bmatrix} x_2 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}.$$

The neighbor sets of the two agents are given by $\mathcal{N}_1^+ = \mathcal{N}_1^- = \{2\}$ and $\mathcal{N}_2^+ = \mathcal{N}_2^- = \{1\}$, resulting in the corresponding dependence graph $\mathcal{G}_d$ with the edge set $\{(1,2),(2,1)\}$.

Finally we formulate the problem to be studied in this chapter as bellow.

Problem 3.1 (Distributed Feasibility Problem) *Design distributed algorithms consistent with the communication graph and maintaining the privacy of individual agents' constraints so that a value of $(x_i)_{i\in\mathcal{I}_m}$ can be (asymptotically) obtained that satisfies the private constraints of all agents.*

Denote $x := (x_i)_{i\in\mathcal{I}_m} \in \mathbb{R}^n$ where $n = \sum_{i\in\mathcal{I}_m} n_i$. The following assumptions are imposed throughout this chapter.

Assumption 3.1.2 (Feasibility) *There exists at least one x that satisfies all m constraints in* (3.2).

Assumption 3.1.3 (Convexity) *The feasible set $\mathcal{F}_i$ in* (3.2) *is nonempty, closed and convex for each $i \in \mathcal{I}_m$.*

As a consequence of Assumption 3.1.3, the feasible set $\mathcal{D}_i\left((x_j)_{j\in\mathcal{N}_i^+}\right)$ in (3.1) is also convex.

For the non-feasible problems where Assumption 2 does not hold, there is no $x = (x_i)_{i\in\mathcal{I}_m}$ satisfying all private constraints. In this case, we will seek the secondary goal of finding $\tilde{x}$ that is "closest" to the value that satisfies all the private constraints. More specifically, x is called δ-feasible if the distance of $(x_i, (x_j)_{j\in\mathcal{N}_i^+})$ to the feasible set $\mathcal{F}_i$ is at most $\delta \geq 0$ for each $i \in \mathcal{I}_m$. We will look for $\tilde{x}$ such that it is δ-feasible for the smallest possible δ (which we denote by $\delta_{\min}$). This secondary problem can be solved via Problem 3.1 by enlarging all the private feasible sets, i.e., by replacing the private constraints with

$$\left(x_i, (x_j)_{j\in\mathcal{N}_i^+}\right) \in \tilde{\mathcal{F}}_i = \mathcal{F}_i + \delta\mathcal{B}_i,\ i \in \mathcal{I}_m, \tag{3.4}$$

where $\mathcal{B}_i$ is the unit ball with dimension n_i and $\delta \geq 0$. Note that the enlargement size δ is identical for all feasible sets. See Fig. 3.3 for illustration.

The desired $\tilde{x}$ and $\delta_{\min}$ can be computed via solving a series of Problem 1: first solve Problem 3.1 with private constraints (3.4) with a large enough δ_0, and then gradually decrease δ and solve the corresponding feasibility problem until it becomes infeasible. The obtained sequence of feasible solution x^* and δ will converge to $\tilde{x}$ and $\delta_{\min}$, respectively.

3.2 Application Examples

Three instances of Problem 3.1 are presented below.

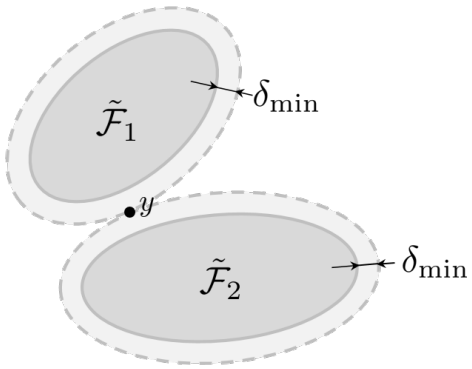


Fig. 3.3. Enlarge private feasible sets $\tilde{\mathcal{F}}_1 := \left\{x \middle| \left(x_1, (x_j)_{j\in\mathcal{N}_1^+}\right) \in \mathcal{F}_2\right\}$ and $\tilde{\mathcal{F}}_2 := \left\{x \middle| \left(x_2, (x_j)_{j\in\mathcal{N}_2^+}\right) \in \mathcal{F}_2\right\}$ by size $\delta_{\min}$ to get the common feasible point y.

3.2.1 Distributed Solution of Linear Programs/Equations

Let $A \in \mathbb{R}^{\ell\times n}$, $b \in \mathbb{R}^{\ell}$ be such that the linear program $Ax \leq b$ has at least one feasible solution x^*. Suppose that different portions of the variable x and the inequalities are held separately by a group of agents indexed by $\mathcal{I}_m$, i.e., there exist the block partitions $x = (x_1, \cdots, x_m)$,

$$\left[\begin{array}{c|c} A & B \end{array}\right] = \left[\begin{array}{ccc|c} A_{11} & \cdots & A_{1m} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ A_{m1} & \cdots & A_{mm} & b_m \end{array}\right]$$

so that agent $i \in \mathcal{I}_m$ has n_i variables, $x_i \in \mathbb{R}^{n_i}$, and ℓ_i private linear inequality constraints, $A_{i1}x_1 + \cdots + A_{im}x_m \leq b_i \in \mathbb{R}^{\ell_i}$. Here, we assume $n_i, \ell_i \geq 0$ with $\sum_i n_i = n$ and $\sum_i \ell_i = \ell$; and "$\leq$" denotes entry-wise comparison. Agent i has the neighbor sets $\mathcal{N}_i^+ = \{j \in \mathcal{I} \,|\, A_{ij} \neq 0\}$ and $\mathcal{N}_i^- = \{j \in \mathcal{I} \,|\, A_{ji} \neq 0\}$ and its constraint can be recast as $A_{ii}x_i + \sum_{j\in\mathcal{N}_i^+} A_{ij}x_j \leq b_i$. Distributed solution of the above linear program (and as a special case, the linear equation $Ax = b$) is an instance of Problem 3.1. Example 3.1.3 is one such instance.

Example 3.2.1 *Consider the linear program* $-\varepsilon\mathbf{1} \leq Ax - b \leq \varepsilon\mathbf{1}$ *with* $x = (x_1, x_2, x_3) \in \mathbb{R}^3$,

$$A = \left[\begin{array}{c|c|c} 1 & 0 & -1 \\ \hline 0 & 0 & 1 \\ \hline 0 & 1 & 1 \end{array}\right], \quad b = \left[\begin{array}{c} 1 \\ \hline -1 \\ \hline 1 \end{array}\right], \quad \mathbf{1} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}. \tag{3.5}$$

There are three agents with the variables $x_1, x_2, x_3 \in \mathbb{R}$ *and the following private constraints, respectively:* $|x_1 - x_3 - 1| \leq \varepsilon$ *for agent 1;* $|x_3 + 1| \leq \varepsilon$ *for agent 2; and* $|x_2 + x_3 - 1| \leq \varepsilon$ *for agent 3. Their neighbor sets are* $\mathcal{N}_1^+ = \{3\}$, $\mathcal{N}_1^- = \emptyset$; $\mathcal{N}_2^+ = \mathcal{N}_2^- = \{3\}$; $\mathcal{N}_3^+ = \{2\}$ *and* $\mathcal{N}_3^- = \{1, 2\}$. *Note that the constraint of agent 2 does not involve its own variable* x_2, *which is allowed in our problem formulation. Further,* $x^* = A^{-1}b = (0, 2, -1)$ *is a feasible solution for any* $\varepsilon \geq 0$.

3.2.2 Network Localization

Consider a group of agents (sensors, robots, vehicles) deployed on $\mathbb{R}^2$ with unknown locations $x_i \in \mathbb{R}^2$, $i \in \mathcal{I}$. Suppose each agent $i \in \mathcal{I}$ is equipped with sensors that can measure its relative distance and/or orientation w.r.t. some other agents $j \in \mathcal{N}_i^+$ within its sensing range.

(i) *Relative orientation (Angle-of-Arrival) measurement*: the direction of the vector $x_j - x_i$ is measured against a compass onboard agent i. This imposes a constraint as $\angle(x_j - x_i) \in \Theta_{ij}$, where $\angle$ denotes the phase angle and Θ_{ij} is a singleton $\{\theta_{ij}\}$ if the measurement is precise and an interval $[\theta_{ij} - \delta, \theta_{ij} + \delta]$ if the measurement is imprecise.

(ii) *Relative distance measurement*: the distance $\|x_j - x_i\|$ is measured using, e.g., the strength of signal received by agent i from agent j. This incurs a constraint as $r_1 \leq \|x_i - x_j\| \leq r_2$.

The private constraint of agent i consists of all the above constraints for $j \in \mathcal{N}_i^+$. The network localization problem is to find the locations of all agents consistent with the measurement data. This is an instance of Problem 3.1 if $r_1 = 0$.

3.2.3 Comfort Assurance in Multi-Zone Buildings

Consider a building with multiple thermal zones indexed by $\mathcal{I}_m$. The thermal dynamics of zone i is

$$x_i(k+1) = A_{ii}x_i(k) + \sum_{j\in\mathcal{N}_i} A_{ij}x_j(k) + B_iu_i(k) + F_iw_i(k),\ k = 0, 1, \ldots, \qquad (3.6)$$

where x_i is the state variable, u_i the local cooling/heating control input, and w_i the (predicted) external perturbations, of zone i. $\mathcal{N}_i$ consists of all those zones with thermal exchanges with zone i. Given the time horizon $k = 0, 1, \ldots, N$, the objective is to determine if there exist control sequences $u_i(k)$, $k = 0, \ldots, N-1$, for all zones $i \in \mathcal{I}_m$ satisfying the following constraints:

(a) *Comfort constraint*: $x_i(k) \in \mathcal{X}_i$ for some compact convex sets $\mathcal{X}_i$, $\forall k$, $i \in \mathcal{I}_m$;

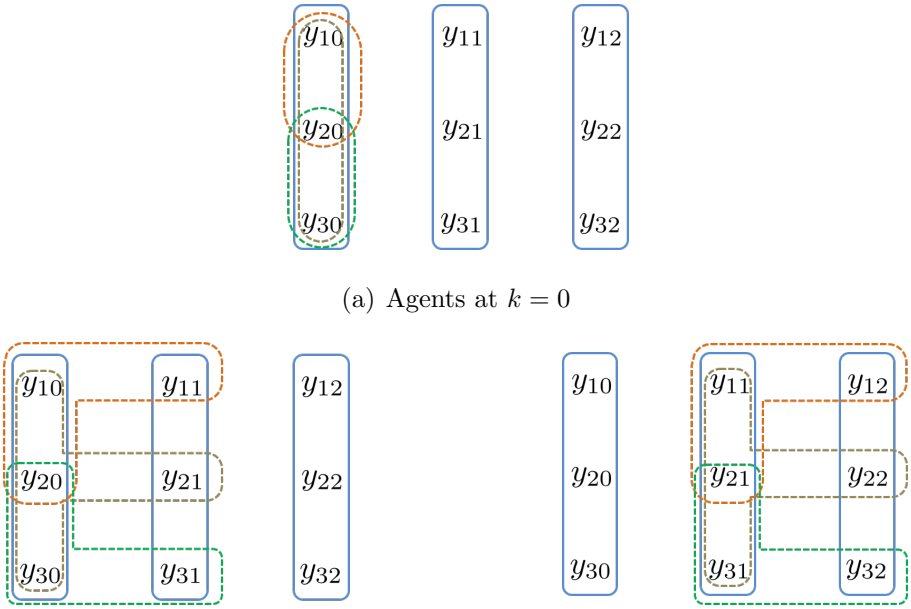
(b) *Peak demand constraint*: $\sum_{i\in\mathcal{I}_m} \phi_i(u_i(k)) \leq \phi_{max}$, $\forall k$, where $\phi_i(\cdot)$ are convex functions.

(c) *Dynamics constraints* as given by (3.6).

The above problem can be cast as a distributed feasibility problem. Each zone i can be identified as an agent with the variable $(x_i(k+1), u_i(k))_{0\leq k\leq N-1}$. Alternatively, one can assign the variable $y_{ik} = (x_i(k+1), u_i(k))$ to an agent (ik) for the zone i at time k. The latter will result in a larger but more sparsely connected dependency graph $\mathcal{G}_d$. For illustration, consider the example of three offices with the simple layout in Fig. 3.4, where only adjacent rooms have thermal exchanges. Given the prediction horizon $N = 3$, the constraint couplings across agents are shown in Fig.3.5.

Fig. 3.4. Layout of three offices

In Fig. 3.5, a blue box associated with each time k indicates the coupling induced by the peak demand constraint at this moment; the 'L' or 'T' shaped dashed line boxes



(a) Agents at $k = 0$

(b) Agents at $k = 1$

(c) Agents at $k = 2$

Fig. 3.5. Three offices: constraint couplings across agents. In three figures, the blue solid line boxes represent the couplings at each time step induced by peak demand constraints while the dashed line boxes show the couplings induced by dynamics constraints where figure (a) is for the three agents at time $k = 0$, (b) for agents at $k = 1$, and (c) for agents at time $k = 2$ and the orange, gray, and green dashed lines are for agents $1, 2, 3$, respectively.

show the spatial and temporal dynamics couplings among agents, e.g., the orange 'L' shaped dashed line box in the figure (b) means that, for the local variable y_{11} of agent 11, the local constraint induced by dynamics also involves the other two agents' variables, y_{10} and y_{20}, induced by the temporal and spatial couplings, respectively; since the comfort constraints are always local, the corresponding couplings are not shown in the figure. Note that the couplings exist only 'locally' between one agent and its 'close' neighbors, which consist of spatially and temporally adjacent agents.

3.3 Problem Reformulation

We first present an equivalent formulation of Problem 3.1. Suppose besides its own variable x_i, agent i maintains an additional set of variables, $(x_{ji})_{j\in\mathcal{N}_i^+}$, where x_{ji} represents the value of agent j's variable as *desired* by agent i (which could differ from the actual value of x_j). Define

$$\mathbf{x}_i := \left(x_i, (x_{ji})_{j\in\mathcal{N}_i^+}\right)$$

to be the *augmented variable* of agent i with the dimension $N_i = n_i + \sum_{j\in\mathcal{N}_i^+} n_j$. Then the totality of all $\mathbf{x}_i$'s, denoted by $\mathbf{x} := (\mathbf{x}_i)_{i\in\mathcal{I}_m}$, has dimension $N = \sum_{i\in\mathcal{I}_m} N_i$. For sparse dependency graph $\mathcal{G}_d$, $N \ll mn$. With $\mathbf{x}_i$'s, Problem 3.1 can be reformulated as follows.

Problem 3.2 *Design distributed algorithms consistent with the communication graph $\mathcal{G}_c$ so that a value of $\mathbf{x}$ is asymptotically obtained that satisfies,*

$$\mathbf{x}_i \in \mathcal{F}_i, \quad \forall i \in \mathcal{I}_m, \tag{3.7}$$

$$x_i = x_{ik}, \quad \forall i \in \mathcal{I}_m,\ \forall k \in \mathcal{N}_i^-. \tag{3.8}$$

The constraint (3.7) is from (3.2) with x_j replaced by x_{ji}, which is local as it only involves agent i's augmented variable $\mathbf{x}_i$. The consensus constraint (3.8) ensures agent i's variable x_i to be the same as that desired by its out-neighbors, inducing the non-local consensus set

$$\mathcal{C}_i := \{(x_i, (x_{ik})_{k\in\mathcal{N}_i^-}) \,|\, x_i = x_{ik},\ \forall k \in \mathcal{N}_i^-\}. \tag{3.9}$$

Define

$$\mathcal{A}_1 = \mathcal{F}_1 \times \cdots \times \mathcal{F}_m, \quad \mathcal{A}_2 = M^\top (\mathcal{C}_1 \times \cdots \times \mathcal{C}_m) \tag{3.10}$$

to be the feasible set and consensus subspace of $\mathbf{x}$, respectively, where $M \in \mathbb{R}^{N\times N}$ is a permutation matrix so that each variable x_i and its desired values by out-neighbors, $x_{ik}, k \in \mathcal{N}_i^-$, are put consecutively in a block in the order of $i = 1, \ldots, m$. Clearly, $\mathcal{A}_1 \cap \mathcal{A}_2$ is the solution set of Problem 3.2.

It is easy to see that the solutions to Problems 3.1 and 3.2 have a one-to-one correspondence; hence they are equivalent. By Assumption 3.1.2, Problem 3.2 has a feasible solution $\mathbf{x}^* = (\mathbf{x}_i^*)_{i\in\mathcal{I}_m}$. Next we present four algorithms to solve Problem 3.2 (and thus Problem 3.1).

3.4 Distribued Synchronous/Asynchronous Algorithms

This section summarizes the four algorithms we proposed to solve Problem 3.1, which include the synchronous/asynchronous algorithms and the synchronous algorithm with general weights algorithm.

3.4.1 Synchronous Algorithm

The first algorithm iteratively solves Problem 3.2 with all agents updating synchronously in each iteration. The update at round t consists of two stages: first each agent i updates its augmented variable from $\mathbf{x}_i^t$ to $\mathbf{z}_i^t$ via the (relaxed) projection operator P_i onto its local feasible set $\mathcal{F}_i$ as in (3.11); then, each agent i simultaneously collects from its out-neighbors their updated desired values of x_i, $(z_{ik}^t)_{k\in\mathcal{N}_i^-}$, to obtain x_i^{t+1} via the consensus operation (3.12), and broadcasts x_i^{t+1} back to all of its out-neighbors as their updated values x_{ik}^{t+1} as in (3.13). The iterations above are detailed below and summarized in Algorithm 1.

(i) (Relaxed projection)

Agent i computes $\mathbf{z}_i^t := (z_i, (z_{ji})_{j\in\mathcal{N}_i^+})$ from $\mathbf{x}_i^t$ via the relaxed projection operation

$$\mathbf{z}_i^t = P_i(\mathbf{x}_i^t) := (1-\alpha_i)\mathbf{x}_i^t + \alpha_i \cdot P_{\mathcal{F}_i}(\mathbf{x}_i^t)\,. \tag{3.11}$$

Here, $P_{\mathcal{F}_i}$ denotes the orthogonal projection operator onto the local feasible set $\mathcal{F}_i$ and $\alpha_i \in (0,2)$ is a constant.

(ii) (Consensus)

Agent i collects $(z_{ik}^t)_{k\in\mathcal{N}_i^-}$ from its out-neighbors to update its variable according to

$$x_i^{t+1} = Q_i(z_i^t, (z_{ik}^t)_{k\in\mathcal{N}_i^-}) := \frac{1}{|\mathcal{N}_i^-|+1}\left(z_i^t + \sum_{k\in\mathcal{N}_i^-} z_{ik}^t\right), \tag{3.12}$$

and then sends back x_i^{t+1} to out-neighbors for updating

$$x_{ik}^{t+1} = x_i^{t+1},\ k \in \mathcal{N}_i^-. \tag{3.13}$$

If agent i has no out-neighbors, i.e., $\mathcal{N}_i^- = \emptyset$, the update (3.12) will be trivial: $x_i^{t+1} = z_i^t$.

Algorithm 1 Synchronous Algorithm

Initialize $\mathbf{x}^0$ and let $t \leftarrow 0$;
repeat
  for all $i \in \mathcal{I}_m$ **do** {Relaxed projection}
    Agent i computes $\mathbf{z}_i^t$ according to (3.11);
  end for
  for all $i \in \mathcal{I}_m$ **do** {Consensus}
    Agent i receives z_{ik}^t from all out-neighbors k;
    Agent i computes x_i^{t+1} according to (3.12);
    Agent i sends back x_i^{t+1} to all out-neighbors for updating x_{ik}^{t+1}as in (3.13);
  end for
  $t \leftarrow t + 1$;
until certain convergence criteria are met
Return $\mathbf{x}^t$.

In Algorithm 1, all agents update their augmented variables in parallel at each round. Intuitively the relaxed projection (3.11) helps to improve the satisfaction of the local feasibility constraint (3.7) while the consensus step (3.12) together with the broadcast step (3.13) helps to reach consensus on the value of x_i among agent i and its out-neighbors.

Note that in Algorithm 1, each agent only communicates with its out-neighbors in the consensus step and this communication is bidirectional, which is allowed by Assumption 3.1.1.

The convergence properties of the synchronous algorithm are characterized by the following two theorems whose proofs will be provided in Section 3.5.1.

Theorem 3.4.1 *Starting from any initial guess* $\mathbf{x}^0$*, the sequence* $\{\mathbf{x}^t\}$ *generated by Algorithm 1 will converge asymptotically to a feasible solution to Problem 3.2.*

Theorem 3.4.2 *Suppose that there exists a feasible solution* $x^{\mathrm{o}} \in \mathbb{R}^n$ *to Problem 3.1 such that for each* $i \in \mathcal{I}_m$, $\left(x_i^{\mathrm{o}}, (x_j^{\mathrm{o}})_{j\in\mathcal{N}_i^+}\right) \in \mathcal{F}_i^{\mathrm{o}}$*, i.e., an interior point of* $\mathcal{F}_i$*. Then Algorithm 1 with* $\alpha_i = 1$ *for all* $i \in \mathcal{I}_m$ *converges exponentially fast to a feasible solution of Problem 3.2 starting from any initial point.*

Remark 3.4.1 *The well-known projected consensus algorithm in [27, 28], denoted as Pro-Con, has been proved that the distance of each iteration to the feasible solution set decays exponentially fast. This is weaker than the conclusion of Theorem 3.4.2 that the iterations themselves converge exponentially to one feasible solution. Without taking account of the differences in implementation details, the main reason is that the relaxed projection and consensus operations are paracontractions (see Definition 2.2.4 in Section 3.5) while the general weight consensus operation adopted in Pro-Con is not.*

3.4.2 Asynchronous Algorithm

The synchronous operations in Algorithm 1 can be difficult to ensure in practice, which is extended to be asynchronous in this section. At round t, each agent i independently determines whether it will update or not and, if so, chooses one of the following two operations to perform: carrying out the relaxed projection operation (3.11) to satisfy its local feasibility constraint; reaching consensus on its own

variable x_i with a subset of its out-neighbors, denoted by $\mathcal{N}_{i,t}^- \subseteq \mathcal{N}_i^-$, through the averaging step

$$x_i^{t+1} = Q_i^t\left(x_i^t, (x_{ik}^t)_{k\in\mathcal{N}_{i,t}^-}\right) := \frac{1}{|\mathcal{N}_{i,t}^-|+1}\left(x_i^t + \sum_{k\in\mathcal{N}_{i,t}^-} x_{ik}^t\right) \tag{3.14}$$

followed by the broadcast step (3.13) with $\mathcal{N}_i^-$ replaced by $\mathcal{N}_{i,t}^-$. In other words, depending on agent i's update choice at round t, it will belong to one of the three sets, the idle, projection, and consensus sets, denoted by $\mathcal{I}_{\text{idle}}^t$, $\mathcal{I}_P^t$, $\mathcal{I}_Q^t$, respectively, and then perform the corresponding operation. Note that $\mathcal{I}_{\text{idle}}^t$, $\mathcal{I}_P^t$ and $\mathcal{I}_Q^t$ constitutes a partition of I_m. The Algorithm 2 bellow describes this asynchronous version.

Algorithm 2 Asynchronous Algorithm

Initialize $\mathbf{x}^0$ and set $t \leftarrow 0$;
repeat
  for all $i \in \mathcal{I}_m$ **do**
    Agent i idles
    or
    {Relaxed projection}
    Agent i updates $\mathbf{x}_i^{t+1}$ according to (3.11) with $\mathbf{z}_i^t$ replaced by $\mathbf{x}_i^t$;
    or
    {Partial consensus}
    Agent i receives x_{ik}^t from the out-neighbor k belonging to the subset $\mathcal{N}_{i,t}^- \subseteq \mathcal{N}_i^-$;
    Agent i computes x_i^{t+1} according to (3.14);
    Agent i sends x_i^{t+1} back to its out-neighbors $k \in \mathcal{N}_{i,t}^-$ as their updated values x_{ik}^{t+1};
  end for
  $t \leftarrow t + 1$;
until certain convergence criteria are met
Return $\mathbf{x}^t$.

Remark 3.4.2 (Algorithm 2b) *A special case of Algorithm 2 is* $|\mathcal{N}_{i,t}^-| = 1$ *in* (3.14)*, i.e., agent* $i \in \mathcal{I}_Q^t$ *performs its consensus operation with only one out-neighbor* $k \in \mathcal{N}_i^-$ *that is either randomly picked or resulted from some extreme situations. In this case, the equally weighted average* (3.14) *can be relaxed to*

$$\begin{bmatrix} x_i^{t+1} \\ x_{ik}^{t+1} \end{bmatrix} = (W_{ik} \otimes I_{n_i}) \begin{bmatrix} x_i^t \\ x_{ik}^t \end{bmatrix}. \tag{3.15}$$

Here $W_{ik} \in \mathbb{R}^{2\times 2}$ *is a constant doubly stochastic matrix with strictly positive entires. To carry out the update* (3.15)*, agent* i *first collects* x_{ik}^t *from agent* k*, then computes the update values for both itself and agent* k*, and finally sends the latter* x_{ik}^{t+1} *back to agent* k*. With this relaxation, agent* i *may not reach consensus with any out-neighbors, i.e.,* $x_i^{t+1} \neq x_{ik}^{t+1}, \forall k \in \mathcal{N}_i^-$*. We will refer to this relaxed algorithm as Algorithm 2b and show its convergence in Theorem 3.4.4.*

To establish the convergences of Algorithms 2 and 2b, we impose two assumptions.

Assumption 3.4.1 (Semaphore) *At round* t*, for any agent* i *carrying out the partial consensus operation* (3.14)*, none of (active) its out-neighbors* $\mathcal{N}_{i,t}^-$ *in* (3.14) *will be performing the relaxed projection operation, i.e.,* $\mathcal{N}_{i,t}^- \cap \mathcal{I}_P^t = \emptyset,\ \forall i \in \mathcal{I}_Q^t$*.*

This assumption implies that at each round, each variable either does not change or changes only once resulted from the relaxed projection or partial consensus. This is critical to establish the convergences later.

Assumption 3.4.2 (Infinite Appearances) *(a) For each* $i \in \mathcal{I}_m$*,* $i \in \mathcal{I}_P^t$ *for infinitely many* $t \in \{0, 1, \ldots\}$*; (b) Any pair of neighboring agents is involved in the (partial) consensus operation* (3.14) *for an infinite number of times.*

Assumption 3.4.2 is less restrictive than both periodic and uniformly repeated appearances which require that the two operations in Assumption 3.4.2 are involved once and at least once every T rounds, respectively, for a positive integer T. Note that Assumption 3.4.2(b) imposes constraints on both $\mathcal{I}_Q^t$ and $\mathcal{N}_{i,t}^-$, $\forall i \in \mathcal{I}_Q^t$, such that their

combinations will guarantee that any neighboring agents have enough communication on their variables to reach consensus.

The following two theorems establish the convergences of Algorithms 2 and 2b, respectively. Their proofs will be given later on in Section 3.5.3.

Theorem 3.4.3 *Suppose Assumptions 3.4.1 and 3.4.2 hold. Starting from any initial guess* $\mathbf{x}^0$*, the sequence* $\{\mathbf{x}^t\}$ *generated by Algorithm 2 converges asymptotically to a feasible solution to Problem 3.2.*

Theorem 3.4.4 *Suppose Assumptions 3.4.1 and 3.4.2 hold. Starting from any initial guess* $\mathbf{x}^0$*, the sequence* $\{\mathbf{x}^t\}$ *returned by Algorithm 2b will converge asymptotically to a feasible solution to Problem 3.2.*

3.4.3 Generalized Synchronous Algorithm

In this section, we generalize Algorithm 1 in two perspectives: (i) along the spirit of Algorithm 2, each agent independently determines in each round if it will be activated to perform updates and, if so, the type of updates to be carried out, (ii) time-varying general weights are adopted in the consensus operation.

For the perspective (i), at round t only agents in the two subsets of $\mathcal{I}_m$, denoted by $\mathcal{I}_P^t$ and $\mathcal{I}_Q^t$, are assumed to perform the relaxed projection and consensus operations of Algorithm 1, respectively. Note that this algorithm remains synchronous in a way that all agents must finish the relaxed projection step before moving to the consensus operation, which is different from the parallel implementation in Algorithm 2. Therefore, $\mathcal{I}_P^t \cap \mathcal{I}_Q^t$ can be non-empty, i.e., an agent can participate in both the projection and the consensus operations. This extension accommodates the practical situation that some agents may be unable to update due to temporary breakdown or communication blackouts.

For the perspective (ii), the most straightforward generalization is replacing step (3.12) of agent $i \in \mathcal{I}_Q^t$ by the following:

$$x_i^{t+1} = w_{ii}^t z_i^t + \sum_{k \in \mathcal{N}_i^-} w_{ik}^t z_{ik}^t, \tag{3.16}$$

where $w_{ii}^t \in \mathbb{R}$ and $w_{ik}^t \in \mathbb{R}, k \in \mathcal{N}_i^-$ are time-varying weights assigned by agent i and satisfy that every weight is bounded from below by $\underline{w} > 0$ and their sum is one. Unfortunately, this generalization does not work in general, even in the simplest case where the weights are constant and $\mathcal{I}_P^t = \mathcal{I}_Q^t = \mathcal{I}_m$, i.e., the extension (i) above is removed. This is shown by Example 3.4.1.

Example 3.4.1 *Consider the linear equation $Ax = b$ where*

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

which has a unique solution $x = (x_1, x_2, x_3) = 0$. Suppose it is solved by three agents each in charge of one component of x and one row constraint. Then the augmented variables are $\mathbf{x}_1 = (x_1, x_{21}, x_{31})$, $\mathbf{x}_2 = (x_2, x_{12})$, and $\mathbf{x}_3 = (x_3, x_{13})$. Suppose the stochastic matrices in (3.16) are given by $w_{11}^t = 0.9$, $w_{12}^t = 0.05$, $w_{13}^t = 0.05$, $w_{21}^t = 0.2$, $w_{22}^t = 0.8$, $w_{31}^t = 0.1$, $w_{33}^t = 0.9$. Assuming that $\mathcal{I}_P^T = \mathcal{I}_Q^t = \mathcal{I}_m$, the iteration of Algorithm 3 can be written as $\mathbf{x}^{k+1} = F\mathbf{x}^k$ for some matrix $F \in \mathbb{R}^{7\times 7}$. It can be verified numerically that F has an eigenvalue 1.1246. *Therefore, Algorithm 3 does not converge to the solution 0 starting from some (indeed, almost all) initial guesses $\mathbf{x}^0$.*

Instead of (3.16), the following operation is adopted to replace the consensus step (3.12) and the broadcast step (3.13) in Algorithm 1:

$$\left(x_i^{t+1}, (x_{ik}^{t+1})_{k \in \mathcal{N}_i^-}\right) = W_i^t \otimes I_{n_i} \left(z_i^t, (z_{ik}^t)_{k \in \mathcal{N}_i^-}\right). \tag{3.17}$$

Here, $W_i^t \in \mathbb{R}^{(|\mathcal{N}_i^-|+1)\times(|\mathcal{N}_i^-|+1)}$ is a time-varying weight matrix specified by agent i that satisifies Assumption 3.4.3 to be defined below. In the case where agent i at

round t receives desired values z_{ik}^t from only a subset of out-neighbors, $\mathcal{N}_{i,t}^- \subset \mathcal{N}_i^-$, the rows and columns of W_i^t corresponding to the other (silent) out-neighbors, i.e., $\mathcal{N}_i^- \backslash \mathcal{N}_{i,t}^-$, are set to proper unit vectors as their desired values of x_i remain unchanged at this round. With this generalized weight matrix W_i^t, the updated values for out-neighbors, $x_{ik}^{t+1}, k \in \mathcal{N}_i^-$, will be different from x_i^{t+1} in general, i.e., agent i does not reach consensus with its out-neighbors on its variable x_i at each round, which is the main difference between (3.17) and the consensus step in (3.12). The potential benefits of adopting W_i^t include 1). speeding up the convergence by properly assigning weights, especially when the desired values from some out-neighbors are known to be more accurate/important than others, and 2). accommodating the practical situation that agent i loses communication with some out-neighbors occasionally.

Assumption 3.4.3 (Weights Rule) *For matrix W_i^t, $\forall t \geq 0$ and $\forall i \in \mathcal{I}_Q^t$,*

(a) W_i^t is doubly stochastic;

(b) there is a scalar $\underline{w} > 0$ such that entries of W_i^t corresponding to all agents in $\{i\} \cup \mathcal{N}_{i,t}^-$ are bounded from below by $\underline{w}$, i.e., $[W_i^t]_{kl} \geq \underline{w}$ for all $k, l \in \{i\} \cup \mathcal{N}_{i,t}^-$;

(c) for agent $k \in \mathcal{N}_i^- \backslash \mathcal{N}_{i,t}^-$, the diagonal entries $[W_i^t]_{kk} = 1$ while the other elements in the row and column related to agent k are set to 0.

Assumption 3.4.3(b) guarantees that once agent i obtains an out-neighbor's desired value z_{ik}^t, this value will make significant contributions to the consensus outcome. Although Assumption 3.4.3(a) requires W_i^t to be doubly stochastic, such a matrix is chosen by agent i alone without any coordination with other agents and will in general be different from those chosen by other agents. In comparison, the traditional double stochasticity assumption (e.g., [83] and Assumption 3 in [27]) needs all of the agents to coordinate to choose a single doubly stochastic matrix.

The generalized synchronous algorithm with the above two extensions is summarized in Algorithm 3. In order to establish its convergence, the following Assumption 3.4.4 is imposed.

Algorithm 3 Generalized Synchronous Algorithm

Initialize $\mathbf{x}^0$, and let $t \leftarrow 0$;
repeat
  for all $i \in \mathcal{I}_P^t$ **do** {Relaxed projection}
    Agent i computes $\mathbf{z}_i^t$ according to (3.11);
  end for
  for all $i \in \mathcal{I}_Q^t$ **do** {Generalized partial consensus}
    Agent i receives z_{ik}^t from out-neighbors $k \in \mathcal{N}_{i,t}^-$;
    Agent i computes x_i^{t+1}, x_{ik}^{t+1} according to (3.17);
    Agent i sends back x_{ik}^{t+1} to out-neighbors in $\mathcal{N}_{i,t}^-$ as their updated values;
  end for
  $t \leftarrow t + 1$;
until certain convergence criteria are met
Return $\mathbf{x}^t$.

Assumption 3.4.4 (Uniform Appearances) *(a) For each $i \in \mathcal{I}_m$, $i \in \mathcal{I}_P^t$ for infinitely many $t \in \{0, 1, \ldots\}$; (b) There exists a finite integer $T > 0$ such that, for any agent $i \in \mathcal{I}_m$, each of its out-neighbor appears at least once in $\cup_{t=t_0}^{t_0+T} \mathcal{N}_{i,t}^-$ for any integer $t_0 \geq 0$.*

Obviously, Assumption 3.4.4 is stronger than Assumption 3.4.2 in part(b) by requiring more frequent consensus operations between neighboring agents. Now we state the convergence result of Algorithm 3 in Theorem 3.4.5 below with its proof provided in Section 3.5.4. As will be seen, the convergence analysis of Algorithm 3 is much more challenging than that of Algorithm 1 since the operation (3.17) is no longer a projection onto the consensus set.

Theorem 3.4.5 *Suppose that Assumptions 3.4.3 and 3.4.4 hold. Starting from any initial guess $\mathbf{x}^0$, the sequence $\{\mathbf{x}^t\}$ generated by Algorithm 3 will asymptotically converge to a feasible solution to Problem 3.2.*

3.4.4 Generalized Asynchronous Algorithm

The following Algorithm 4 is the asynchronous version of Algorithm 3 in a way that the relaxed projection and the generalized partial consensus operations can be carried out simultaneously rather than consecutively. Its convergence is shown in Theorem 3.4.6 below with proof given in Section 3.5.5.

Algorithm 4 Generalized Asynchronous Algorithm

Initialize $\mathbf{x}^0$ and set $t \leftarrow 0$;
repeat
  for all $i \in \mathcal{I}_m$ **do**
    Agent i idles
    or
    {Relaxed projection}
    Agent i updates $\mathbf{x}_i^{t+1}$ according to (3.11) with $\mathbf{z}_i^t$ replaced by $\mathbf{x}_i^t$;
    or
    {Generalized partial consensus}
    Agent i receives x_{ik}^t from out-neighbors k belonging to the subset $\mathcal{N}_{i,t}^- \subseteq \mathcal{N}_i^-$;
    Agent i computes x_i^{t+1}, x_{ik}^{t+1} according to (3.17);
    Agent i sends x_{ik}^{t+1} back to its out-neighbors $k \in \mathcal{N}_{i,t}^-$ as their updated values;
  end for
  $t \leftarrow t + 1$;
until certain convergence criteria are met
Return $\mathbf{x}^t$.

Theorem 3.4.6 *Suppose Assumptions 3.4.1, 3.4.3, and 3.4.4 hold. Starting at any initial guess* $\mathbf{x}^0$*, the sequence* $\{\mathbf{x}^t\}$ *generated by Algorithm 4 converges asymptotically to a feasible solution to Problem 3.2.*

3.5 Convergence Proofs

3.5.1 Convergence Proof of Synchronous Algorithm

This section aims to establish the convergence and convergence rate of the synchronous Algorithm 1.

At round t of Algorithm 1, each agent $i \in \mathcal{I}_m$ first computes $\mathbf{z}_i^t = P_i(\mathbf{x}_i^t)$, where P_i is a $\frac{2-\alpha_i}{\alpha_i}$-strongly quasi-nonexpansive map with the fixed point set $\mathcal{F}_i$ according to Proposition 2.2.1. Denoting by $\mathbf{z}$ the stacked vector of all $\mathbf{z}_i$, we have

$$\mathbf{z}^t = P\left(\mathbf{x}^t\right) \tag{3.18}$$

where $P := P_1 \times \cdots \times P_m$ is a paracontraction with $\operatorname{Fix} P = \mathcal{A}_1$ defined in (3.10).

The second step, consisting of the consensus operation (3.12) followed by the broadcast (3.13), can be expressed as

$$x_{ik}^{t+1} = x_i^{t+1} = Q_i(z_i^t, (z_{ik}^t)_{k\in\mathcal{N}_i^-}),\ k \in \mathcal{N}_i^-,$$

or in a compact form

$$(x_i^{t+1}, (x_{ik}^{t+1})_{k\in\mathcal{N}_i^-}) = \tilde{Q}_i(z_i^t, (z_{ik}^t)_{k\in\mathcal{N}_i^-})$$

where $\tilde{Q}_i(\cdot) := [Q_i, \ldots, Q_i](\cdot)$ is the column concatenation of Q_i's and, with some abuse of notation, Q_i is the matrix $\frac{1}{|\mathcal{N}_i^-|+1}\mathbf{1}^\top \otimes I_{n_i}$ corresponding to the consensus operation in (3.12). It can be easily seen that, for any $i \in \mathcal{I}_m$, $\tilde{Q}_i$ is exactly the projection operation onto the consensus set $\mathcal{C}_i$ in (3.9). For simplicity, we reorder the variables of $\mathbf{x}$ as $\tilde{\mathbf{x}} = M\mathbf{x}$, where M is the permutation matrix used in (3.10). Similarly $\tilde{\mathbf{z}} = M\mathbf{z}$. Then by Lemma 2.2.1 the operator $\tilde{Q} : \tilde{\mathbf{z}}^t \mapsto \tilde{\mathbf{x}}^{t+1}$, being $\tilde{Q}_1 \times \cdots \times \tilde{Q}_m$, is a paracontraction w.r.t. the Euclidean norm with the fixed point set $\mathcal{C}_1 \times \cdots \times \mathcal{C}_m$. For the original $\mathbf{x}^t$, it follows that

$$\mathbf{x}^{t+1} = M^\top \tilde{\mathbf{x}}^{t+1} = M^\top \tilde{Q}\left(\tilde{\mathbf{z}}^t\right) = M^\top \tilde{Q} M\left(\mathbf{z}^t\right) = Q\left(\mathbf{z}^t\right),$$

with $Q := M^\top \tilde{Q} M$. By Lemma 2.2.2, the operator Q is a paracontraction w.r.t. the Euclidean norm with the fixed point set $\mathcal{A}_2$ in (3.10).

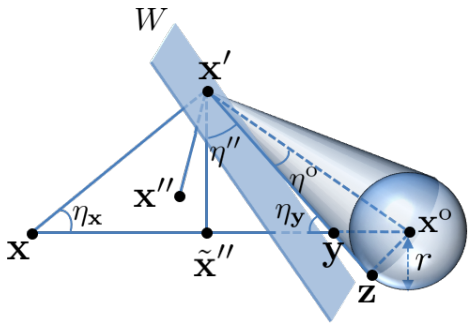


Fig. 3.6. Proof of Lemma 3.5.1.

Proof [Theorem 3.4.1]

As discussed above, the sequence $\mathbf{x}^t$ generated by Algorithm 1 is obtained from the iteration

$$\mathbf{x}^{t+1} = Q \circ P(\mathbf{x}^t), \quad t = 0, 1, \ldots,$$

where P and Q are two paracontractions w.r.t. the Euclidean norm whose sets of fixed points are specified by $\mathcal{A}_1$ and $\mathcal{A}_2$ in (3.10), respectively. By Theorem 2.2.4, $\mathbf{x}^t$ will converge to some $\mathbf{x}^* \in \mathcal{A}_1 \cap \mathcal{A}_2$, namely, a solution to Problem 3.2. ■

3.5.2 Convergence Rate of Synchronous Algorithm

To study the convergence rate of Algorithm 1, we need the following result.

Lemma 3.5.1 *Suppose that $\mathcal{E}_1$ and $\mathcal{E}_2$ are two closed convex subsets of $\mathbb{R}^d$ whose intersection $\mathcal{E}_1 \cap \mathcal{E}_2$ contains an interior point $\mathbf{x}^o$ of $\mathcal{E}_1$, i.e., there exists $r > 0$ such that the closed ball $\mathcal{B}(\mathbf{x}^o, r)$ centered at $\mathbf{x}^o$ with the radius r is contained in $\mathcal{E}_1$. Let $P_{\mathcal{E}_1}$ and $P_{\mathcal{E}_2}$ be the projection operators onto these two sets, respectively. Then, for any $\mathbf{x} \in \mathcal{E}_2 \setminus \mathcal{E}_1$, we have*

(a) $d_{\mathcal{E}_1 \cap \mathcal{E}_2}(\mathbf{x}) \leq \frac{\|\mathbf{x}-\mathbf{x}^o\|}{r} \cdot d_{\mathcal{E}_1}(\mathbf{x})$,

(b) $d_{\mathcal{E}_1}\left(P_{\mathcal{E}_2}(P_{\mathcal{E}_1}(\mathbf{x}))\right) \leq \gamma \cdot d_{\mathcal{E}_1}(\mathbf{x})$,

with the constant $\gamma := \frac{\sqrt{\|\mathbf{x}-\mathbf{x}^o\|^2 - r^2}}{\|\mathbf{x}-\mathbf{x}^o\|} \in [0, 1)$.

Proof Let $\mathbf{x} \in \mathcal{E}_2 \setminus \mathcal{E}_1$ be arbitrary and denote $\mathbf{x}' = P_{\mathcal{E}_1}(\mathbf{x})$ and $\mathbf{x}'' = P_{\mathcal{E}_2}(\mathbf{x}')$ (see Fig. 3.6). Without loss of generality assume $\mathbf{x}' \notin \mathcal{E}_2$ (otherwise $\mathbf{x}' = \mathbf{x}''$ resulting zero in the left-hand side and both conclusions are trivial), which implies that $\mathbf{x} \notin \mathcal{B}(\mathbf{x}^{\mathrm{o}}, r)$. Hence $\mathbf{x} \neq \mathbf{x}'$ and $\mathbf{x}' \neq \mathbf{x}''$. Since $\mathbf{x}', \mathbf{x}^{\mathrm{o}} \in \mathcal{E}_1$, the line segment $\overline{\mathbf{x}'\mathbf{x}^{\mathrm{o}}}$ between $\mathbf{x}'$ and $\mathbf{x}^{\mathrm{o}}$ is contained entirely in $\mathcal{E}_1$.

The fact $\mathbf{x}' = P_{\mathcal{E}_1}(\mathbf{x})$ implies that 1) there is a supporting hyperplane W of $\mathcal{E}_1$ that passes through $\mathbf{x}'$ and is orthogonal to $\overline{\mathbf{x}\mathbf{x}'}$, 2) the angle that $\overline{\mathbf{x}'\mathbf{x}^{\mathrm{o}}}$ and $\overline{\mathbf{x}'\mathbf{x}}$ make at $\mathbf{x}'$ is obtuse and thus $\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\| < \|\mathbf{x} - \mathbf{x}^{\mathrm{o}}\|$, and 3) the points $\mathbf{x}$, $\mathbf{x}'$, $\mathbf{x}^{\mathrm{o}}$ constitute a plane W^{o} that is orthogonal to W. Note that $\mathbf{x}$ is on one side of W while the convex hull $\mathcal{C} \subset \mathcal{E}_1$ of the point $\mathbf{x}'$ and the ball $\mathcal{B}(\mathbf{x}^{\mathrm{o}}, r)$ is on the other side.

As shown in Fig. 3.6, let $\overline{\mathbf{x}'\mathbf{z}} \subset W^{\mathrm{o}}$ be the line segment that is tangential to the sphere $\partial\mathcal{B}(\mathbf{x}^{\mathrm{o}}, r)$ at the point $\mathbf{z}$ and intersects $\overline{\mathbf{x}\mathbf{x}^{\mathrm{o}}}$ at a point $\mathbf{y}$, and let $\tilde{\mathbf{x}}''$ be a point on the line segment $\overline{\mathbf{x}\mathbf{x}^{\mathrm{o}}} \subset \mathcal{E}_2$ such that $\overline{\mathbf{x}'\tilde{\mathbf{x}}''} \perp \overline{\mathbf{x}\mathbf{x}^{\mathrm{o}}}$. Then $d_{\mathcal{E}_1}(\mathbf{x}'') \leq \|\mathbf{x}'' - \mathbf{x}'\| \leq \|\mathbf{x}' - \tilde{\mathbf{x}}''\|$, with the two inequalities following from the fact that $\mathbf{x}' \in \mathcal{E}_1$ and $\tilde{\mathbf{x}}'' \in \mathcal{E}_2$ are not necessarily the projection points of $\mathbf{x}''$ onto $\mathcal{E}_1$ and $\mathbf{x}'$ onto $\mathcal{E}_2$, respectively.

The angles between the line segments $\overline{\mathbf{x}'\mathbf{y}}$ and $\overline{\mathbf{x}'\mathbf{x}^{\mathrm{o}}}$, $\overline{\mathbf{x}'\tilde{\mathbf{x}}''}$ and $\overline{\mathbf{x}'\mathbf{y}}$, $\overline{\mathbf{x}\mathbf{x}'}$ and $\overline{\mathbf{x}\tilde{\mathbf{x}}''}$, $\overline{\mathbf{y}\mathbf{x}'}$ and $\overline{\mathbf{y}\tilde{\mathbf{x}}''}$, are denoted by η^{o}, η'', $\eta_{\mathbf{x}}$, $\eta_{\mathbf{y}}$, respectively. Obviously, $\sin(\eta^{\mathrm{o}}) = r/\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\|$, $\eta'' \leq 90^\circ - \eta^{\mathrm{o}}$ and $\eta_{\mathbf{y}} \geq \eta^{\mathrm{o}}$.

By the geometric relationship in the plane W^{o}, we have

$$
\begin{aligned}
& d_{\mathcal{E}_1 \cap \mathcal{E}_2}(\mathbf{x}) \\
& \quad \leq \|\mathbf{x} - \mathbf{y}\| = \|\mathbf{x} - \tilde{\mathbf{x}}''\| + \|\tilde{\mathbf{x}}'' - \mathbf{y}\| = \|\mathbf{x} - \mathbf{x}'\| \cos(\eta_{\mathbf{x}}) + \|\mathbf{x} - \mathbf{x}'\| \sin(\eta_{\mathbf{x}}) \tan(\eta'') \\
& \quad \leq \|\mathbf{x} - \mathbf{x}'\| \cos(\eta_{\mathbf{x}}) + \|\mathbf{x} - \mathbf{x}'\| \sin(\eta_{\mathbf{x}}) \tan(90^\circ - \eta^{\mathrm{o}}) = \|\mathbf{x} - \mathbf{x}'\| \frac{\sin(\eta_{\mathbf{x}} + \eta^{\mathrm{o}})}{\sin(\eta^{\mathrm{o}})} \\
& \quad \leq \|\mathbf{x} - \mathbf{x}'\| / \sin(\eta^{\mathrm{o}}) = d_{\mathcal{E}_1}(\mathbf{x}) \frac{\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\|}{r} \leq d_{\mathcal{E}_1}(\mathbf{x}) \frac{\|\mathbf{x} - \mathbf{x}^{\mathrm{o}}\|}{r},
\end{aligned}
$$

which is the desired conclusion (a).

Since the angle that $\overline{\mathbf{x}'\mathbf{x}^{\mathrm{o}}}$ and $\overline{\mathbf{x}'\mathbf{x}}$ make at $\mathbf{x}'$ can not be acute, $\eta_{\mathbf{x}}+\eta_{\mathbf{y}} \leq 90^\circ$ holds, implying that $0 \leq \eta_{\mathbf{x}} \leq 90^\circ - \eta_{\mathbf{y}} \leq 90^\circ - \eta^{\mathrm{o}} \leq 90^\circ$. Then $\sin(\eta_{\mathbf{x}}) \leq \sin(90^\circ - \eta^{\mathrm{o}}) = \cos(\eta^{\mathrm{o}}) = \sqrt{\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\|^2 - r^2}/\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\|$. Therefore,

$$\frac{d_{\mathcal{E}_1}(\mathbf{x}'')}{d_{\mathcal{E}_1}(\mathbf{x})} \leq \frac{\|\mathbf{x}' - \tilde{\mathbf{x}}''\|}{\|\mathbf{x}' - \mathbf{x}\|} = \sin(\eta_x) \leq \frac{\sqrt{\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\|^2 - r^2}}{\|\mathbf{x}' - \mathbf{x}^{\mathrm{o}}\|} \leq \frac{\sqrt{\|\mathbf{x} - \mathbf{x}^{\mathrm{o}}\|^2 - r^2}}{\|\mathbf{x} - \mathbf{x}^{\mathrm{o}}\|} = \gamma.$$

Combined with the trivial case that $\mathbf{x}' = \mathbf{x}''$ when $\mathbf{x} \in \mathcal{B}(\mathbf{x}^{\mathrm{o}}, r)$, the conclusion (b) is proved. ■

Using Lemma 3.5.1, we are ready to prove the exponential convergence rate of Algorithm 1.

Proof [Theorem 3.4.2]

As shown in the proof of Theorem 3.4.1, given $\alpha_i = 1, \forall i \in \mathcal{I}_m$, the sequence $\{\mathbf{x}^t\}$ generated by Algorithm 1 satisfies the condition that $\mathbf{x}^t \in \mathcal{A}_2$ and $\mathbf{x}^{t+1} = P_{\mathcal{A}_2}(P_{\mathcal{A}_1}(\mathbf{x}^t))$, $\forall t = 0, 1, \ldots$, with $\mathcal{A}_1$ and $\mathcal{A}_2$ defined in (3.10), and that $\lim_{t\to\infty} \mathbf{x}^t = \mathbf{x}^* \in \mathcal{A}_1 \cap \mathcal{A}_2$ as a consequence.

By our assumption on x^{o} and setting $x^{\mathrm{o}}_{ik} = x^{\mathrm{o}}_i, \forall k \in \mathcal{N}_i^-$, the corresponding $\mathbf{x}^{\mathrm{o}}$ has the properties that $\mathbf{x}^{\mathrm{o}} \in \mathcal{A}_2$ by construction, and each $\mathbf{x}^{\mathrm{o}}_i \in \mathcal{F}^{\mathrm{o}}_i$ which leads to the fact that $\mathbf{x}^{\mathrm{o}}$ is an interior point of $\mathcal{A}_1$. Thus by combining two conclusions in Lemma 3.5.1 and the fact that $\|\mathbf{x}^t - \mathbf{x}^{\mathrm{o}}\|$ is nonincreasing and hence bounded, $d_{\mathcal{A}_1\cap\mathcal{A}_2}(\mathbf{x}^t)$ decays to zero exponentially fast.

Let $t \geq 0$ be arbitrary and denote $\mathbf{y}^t := \mathcal{P}_{\mathcal{A}_1\cap\mathcal{A}_2}(\mathbf{x}^t)$. Since $\mathbf{y}^t \in \mathcal{A}_1 \cap \mathcal{A}_2$, $\|\mathbf{x}^{t+s} - \mathbf{y}^t\|$ is nonincreasing in s for $s \geq 0$, resulted from the facts that $\mathbf{x}^{t+s+1} = P_{\mathcal{A}_2}(P_{\mathcal{A}_1}(\mathbf{x}^{t+s}))$ and that both $P_{\mathcal{A}_1}$ and $P_{\mathcal{A}_2}$ are paracontractions with $\mathbf{y}^t$ being one of their fixed points. Thus $\|\mathbf{x}^t - \mathbf{y}^t\| \geq \lim_{s\to\infty} \|\mathbf{x}^{t+s} - \mathbf{y}^t\| = \|\mathbf{x}^* - \mathbf{y}^t\|$, which leads to

$$\|\mathbf{x}^t - \mathbf{x}^*\| \leq \|\mathbf{x}^t - \mathbf{y}^t\| + \|\mathbf{x}^* - \mathbf{y}^t\| \leq 2\|\mathbf{x}^t - \mathbf{y}^t\| = 2d_{\mathcal{A}_1\cap\mathcal{A}_2}(\mathbf{x}^t).$$

Therefore, $\mathbf{x}^t$ converges to $\mathbf{x}^*$ exponentially fast. ■

3.5.3 Convergence Proof of Asynchronous Algorithm

Similarly to Algorithm 1, we will show in the following that the augmented variable $\mathbf{x}^t$ is updated at each round of Algorithm 2 by the composition of two paracontractions.

Proof [Theorem 3.4.3]

Under Assumption 3.4.1, the update at round t can be written as

$$\mathbf{x}^{t+1} = Q^t \circ P^t \left(\mathbf{x}^t\right).$$

Here, the operator P^t is defined as

$$P^t = P_1^t \times \cdots \times P_m^t, \tag{3.19}$$

where $P_i^t : \mathbb{R}^{N_i} \to \mathbb{R}^{N_i}$ is the P_i defined in (3.11) if agent i performs the relaxed projection at this round, and the identity map if otherwise. The operator Q^t is defined as

$$Q^t = M^\top \tilde{Q}^t M$$

where M is the same permutation matrix defined in (3.10) and

$$\tilde{Q}^t = \tilde{Q}_1^t \times \cdots \times \tilde{Q}_m^t$$

with each $\tilde{Q}_i^t : \mathbb{R}^{n_i(1+|\mathcal{N}_i^-|)} \to \mathbb{R}^{n_i(1+|\mathcal{N}_i^-|)}$ being

$$\tilde{Q}_i^t = (M_i^t)^\top \left[Q_i^t, \cdots, Q_i^t, I_{n_i}, \cdots, I_{n_i}\right] M_i^t, \tag{3.20}$$

if agent i performs the partial consensus, and the identity map if otherwise. In (3.20), M_i^t is a permutation matrix that puts the agents in $\{i\} \cup \mathcal{N}_{i,t}^-$ at the front of the group $\{i\} \cup \mathcal{N}_i^-$; with a little abuse of notation, Q_i^t being the matrix $\frac{1}{|\mathcal{N}_{i,t}^-|+1}\mathbf{1}^\top \otimes I_{n_i}$ corresponding to the operation (3.14) appears $|\mathcal{N}_{i,t}^-|+1$ times. By repeatedly applying Lemmas 2.2.1 and 2.2.2, we know P^t and Q^t are paracontractions w.r.t. the Euclidean norm.

It is easy to see that the number of possible operators P^t and Q^t for $t = 0, 1, \ldots,$ is finite. Moreover, P^t's and Q^t's have the common fixed point sets $\mathcal{A}_1$ and $\mathcal{A}_2$ defined in (3.10), respectively. Hence, under Assumption 3.4.2, $\mathbf{x}^t$ will converge to some $\mathbf{x}^* \in \mathcal{A}_1 \cap \mathcal{A}_2$ as a consequence of Theorem 2.2.4. ■

The convergence of Algorithm 2b can be proven similarly.

Proof [Theorem 3.4.4]

Since the matrix $W_{ik} \in \mathbb{R}^{2\times 2}$ used in the update (3.15) is doubly stochastic, it can be explicitly expressed as

$$W_{ik} = \begin{bmatrix} 1-\beta_{ik} & \beta_{ik} \\ \beta_{ik} & 1-\beta_{ik} \end{bmatrix}$$

with $\beta_{ik} \in (0, 1)$. The corresponding update is

$$\begin{bmatrix} x_i^{t+1} \\ x_{ik}^{t+1} \end{bmatrix} = (1-2\beta_{ik}) \begin{bmatrix} x_i^t \\ x_{ik}^t \end{bmatrix} + 2\beta_{ik} \begin{bmatrix} (x_i^t + x_{ik}^t)/2 \\ (x_i^t + x_{ik}^t)/2 \end{bmatrix},$$

which is a $(2\beta_{ik})$-relaxed projection onto the consensus set $\{(x_i, x_{ik}) | x_i = x_{ik}\}$, and therefore a paracontraction w.r.t. the Euclidean norm. The remaining proof is exactly the same as that of Theorem 3.4.3. ■

In general, the convergence of Algorithms 2 and 2b is not exponential.

3.5.4 Convergence Proof of Generalized Synchronous Algorithm

The convergence analysis of Algorithm 3 is more challenging than Algorithms 1 and 2 because the operation in (3.17) is no longer a paracontraction. Instead, our proof will utilize the property of strongly quasi-nonexpansive maps.

For any agent $i \in \mathcal{I}_m$, its augmented variable $\mathbf{x}_i$'s update at round t of Algorithm 3 can be summarized as

$$\mathbf{z}_i^t = P_i^t(\mathbf{x}_i^t) \tag{3.21}$$

$$\mathbf{x}_i^{t+1} = \sum_{k\in\mathcal{I}_m} \left(Q^t\right)_{ik} \mathbf{z}_k^t, \tag{3.22}$$

where

$$P_i^t = \begin{cases} P_i, & \text{if } i \in \mathcal{I}_P^t, \\ \text{Id}, & \text{otherwise;} \end{cases} \tag{3.23}$$

and Q^t is a doubly stochastic matrix whose block in the i-th row and k-th column, denoted as $(Q^t)_{ik}, \forall i, k \in \mathcal{I}_m$, will be defined shortly. In sum, the dynamics are

$$\mathbf{z}^t = P^t(\mathbf{x}^t) \tag{3.24}$$

$$\mathbf{x}^{t+1} = Q^t \mathbf{z}^t. \tag{3.25}$$

To define Q^t, we first reorder $\mathbf{x}$ and $\mathbf{z}$ as $\tilde{\mathbf{x}} = M\mathbf{x}$ and $\tilde{\mathbf{z}} = M\mathbf{z}$, respectively, using the same permutation matrix M in (3.10). Then the consensus step (3.17) will result in

$$\tilde{\mathbf{x}}^{t+1} = W^t \tilde{\mathbf{z}}^t, \tag{3.26}$$

with $W^t := \text{diag}\left(W_1^t \otimes I_{n_1}, W_2^t \otimes I_{n_2}, \cdots, W_m^t \otimes I_{n_m}\right)$. If $i \notin \mathcal{I}_Q^t$, i.e., agent i is not activated to perform the consensus update (3.17) at round t, W_i^t is set to be the identity map Id. For $\mathbf{x}$, the following holds

$$\mathbf{x}^{t+1} = M^\top \tilde{\mathbf{x}}^{t+1} = M^\top W^t \tilde{\mathbf{z}}^t = M^\top W^t M \mathbf{z}^t = Q^t \mathbf{z}^t,$$

under the definition $Q^t := M^\top W^t M$. As a consequence of Assumption 3.4.3(a), W^t, and hence Q^t, is doubly stochastic.

With the dynamics (3.24) and (3.25), we next establish the convergence by showing first the intermediate values $\mathbf{z}^t$ will converge to a point in $\mathcal{A}_1 \cap \mathcal{A}_2$, i.e., a solution to Problem 3.2.

To proceed, define agent i's displacement vector as

$$\mathbf{e}_i^t := P_i^t(\mathbf{x}_i^t) - \mathbf{x}_i^t = \mathbf{z}_i^t - \mathbf{x}_i^t, \tag{3.27}$$

with P_i^t in (3.23). Note that if $P_i^t = P_i$, then

$$\|\mathbf{e}_i^t\| = \|(1-\alpha_i)\mathbf{x}_i^t + \alpha_i P_{\mathcal{F}_i}\left(\mathbf{x}_i^t\right) - \mathbf{x}_i^t\| = \alpha_i \|P_{\mathcal{F}_i}\left(\mathbf{x}_i^t\right) - \mathbf{x}_i^t\| = \alpha_i \, d_{\mathcal{F}_i}\left(\mathbf{x}_i^t\right).$$

The following lemma shows that $\mathbf{e}_i^t$ will converge to zero.

Lemma 3.5.2 *Suppose Assumption 3.4.3 holds. As the iteration index $t \to \infty$, the displacement vector $\mathbf{e}^t := (\mathbf{e}_i^t)_{i\in\mathcal{I}_m} \to 0$.*

Proof Let $\mathbf{y} \in \mathcal{A}_1 \cap \mathcal{A}_2$ be a solution of Problem 3.2. Then for all $t = 0, 1, \ldots$, $\mathbf{y} = Q^t\mathbf{y}$ because the matrix Q^t is stochastic. As discussed at the beginning of this section, P_i defined in (3.11) is a β_i-strongly quasi-nonexpansive map with $\beta_i := \frac{2-\alpha_i}{\alpha_i}$, i.e.,

$$\|P_i(\mathbf{x}_i^t) - \mathbf{y}_i\|^2 \leq \|\mathbf{x}_i^t - \mathbf{y}_i\|^2 - \beta_i\|P_i(\mathbf{x}_i^t) - \mathbf{x}_i^t\|^2.$$

When replacing P_i by the identity map Id, the above inequality still holds for the same β_i. It then follows that

$$\|\mathbf{z}_i^t - \mathbf{y}_i\|^2 \leq \|\mathbf{x}_i^t - \mathbf{y}_i\|^2 - \underline{\beta}\|\mathbf{e}_i^t\|^2,$$

with $\underline{\beta} := \min\limits_{i\in\mathcal{I}_m}\beta_i = \min\limits_{i\in\mathcal{I}_m}\left\{\frac{2-\alpha_i}{\alpha_i}\right\}$. Combining with (3.22) we have

$$\underline{\beta}\|\mathbf{e}_i^t\|^2 \leq \left\|\sum_{k\in\mathcal{I}_m}(Q^{t-1})_{ik}\mathbf{z}_k^{t-1} - \mathbf{y}_i\right\|^2 - \left\|\mathbf{z}_i^t - \mathbf{y}_i\right\|^2. \tag{3.28}$$

Define an element-wise convex map $\Gamma : \mathbb{R}^N \to \mathbb{R}^N$ such that $\Gamma(x)_l = x_l^2$, $l \in \mathcal{I}_N$. Then (3.28) will lead to

$$\begin{aligned}
\underline{\beta}\|\mathbf{e}^t\|^2 = \sum_{i\in\mathcal{I}_m}\underline{\beta}\|\mathbf{e}_i^t\|^2 &\leq \mathbf{1}^\top\Big\{\Gamma\big(Q^{t-1}\mathbf{z}^{t-1} - \mathbf{y}\big) - \Gamma\big(\mathbf{z}^t - \mathbf{y}\big)\Big\} \\
&= \mathbf{1}^\top\Big\{\Gamma\left(Q^{t-1}\left(\mathbf{z}^{t-1} - \mathbf{y}\right)\right) - \Gamma\big(\mathbf{z}^t - \mathbf{y}\big)\Big\} \leq \mathbf{1}^\top\Big\{Q^{t-1}\Gamma\big(\mathbf{z}^{t-1} - \mathbf{y}\big) - \Gamma\big(\mathbf{z}^t - \mathbf{y}\big)\Big\} \\
&= \mathbf{1}^\top\Gamma\big(\mathbf{z}^{t-1} - \mathbf{y}\big) - \mathbf{1}^\top\Gamma\big(\mathbf{z}^t - \mathbf{y}\big) = \|\mathbf{z}^{t-1} - \mathbf{y}\|^2 - \|\mathbf{z}^t - \mathbf{y}\|^2,
\end{aligned} \tag{3.29}$$

where the second inequality and the third equality follow from the convexity of Γ and the doubly stochasticity of Q^{t-1}, respectively. For any time instant $\bar{t} > 0$, summing the above inequalities for $t = 1, \ldots, \bar{t}$ will result in

$$\sum_{t=1}^{\bar{t}}\underline{\beta}\|\mathbf{e}^t\|^2 \leq \|\mathbf{z}^0 - \mathbf{y}\|^2 - \|\mathbf{z}^{\bar{t}} - \mathbf{y}\|^2 \leq \|\mathbf{z}^0 - \mathbf{y}\|^2.$$

Since the inequality holds for arbitrarily large $\bar{t}$, we have

$$\sum_{t=1}^{\infty} \underline{\beta} \|\mathbf{e}^t\|^2 \leq \|\mathbf{z}^0 - \mathbf{y}\|^2 < \infty.$$

With $\underline{\beta} > 0$, this directly leads to $\lim_{t\to\infty} \mathbf{e}^t = 0$. ■

With Lemma 3.5.2, we next show that $\mathbf{z}^t$ will asymptotically satisfy the consensus constraint in (3.8) as $t \to \infty$.

Lemma 3.5.3 *Suppose Assumptions 3.4.3 and 3.4.4 hold. Then* $\forall i \in \mathcal{I}_m$ *and* $\forall k \in \mathcal{N}_i^-$, $\lim_{t\to\infty} \|z_i^t - z_{ik}^t\| = 0$.

Proof In this proof we focus on the reordered variables $\tilde{\mathbf{x}}_i = \left(x_i, (x_{ik})_{k\in\mathcal{N}_i^-}\right)$ and $\tilde{\mathbf{z}}_i = \left(z_i, (z_{ik})_{k\in\mathcal{N}_i^-}\right)$ for an arbitrary $i \in \mathcal{I}_m$, whose dynamics according to (3.17) can be written as $\tilde{\mathbf{x}}_i^{t+1} = (W_i^t \otimes I_{n_i})\, \tilde{\mathbf{z}}_i^t$. Combined with the fact $\tilde{\mathbf{z}}_i^{t+1} = \tilde{\mathbf{x}}_i^{t+1} + \tilde{\mathbf{e}}_i^{t+1}$, it follows that

$$\tilde{\mathbf{z}}_i^{t+1} = \left(W_i^t \otimes I_{n_i}\right) \tilde{\mathbf{z}}_i^t + \tilde{\mathbf{e}}_i^{t+1}.$$

For $s \leq t$, repeatedly applying the above equation yields

$$\tilde{\mathbf{z}}_i^{t+1} = \left(\Phi_i^{t,s} \otimes I_{n_i}\right) \tilde{\mathbf{z}}_i^s + \sum_{r=s+1}^{t} \left(\Phi_i^{t,r} \otimes I_{n_i}\right) \tilde{\mathbf{e}}_i^r + \tilde{\mathbf{e}}_i^{t+1}, \tag{3.30}$$

where $\Phi_i^{t,s} := W_i^t W_i^{t-1} \cdots W_i^{s+1} W_i^s$ and $\Phi_i^{t,s} = W_i^t$ when $s = t$. Obviously, $\Phi_i^{t,s}$ is doubly stochastic under Assumption 3.4.3(a).

Under Assumption 3.4.4(b), the sequence of graphs $\{\mathcal{G}_i^t\}$ associated with the matrix sequence $\{W_i^t\}$ is repeatedly jointly strongly connected (see Section 2.6 for the definitions). Together with Assumption 3.4.3 on W_i^t, this implies that, for any fixed s, every entry of $\Phi_h^{t,s}$ will converge to $1/\left(1 + |\mathcal{N}_i^-|\right)$ exponentially fast as $t \to \infty$ as shown by [27, Prop. 1]. More precisely,

$$\left| [\Phi_i^{t,s}]_{kl} - \frac{1}{1 + |\mathcal{N}_i^-|} \right| \leq c\lambda^{t-s}$$

for all $k, l \in \mathcal{I}_{1+|\mathcal{N}_i^-|}$. Here, the constants $c > 0$ and $\lambda \in [0, 1)$ are determined by the cardinality of $\mathcal{N}_i^-$, $\underline{w}$ in Assumption 3.4.3(b) and T from Assumption 3.4.4(b) (see [27, Prop. 1]).

Following (3.30), for $k \in \mathcal{I}_{1+|\mathcal{N}_i^-|}$, the k-th subvector in $\tilde{\mathbf{z}}_i^{t+1}$ (namely, z_i^{t+1} and its desired values by its out-neighbors) is given by

$$[\tilde{\mathbf{z}}_i^{t+1}]_k = \sum_{l=1}^{1+|\mathcal{N}_i^-|} [\Phi_i^{t,s}]_{kl}[\tilde{\mathbf{z}}_i^s]_l + \sum_{r=s+1}^{t} \sum_{l=1}^{1+|\mathcal{N}_i^-|} [\Phi_i^{t,r}]_{kl}[\tilde{\mathbf{e}}_i^r]_l + [\tilde{\mathbf{e}}_i^{t+1}]_l.$$

Define y_i^{t+1} to be the average of z_i^{t+1} and its desired values from out-neighbors. Then

$$\begin{aligned} y_i^{t+1} &= \frac{1}{1+|\mathcal{N}_i^-|} \left(\mathbf{1}^\top \otimes I_{n_i}\right) \tilde{\mathbf{z}}_i^{t+1} \\ &= \frac{1}{1+|\mathcal{N}_i^-|} \Big\{ \left(\mathbf{1}^\top \otimes I_{n_i}\right) \tilde{\mathbf{z}}_i^s + \sum_{r=s+1}^{t} \left(\mathbf{1}^\top \otimes I_{n_i}\right) \tilde{\mathbf{e}}_i^r + \left(\mathbf{1}^\top \otimes I_{n_i}\right) \tilde{\mathbf{e}}_i^{t+1} \Big\} \\ &= \frac{1}{1+|\mathcal{N}_i^-|} \Big\{ \sum_{l=1}^{1+|\mathcal{N}_i^-|} [\tilde{\mathbf{z}}_i^s]_l + \sum_{r=s+1}^{t} \sum_{l=1}^{1+|\mathcal{N}_i^-|} [\tilde{\mathbf{e}}_i^r]_l + \sum_{l=1}^{1+|\mathcal{N}_i^-|} [\tilde{\mathbf{e}}_i^{t+1}]_l \Big\}. \end{aligned}$$

Note that the second equality follows from (3.30) and the fact that $\Phi_i^{t,s}$ is doubly stochastic. Then for $k \in \mathcal{I}_{1+|\mathcal{N}_i^-|}$, we have

$$\begin{aligned} &\left\| [\tilde{\mathbf{z}}_i^{t+1}]_k - y_i^{t+1} \right\|_1 \\ &= \Bigg\| \sum_{l=1}^{1+|\mathcal{N}_i^-|} \Big([\Phi_i^{t,s}]_{kl} - \frac{1}{1+|\mathcal{N}_i^-|} \Big) [\tilde{\mathbf{z}}_i^s]_l + \sum_{r=s+1}^{t} \sum_{l=1}^{1+|\mathcal{N}_i^-|} \Big([\Phi_i^{t,r}]_{kl} - \frac{1}{1+|\mathcal{N}_i^-|} \Big) [\tilde{\mathbf{e}}_i^r]_l \\ &\qquad + [\tilde{\mathbf{e}}_i^{t+1}]_k - \frac{1}{1+|\mathcal{N}_i^-|} \sum_{l=1}^{1+|\mathcal{N}_i^-|} [\tilde{\mathbf{e}}_i^{t+1}]_l \Bigg\|_1 \\ &\leq c\lambda^{t-s} \|\tilde{\mathbf{z}}_i^s\|_1 + \sum_{r=s+1}^{t} c\lambda^{t-r} \|\tilde{\mathbf{e}}_i^r\|_1 + \|\tilde{\mathbf{e}}_i^{t+1}\|_1 \\ &\leq cc_1\lambda^{t-s} \|\tilde{\mathbf{z}}_i^s\|_2 + \sum_{r=s+1}^{t} cc_1\lambda^{t-r} \|\tilde{\mathbf{e}}_i^r\|_2 + c_1 \|\tilde{\mathbf{e}}_i^{t+1}\|_2 \end{aligned}$$

where $c_1 = \sqrt{1+|\mathcal{N}_i^-|}$, following from the fact $\|x\|_1 \leq \sqrt{n}\|x\|_2$, $x \in \mathbb{R}^n$. By Lemma 3.5.2, $\forall \epsilon > 0$, there exists an s such that for all $t \geq s$, $\|\tilde{\mathbf{e}}_i^t\| < \epsilon$. This leads to

$$\left\| [\tilde{\mathbf{z}}_i^{t+1}]_k - y_i^{t+1} \right\|_1 \leq cc_1\lambda^{t-s} \|\tilde{\mathbf{z}}_i^s\|_2 + cc_1\epsilon \frac{1-\lambda^{t-s}}{1-\lambda} + c_1\epsilon.$$

Since ϵ can be arbitrarily small, the following holds

$$\lim_{t\to\infty} \left\| [\tilde{\mathbf{z}}_i^{t+1}]_k - y_i^{t+1} \right\|_1 = 0,$$

i.e., z_i^t and z_{ik}^t for $k \in \mathcal{N}_i^-$ reach consensus asymptotically. ■

Finally, we establish the convergence of the generalized synchronous algorithm stated in Theorem 3.4.5.

Proof [Theorem 3.4.5]

From (3.29) we have

$$\|\mathbf{z}^t - \mathbf{y}\|^2 \leq \|\mathbf{z}^{t-1} - \mathbf{y}\|^2. \tag{3.31}$$

Thus the sequence $\left\{\|\mathbf{z}^t - \mathbf{y}\|^2\right\}$ is non-increasing for any $\mathbf{y} \in \mathcal{A}_1 \cap \mathcal{A}_2$. In particular, this implies that the sequence $\{\mathbf{z}^t\}$ is bounded and has accumulation points.

Next we are going to prove the accumulation point is unique. Let $\mathbf{z}^*$ be a point that $\mathbf{z}^t$ converges to along the time subsequence $\{t_s\}$. As a consequence of Lemma 3.5.3, $\mathbf{z}^* \in \mathcal{A}_2$.

In the first case, assume $\mathbf{z}^* \in \mathcal{A}_1$, implying that $\mathbf{z}^* \in \mathcal{A}_1 \cap \mathcal{A}_2$. Let $\hat{\mathbf{z}}^* \neq \mathbf{z}^*$ be a distinct accumulation point that $\{\mathbf{z}^t\}$ converges to along the time subsequence $\{\hat{t}_s\}$. Without loss of generality, assume $\hat{t}_s > t_s$ for all s. Then by replacing $\mathbf{y}$ in (3.31) with $\mathbf{z}^*$, we have $\|\mathbf{z}^{\hat{t}_s} - \mathbf{z}^*\|^2 \leq \|\mathbf{z}^{t_s} - \mathbf{z}^*\|^2$. As $s \to \infty$, we have $\|\hat{\mathbf{z}}^* - \mathbf{z}^*\|^2 \leq 0$, which contradicts the assumption that $\hat{\mathbf{z}}^* \neq \mathbf{z}^*$. Therefore there is only one accumulation point.

In the second case, assume $\mathbf{z}^* \notin \mathcal{A}_1$. Then there exists an integer $r \in \mathcal{I}_m$ such that $\mathbf{z}^*$ violates a total of r out of the m feasibility constraints in (3.7). Without loss of generality, the first r constraints are assumed to be violated, i.e., $d_{\mathcal{F}_i}(\mathbf{z}_i^*) > 0$ for all $i \in \mathcal{I}_r$. Pick any δ such that $0 < \delta \leq \min_{i \in \mathcal{I}_r} \; d_{\mathcal{F}_i}(\mathbf{z}_i^*)$. Then as a consequence of Lemma 3.5.2, there exists a large enough integer $K > 0$ such that for all $t \geq K$, $\|\mathbf{e}^t\| \leq \delta\underline{\alpha}/8$ with $\underline{\alpha} := \min_{i \in \mathcal{I}_m} \alpha_i$. Suppose at time $t_1 \geq K$, $\mathbf{z}^{t_1} \in \mathcal{B}(\mathbf{z}^*, \delta/4)$. This implies that for all $i \in \mathcal{I}_r$,

$$d_{\mathcal{F}_i}(\mathbf{z}_i^{t_1}) \geq d_{\mathcal{F}_i}(\mathbf{z}_i^*) - \|\mathbf{z}_i^{t_1} - \mathbf{z}_i^*\| \geq \delta - \frac{\delta}{4} = \frac{3\delta}{4}.$$

At the same time, the next iteration value $\mathbf{x}^{t_1+1}$ satisfies

$$\begin{aligned}\|\mathbf{x}^{t_1+1}-\mathbf{z}^{t_1}\| &= \|Q^{t_1}\mathbf{z}^{t_1}-Q^{t_1}\mathbf{z}^*+\mathbf{z}^*-\mathbf{z}^{t_1}\| \le \|Q^{t_1}-I\|\|\mathbf{z}^{t_1}-\mathbf{z}^*\| \\ &\le (\|Q^{t_1}\|+1)\|\mathbf{z}^{t_1}-\mathbf{z}^*\| = 2\|\mathbf{z}^{t_1}-\mathbf{z}^*\| \le \delta/2.\end{aligned}$$

Here, we use the fact that $\|Q^{t_1}\|=1$ for the doubly stochastic matrix Q^{t_1}. Combining the two results above, we obtain $\forall i \in \mathcal{I}_r$,

$$d_{\mathcal{F}_i}\left(\mathbf{x}_i^{t_1+1}\right) \ge d_{\mathcal{F}_i}\left(\mathbf{z}_i^{t_1}\right) - \|\mathbf{x}_i^{t_1+1}-\mathbf{z}_i^{t_1}\| \ge \frac{3\delta}{4}-\frac{\delta}{2}=\frac{\delta}{4}.$$

In the next relaxed projection update, if $i \in \mathcal{I}_r \cap \mathcal{I}_P^{t_1+1}$, i.e., agent $i \in \mathcal{I}_r$ is activated to carry out projection at round t_1+1, the resulted displacement vector $\mathbf{e}^{t_1+1}$ satisfies

$$\|\mathbf{e}^{t_1+1}\| \ge \|\mathbf{e}_i^{t_1+1}\| = \alpha_i d_{\mathcal{F}_i}\left(\mathbf{x}_i^{t_1+1}\right) \ge \frac{\underline{\alpha}\delta}{4},\ i \in \mathcal{I}_r,$$

which contradicts the previous assumption that $\|\mathbf{e}^t\| \le \delta\underline{\alpha}/8$ for any $t \ge K$. Therefore, we must have $\mathcal{I}_r \cap \mathcal{I}_P^{t_1+1} = \emptyset$. This implies that the iteration from $\mathbf{z}^{t_1}$ to $\mathbf{z}^{t_1+1}$ is through the operator $P^{t_1+1} \circ Q^{t_1}$ where P^{t_1+1} satisfies that $P_i^{t_1+1} = \mathrm{Id}$ for $i \in \mathcal{I}_r$. Equivalently, we can view this step as one iteration of Algorithm 3 applied to a new problem, which is the same as Problem 3.2 except that the feasible sets $\mathcal{F}_1,\ldots,\mathcal{F}_r$ are relaxed to be the entire spaces of proper dimensions while $\mathcal{F}_{r+1},\ldots,\mathcal{F}_m$ remain unchanged. Since $\mathbf{z}^*$ is in the consensus subspace $\mathcal{A}_2$ and satisfies the constraints $\mathcal{F}_{r+1},\ldots,\mathcal{F}_m$, it is a solution to the relaxed problem. By following the same arguments we used previously to derive (3.29), we can show that

$$\|\mathbf{z}^{t_1+1}-\mathbf{z}^*\| \le \|\mathbf{z}^{t_1}-\mathbf{z}^*\| \le \delta/4.$$

In other words, $\mathbf{z}^{t_1+1} \in \mathcal{B}(\mathbf{z}^*, \delta/4)$. By repeating the above steps and induction, we conclude that the sequence $\{\mathbf{z}^t\}$ will stay inside the closed ball $\mathcal{B}(\mathbf{z}^*,\delta/4)$ for all $t \ge t_1$. Since the choice of $\delta > 0$ can be arbitrarily small, there will be no other accumulation points besides $\mathbf{z}^*$.

In summary, the accumulation point of $\{\mathbf{z}^t\}$ is unique, i.e., $\lim_{t\to\infty}\mathbf{z}^t = \mathbf{z}^*$. Also $\lim\limits_{t\to\infty}\mathbf{x}^t = \mathbf{z}^*$ holds based on the facts that $\mathbf{x}^t = \mathbf{z}^t - \mathbf{e}^t$ from (3.27) and $\lim\limits_{t\to\infty}\mathbf{e}^t = 0$ in Lemma 3.5.2.

Now we show that $\mathbf{z}^* \in \mathcal{A}_1$. With P_i^t defined in (3.23), under Assumption 3.4.4(a), let $\{\tau\}$ be the subsequence of $\{t\}$ that $P_i^\tau = P_i$. Then $\{\mathbf{x}_i^\tau\}$ and $\{\mathbf{e}_i^\tau\}$ are subsequences of $\{\mathbf{x}_i^t\}$ and $\{\mathbf{e}_i^t\}$, respectively. Since $\lim_{t\to\infty} \mathbf{x}_i^t = \mathbf{z}_i^*$, $\forall i \in \mathcal{I}_m$, we have

$$d_{\mathcal{F}_i}(\mathbf{z}_i^*) = \lim_{\tau\to\infty} d_{\mathcal{F}_i}(\mathbf{x}_i^\tau) = (1/\alpha_i) \lim_{\tau\to\infty} \|\mathbf{e}_i^\tau\| = 0,$$

where the second equality follows from the argument after (3.27). Therefore, $\mathbf{z}_i^* \in \mathcal{F}_i$, $\forall i \in \mathcal{I}_m$, or equivalently, $\mathbf{z}^* \in \mathcal{A}_1$. This completes the proof. ■

3.5.5 Convergence Proof of Generalized Asynchronous Algorithm

Under Assumption 3.4.1, the relaxed projection and generalized partial consensus operations operate on two disjoint sets of variables. Therefore the operation at round t can be split into two steps:

$$\mathbf{z}^t = P^t(\mathbf{x}^t),$$
$$\mathbf{x}^{t+1} = Q^t \mathbf{z}^t,$$

where $\mathbf{z}^t$ denotes the intermediate value resulted from all the relaxed projections of this round, $P^t = P_1^t \times \cdots \times P_m^t$ with $P_i^t = P_i$ defined in (3.11) if agent i performs the relaxed projection at this round and $P_i^t = \mathrm{Id}$ if otherwise, and Q^t is the same stochastic matrix defined in (3.22).

Obviously, the augmented variable $\mathbf{x}^t$'s dynamics is identical to that of Algorithm 3. Under Assumptions 3.4.3 and 3.4.4, the convergence proof of Algorithm 4 is the same as that of Algorithm 3 and therefore omitted here.

4. CONVEX OPTIMIZATION PROBLEMS WITH LOCAL COUPLINGS

In Chapter 3, we focus on the convex feasibility problems with locally coupled constraints and four distributed algorithms requiring only information from intermediate neighbors are proposed. In this chapter, these distributed solutions are extended to solve convex optimization problems with the local couplings resulted from not only constraints but also objective functions. To incorporate this difference, the dependency graph is redefined and the algorithms are modified by replacing the projection step with the proximal operator while, however, the convergence proofs are established through operator splitting methods rather than paracontractions.

4.1 Problem Formulation

Consider a group of m agents indexed by $i \in \mathcal{I}_m$. Each agent i has a local variable $x_i \in \mathbb{R}^{n_i}$ (which could be null) as well as a local objective function[1] f_i (which could also be null) that depends on not only x_i but also its neighboring agents' variables. This dependency of local objective functions (with local constraints included through indicator functions) is modeled by a directed graph called the *dependency graph* $(\mathcal{I}_m, \mathcal{E})$ with the vertex set $\mathcal{I}_m$ and the edge set $\mathcal{E} \subset \mathcal{I}_m \times \mathcal{I}_m$ so that an edge $(j, i) \in \mathcal{E}$ indicates that the objective function f_i of agent i depends on the variable x_j of agent j. Denote by $\mathcal{N}_i^+ := \{j \in \mathcal{I}_m \,|\, (j, i) \in \mathcal{E}\}$ and $\mathcal{N}_i^- := \{j \in \mathcal{I}_m \,|\, (i, j) \in \mathcal{E}\}$ the sets of in-neighbors and out-neighbors, respectively, and by $\mathcal{N}_i = \mathcal{N}_i^+ \cup \mathcal{N}_i^-$ the set of neighbors, all of agent i. Then, the objective function of agent i is $f_i(\check{x}_i)$, where $\check{x}_i := \left(x_i, (x_j)_{j \in \mathcal{N}_i^+}\right)$ has the dimension $N_i = n_i + \sum_{j \in \mathcal{N}_i^+} n_j$.

[1] We assume that all the local constraints of agent i have been incorporated into f_i using convex indicator functions.

Assumption 4.1.1 *Each $f_i : \mathbb{R}^{N_i} \to \overline{\mathbb{R}}$ is an extended-real-valued closed convex proper (CCP) function, i.e., f_i is lower semi-continuous, convex, and $f_i \not\equiv +\infty$.*

Denote by $x := (x_i)_{i\in\mathcal{I}_m}$ the concatenated vector of all x_i's. Then $x \in \mathbb{R}^n$ where $n = \sum_{i\in\mathcal{I}_m} n_i$. Our objective is to solve the following problem:

$$\text{minimize} \quad f(x) := \sum_{i\in\mathcal{I}_m} f_i(\check{x}_i). \tag{4.1}$$

Assumption 4.1.2 *The set of minimizers of f, $C := \{x \,|\, f(x) < \infty,\ f(x) \le f(x'),\ \forall x'\}$, is nonempty.*

The following Example 4.1.1 will be used later to illustrate the proposed algorithms.

Example 4.1.1 *Consider two agents with variables $x_1, x_2 \in \mathbb{R}$ and local objective functions $f_1(x_1, x_2) = (x_1^2 + x_2^2)/2$ and $f_2(x_2) = -x_2$. The dependency graph has only one edge $(2, 1)$. Note that $f = f_1 + f_2$ has a unique minimizer $x_1^* = 0$ and $x_2^* = 1$, even though no minimizer exists for f_2.*

Our goal is to design distributed iterative algorithms $x^{t+1} = T(x^t)$ for solving Problem (4.1) so that, at any iteration, each agent updates its variable by using only the variables of its neighbors and itself, and that the iteration result x^t converges to a solution $x^* \in C$ for all initial x^0.

Towards this goal, we first reformulate Problem (4.1). For each neighboring agent pair $(j, i) \in \mathcal{E}$, suppose agent i maintains an extra variable $x_{ji} \in \mathbb{R}^{n_j}$ representing its desired value for the variable x_j of its in-neighboring agent j (it is possible that $x_{ji} \neq x_j$). Denote by $\mathbf{x}_i = \big(x_i, (x_{ji})_{j\in\mathcal{N}_+}\big) \in \mathbb{R}^{N_i}$ the augmented variable of agent i, and let $\mathbf{x} := (\mathbf{x}_i)_{i\in\mathcal{I}_m} \in \mathbb{R}^N$ where $N = \sum_{i\in\mathcal{I}_m} N_i$. Then, Problem (4.1) is equivalent to the following optimization problem:

$$\text{minimize} \ \ F(\mathbf{x}) := \sum_{i\in\mathcal{I}_m} f_i(\mathbf{x}_i) \ \ \text{subject to} \ \ \mathbf{x} \in \mathcal{A}. \tag{4.2}$$

Here, $\mathcal{A}$ is the consensus subspace defined by

$$\mathcal{A} = \{\mathbf{x} \,|\, x_{ji} = x_j,\ \forall (j,i) \in \mathcal{E}\} \subset \mathbb{R}^N. \tag{4.3}$$

Problem (4.2) is further equivalent to

$$\text{minimize} \quad F(\mathbf{x}) + \imath_{\mathcal{A}}(\mathbf{x}). \tag{4.4}$$

As can be seen later that all algorithms proposed in this chapter require bidirectional communication between neighboring agents in the dependency graph $(\mathcal{I}_m, \mathcal{E})$, we impose the following Assumption 4.1.3.

Assumption 4.1.3 *The communication in the dependency graph $(\mathcal{I}_m, \mathcal{E})$ is bidirectional, i.e., the communication graph is the union of $(\mathcal{I}_m, \mathcal{E})$ and its transpose $(\mathcal{I}_m, \mathcal{E}^\top)$.*

4.2 Application Examples

We list several examples of Problem (4.1) below.

Example 4.2.1 (L_1-regularized least square problem) *To find sparse approximate solutions to the linear equation $Ax = b$, one can solve the optimization problem $\min(\|Ax - b\|^2 + \lambda\|x\|_1)$ for given $\lambda > 0$. Here, $\|\cdot\|$ and $\|\cdot\|_1$ denote the l_2 and l_1 norms, respectively. By decomposing x into $(x_i)_{i\in\mathcal{I}_m}$ and A into block matrices $(A_{ij})_{i,j\in\mathcal{I}_m}$, this is equivalent to Problem (4.1) with $f_i = \|\sum_{j\in\mathcal{N}_i^+} A_{ij}x_j - b_i\|^2 + \lambda\|x_i\|_1$. Note that $\mathcal{N}_i^+ = \{j \,|\, A_{ij} \neq 0\}$ and $\mathcal{N}_i^- = \{j \,|\, A_{ji} \neq 0\}$.*

Example 4.2.2 (Convex feasibility problem) *Let $f_i(\check{x}_i) = \imath_{\mathcal{F}_i}(\check{x}_i)$ be the convex indicator function of some convex set $\mathcal{F}_i \subset \mathbb{R}^{N_i}$. Then Problem (4.1) is equivalent to finding a point x in the intersection of the sets $\{x \,|\, \check{x}_i \in \mathcal{F}_i\}$ for $i \in \mathcal{I}_m$.*

Example 4.2.3 (Consensus optimization) *Suppose for each i, $x_i \in \mathbb{R}^n$, $f_i(\check{x}_i) = g_i(x_i) + \imath_{\{x_i = x_j, \forall j\in\mathcal{N}_i^+\}}$, and the dependency graph is weakly connected. Then Problem (4.1) is equivalent to the problem of minimizing $g_1(x) + \cdots + g_m(x)$.*

Example 4.2.4 (Min-max coordinated optimization) *Suppose each agent* $i \in \{2, \dots, m\}$ *has a local variable* x_i *and a local objective function* $f_i(x_i)$*, while agent* 1 *is a coordinator with no local variables and a non-separable objective function in the form of* $f_1 = \max_{i=2,\dots,m} |||x_i|||$*, where* $|||\cdot|||$ *denotes the general norm, which can* l_1*-,* l_2*- or* l_∞*-norm. In this case, the dependency graph has the edges* $(2,1), (3,1), \dots, (m,1)$*.*

4.3 Synchronous Algorithms

In this section, several distributed algorithms for solving Problem (4.2) are proposed. Recall that $\mathbf{x} = (\mathbf{x}_i)_{i\in\mathcal{I}_m} \in \mathbb{R}^N$, where $\mathbf{x}_i = \left(x_i, (x_{ji})_{j\in\mathcal{N}_i^+}\right)$ is the local augmented variable kept by agent i. Denote by $\bar{\mathbf{x}} := \Pi_{\mathcal{A}}(\mathbf{x})$ the orthogonal projection of $\mathbf{x}$ onto the consensus subspace $\mathcal{A}$ defined in (4.3). Then, $\bar{\mathbf{x}} = (\bar{\mathbf{x}}_i)_{i\in\mathcal{I}_m}$ is given by

$$\bar{x}_{ji} = \bar{x}_j = \tfrac{1}{|\mathcal{N}_j^-|+1}\left(x_j + \textstyle\sum_{k\in\mathcal{N}_j^-} x_{jk}\right), \quad \forall (j,i) \in \mathcal{E}. \tag{4.5}$$

The projection of $\mathbf{x}$ onto the orthogonal complementary subspace $\mathcal{A}^\perp$ is given by $\Pi_{\mathcal{A}^\perp}(\mathbf{x}) = \mathbf{x} - \bar{\mathbf{x}}$.

4.3.1 Synchronous Douglas-Rachford Algorithm

We now apply the D-R algorithm to Problem (4.4). Choose $T_1 = \partial F$ and $T_2 = \partial \imath_{\mathcal{A}}$. Then, $J_{\rho T_1} = \mathbf{prox}_{\rho F}$ and $J_{\rho T_2} = \Pi_{\mathcal{A}}$. The Douglas-Rachford algorithm for $\alpha \in (0,1)$ becomes

$$\mathbf{x}_i^{t+1} = \bar{\mathbf{z}}_i^t, \quad i \in \mathcal{I}_m; \tag{4.6a}$$

$$\mathbf{z}_i^{t+1} = \mathbf{z}_i^t + 2\alpha\left(\mathbf{prox}_{\rho f_i}(2\mathbf{x}_i^{t+1} - \mathbf{z}_i^t) - \mathbf{x}_i^{t+1}\right), \ i \in \mathcal{I}_m. \tag{4.6b}$$

Note that step (4.6a) is carried out in two fully synchronous stages: first each agent i gathers variables z_{ij}^t from all of its out-neighbors j and computes the average of z_i^t and the gathered variables as x_i^{t+1}; then, after the first stage is completed for all agents, each agent i gathers the variables x_j^{t+1} from all of its in-neighbors j and updates x_{ji}^{t+1} to x_j^{t+1}. Step (4.6b) requires no inter-agent communication. The whole algorithm is summarized in Algorithm 5.

Algorithm 5 Synchronous Douglas-Rachford Algorithm

Initialize $\mathbf{z}^0$, and let $t \leftarrow 0$
repeat
  for $i = 1, \ldots, m$ **do**
    $\mathbf{x}_i^{t+1} \leftarrow \bar{\mathbf{z}}_i^t$
  end for
  for $i = 1, \ldots, m$ **do**
    $\mathbf{z}_i^{t+1} \leftarrow \mathbf{z}_i^t + 2\alpha \left(\mathbf{prox}_{\rho f_i}(2\mathbf{x}_i^{t+1} - \mathbf{z}_i^t) - \mathbf{x}_i^{t+1} \right)$
  end for
  $t \leftarrow t + 1$
until $|\mathbf{z}^t - \mathbf{z}^{t-1}|$ is sufficiently small
return $\mathbf{x}^t$

As the iteration from $\mathbf{z}^t$ to $\mathbf{z}^{t+1}$ is given by an α-averaged map in (2.8), the sequence $\mathbf{z}^t$ obtained by Algorithm 5 converges to some $\mathbf{z}^*$ for which $\mathbf{x}^* := \bar{\mathbf{z}}^*$ yields a solution to Problem (4.4). Note that in general $\mathbf{x}^* \neq \mathbf{z}^*$, i.e., it is possible that $z_{ji}^* \neq z_j^*$ for some $(j,i) \in \mathcal{E}$.

Remark 4.3.1 *Another version of the Douglas-Rachford algorithm is obtained by letting $T_1 = \partial \imath_{\mathcal{A}}$ and $T_2 = \partial F$:*

$$\mathbf{x}_i^{t+1} = \mathbf{prox}_{\rho f_i}(\mathbf{z}_i^t), \quad i \in \mathcal{I}_m; \tag{4.7a}$$

$$\mathbf{z}_i^{t+1} = \mathbf{z}_i^t + 2\alpha\big(2\bar{\mathbf{x}}_i^{t+1} - \bar{\mathbf{z}}_i^t - \mathbf{x}_i^{t+1}\big), \quad i \in \mathcal{I}_m, \tag{4.7b}$$

for $t = 0, 1, \ldots$. The sequence $\mathbf{z}^t$ converges to some $\mathbf{z}^*$ so that $\mathbf{prox}_{\rho F}(\mathbf{z}^*)$ yields a solution to Problem (4.4). Generally, $\mathbf{prox}_{\rho f_i}(\mathbf{z}_i^*) \neq \mathbf{z}_i^*$, i.e., $\mathbf{z}_i^*$ is not a minimizer of f_i.

Example 4.3.1 *In Example 4.1.1, let $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2) \in \mathbb{R}^3$ where $\mathbf{z}_1 = (z_1, z_{21}) \in \mathbb{R}^2$ and $\mathbf{z}_2 = z_2 \in \mathbb{R}$. Then $\mathbf{prox}_{\rho f_1}(\mathbf{z}_1) = \mathbf{z}_1/(1+\rho)$ and $\mathbf{prox}_{\rho f_2}(\mathbf{z}_2) = \mathbf{z}_2 + \rho$. Algorithm 5 becomes*

$$\begin{aligned}
z_1^{t+1} &= (1 - 2\alpha\rho/(1+\rho))z_1^t, \\
z_{21}^{t+1} &= (1-\alpha)z_{21}^t + \alpha(1-\rho)/(1+\rho)z_2^t, \\
z_2^{t+1} &= (1-\alpha)z_2^t + \alpha z_{21}^t + 2\alpha\rho.
\end{aligned}$$

It is easily verified that $\mathbf{z}^t$ converges to $\mathbf{z}^* = (0, 1-\rho, 1+\rho)$. From this, $\mathbf{x}^* = \bar{\mathbf{z}}^* = (0,1,1)$ is a solution to Problem (4.4) and thus $x^* = (0,1)$ is a solution to Problem (4.1). In comparison, the sequence $\mathbf{z}^t$ generated by Algorithm (4.7) converges to a different $\mathbf{z}^* = (0, 1+\rho, 1-\rho)$, while resulting in the same solution $\mathbf{x}^* = \mathbf{prox}_{\rho F}(\mathbf{z}^*) = (0,1,1)$.

Douglas-Rachford algorithms are compatible with the agent network topology and can be implemented distributively. However, each of its iteration requires two synchronized rounds of communications and one synchronized round of proximal operator

computation for all the agents. This can slow down its execution due to heterogenous agent computing power and communication delays. We will look for its asynchronous implementation in the next section.

4.3.2 Douglas-Rachford Algorithm for the Dual Problem

Let $S_i \in \mathbb{R}^{N_i \times n}$ be the selection matrix consisting of rows of the n-by-n identify matrix such that $\check{x}_i = S_i x$, $i \in \mathcal{I}_m$. In other words, S_i selects from $x = (x_i)_{i\in\mathcal{I}_m}$ those variables that f_i depends on and arranges them as $\check{x}_i = \big(x_i, (x_j)_{j\in\mathcal{N}_i^+}\big)$. Let $S = \begin{bmatrix} S_1^\top & \cdots & S_m^\top \end{bmatrix}^\top$. Then $\mathbf{z} = Sx$ satisfies $\mathbf{z} = (\mathbf{z}_i)_{i\in\mathcal{I}_m}$ where $z_{ji} = z_j = x_j$ for all $(j,i) \in \mathcal{E}$, i.e., $\mathbf{z} \in \mathcal{A}$. Thus, the range space of S is the consensus subspace $\mathcal{A}$ and the null space of $S^\top$ is $\mathcal{A}^\perp$.

Problem (4.2) can be reformulated as follows:

$$\text{minimize} \ \sum_{i\in\mathcal{I}_m} f_i(\mathbf{z}_i) \ \text{ subject to } \ \mathbf{z} = Sx. \tag{4.8}$$

Introduce the dual variable $\mathbf{p} \in \mathbb{R}^N$ and define the Lagrangian

$$L(x, \mathbf{z}, \mathbf{p}) = \sum_{i\in\mathcal{I}_m} (f_i(\mathbf{z}_i) - \mathbf{p}_i^\top(\mathbf{z}_i - S_i x)).$$

Then the dual problem of (4.8) is

$$\text{minimize } F^*(\mathbf{p}) := \sum_{i\in\mathcal{I}_m} f_i^*(\mathbf{p}_i) \ \text{ subject to } \ \mathbf{p} \in \mathcal{A}^\perp, \tag{4.9}$$

where $f_i^*(\mathbf{p}_i) := \sup_{\mathbf{z}_i} \big(\mathbf{p}_i^\top \mathbf{z}_i - f_i(\mathbf{z}_i)\big)$ is the convex conjugate of f_i. We assume that the dual problem has an optimal solution $\mathbf{p}^*$ with the same optimal value as that of the problem (4.2). This is the case, e.g., if L has a saddle point.

By applying the Douglas-Rachford algorithm in (2.8) to problem (4.9) with $T_1 = \partial F^*$, $T_2 = \partial \imath_{\mathcal{A}^\perp}$, $\alpha \in (0,1)$, and with ρ^{-1} in place of ρ, we have, for $t = 0, 1, \ldots$,

$$\begin{aligned} \mathbf{p}^{t+1} &= \Pi_{\mathcal{A}^\perp}(\mathbf{w}^t) = \mathbf{w}^t - \bar{\mathbf{w}}^t, \\ \mathbf{w}_i^{t+1} &= \mathbf{w}_i^t + 2\alpha \left(\mathbf{prox}_{f_i^*/\rho}(2\mathbf{p}_i^{t+1} - \mathbf{w}_i^t) - \mathbf{p}_i^{t+1} \right). \end{aligned}$$

By the Moreau's decomposition $\mathbf{prox}_{f_i^*/\rho} + \rho^{-1}\,\mathbf{prox}_{\rho f_i} \circ \rho I = I$, the above iteration can be rewritten as

$$\mathbf{u}_i^{t+1} = \bar{\mathbf{w}}_i^t, \ \ \forall i \in \mathcal{I}_m; \tag{4.10a}$$

$$\mathbf{w}_i^{t+1} = \mathbf{w}_i^t - 2\alpha\mathbf{u}_i^{t+1} - 2\alpha\rho^{-1}\mathbf{prox}_{\rho f_i}(\rho\mathbf{w}_i^t - 2\rho\mathbf{u}_i^{t+1}), \ \forall i \in \mathcal{I}_m. \tag{4.10b}$$

which is carried out by Algorithm 6 below. Starting from any $\mathbf{w}^0$, the sequence $\mathbf{w}^t$ converges to some $\mathbf{w}^*$ for which $\mathbf{p}^* = \mathbf{w}^* - \bar{\mathbf{w}}^*$ is an optimal solution to the dual problem (4.9).

Algorithm 6 Synchronous Dual D-R Algorithm

Initialize $\mathbf{w}^0$, and let $t \leftarrow 0$
repeat
  for $i = 1, \ldots, m$ **do**
    $\mathbf{u}_i^{t+1} \leftarrow \bar{\mathbf{w}}_i^t$
  end for
  for $i = 1, \ldots, m$ **do**
    $\mathbf{v}_i^{t+1} \leftarrow \mathbf{prox}_{\rho f_i}(\rho\mathbf{w}_i^t - 2\rho\mathbf{u}_i^{t+1})$
    $\mathbf{w}_i^{t+1} \leftarrow \mathbf{w}_i^t - 2\alpha\mathbf{u}_i^{t+1} - 2\alpha\rho^{-1}\mathbf{v}_i^{t+1}$
  end for
  $t \leftarrow t + 1$
until $|\mathbf{w}^t - \mathbf{w}^{t-1}|$ is sufficiently small
return $\mathbf{w}^t - \bar{\mathbf{w}}^t$

Remark 4.3.2 *Switching the order of the two operators when applying the Douglas-Rachford algorithm to problem* (4.8) *by choosing* $T_1 = \partial\imath_{\mathcal{A}^\perp}$ *and* $T_2 = \partial F^*$*, we obtain:*

$$\mathbf{p}_i^t = \mathbf{prox}_{f_i^*/\rho}(\mathbf{w}_i^t) = \mathbf{w}_i^t - \rho^{-1}\mathbf{prox}_{\rho f_i}(\rho\mathbf{w}_i^t), \ \ \forall i \in \mathcal{I}_m;$$
$$\mathbf{w}^{t+1} = \mathbf{w}^t + 2\alpha\left(\Pi_{\mathcal{A}^\perp}(2\mathbf{p}^{t+1} - \mathbf{w}^t) - \mathbf{p}^{t+1}\right).$$

Although convergence of $\mathbf{p}^t$ *to the dual optimal solution* $\mathbf{p}^*$ *still holds, this version is generally more computationally costly for asynchronous implementation.*

Example 4.3.2 *For the problem in Example 4.1.1, the dual variables are* $\mathbf{p} = (\mathbf{p}_1, \mathbf{p}_2)$ *with* $\mathbf{p}_1 = (p_1, p_{12})$ *and* $\mathbf{p}_2 = p_2$. *The convex conjugate functions are* $f_1^*(\mathbf{p}_1) = \frac{1}{2}\|\mathbf{p}_1\|^2$ *and* $f_2^*(\mathbf{p}_2) = \imath_{\{p_2=-1\}}$. *The dual problem* (4.9) *is to minimize* $(p_1^2 + p_{12}^2)/2$ *subject to* $p_2 = -1$ *and* $\mathbf{p} \in \mathcal{A}^\perp$ *(i.e.,* $p_{12} + p_2 = 0$*). Thus, the dual problem has a unique solution* $\mathbf{p}^* = (0, 1, -1)$. *Algorithm 6 becomes the iteration*

$$\begin{aligned} w_1^{t+1} &= (1 - 2\alpha\rho/(1+\rho))w_1^t \\ w_{12}^{t+1} &= (1-\alpha)w_{12}^t + \alpha(1-\rho)/(1+\rho)w_2^t \\ w_2^{t+1} &= (1-\alpha)w_2^t + \alpha w_{12}^t - 2\alpha, \end{aligned}$$

which converges to $w_1^* = 0$, $w_{12}^* = 1 - \rho^{-1}$, *and* $w_2^* = -1 - \rho^{-1}$. *As expected,* $\mathbf{w}^* - \bar{\mathbf{w}}^* = (0, 1, -1)$ *yields the solution* $\mathbf{p}^*$ *to the dual problem.*

4.3.3 ADMM Algorithm

For problem (4.8) we define the augmented Lagrangian

$$L_\rho(x, \mathbf{z}, \mathbf{y}) = \sum_{i \in \mathcal{I}_m} f_i(\mathbf{z}_i) + \mathbf{y}_i^\top (S_i x - \mathbf{z}_i) + \frac{1}{2\rho}\|S_i x - \mathbf{z}_i\|^2.$$

The ADMM algorithm [49] first minimizes L_ρ w.r.t. the primal variables x and $\mathbf{z}$ in a Gauss-Seidel pass, then updates the dual variable $\mathbf{y}$ using the primal residue:

$$\begin{aligned} x^{t+1} &= \operatorname*{argmin}_x \; L_\rho(x, \mathbf{z}^t, \mathbf{y}^t) = (S^\top S)^{-1} S^\top (\mathbf{z}^t - \rho \mathbf{y}^t) \\ \mathbf{z}_i^{t+1} &= \operatorname*{argmin}_{\mathbf{z}_i} \; L_\rho(x^t, \mathbf{z}, \mathbf{y}^t) = \mathbf{prox}_{\rho f_i}\left(S_i x^{t+1} + \rho \mathbf{y}_i^t\right) \\ \mathbf{y}_i^{t+1} &= \mathbf{y}_i^t + (S_i x^{t+1} - \mathbf{z}_i^{t+1})/\rho, \;\; i \in \mathcal{I}_m. \end{aligned}$$

Since $(S^\top S)^{-1} S^\top = S^\dagger$ is the pseudo inverse of S, $S(S^\top S)^{-1} S^\top$ is the projection onto $\mathcal{A}$. By replacing x^{t+1} with $\mathbf{x}^{t+1} = Sx^{t+1}$, the above algorithm becomes

$$\mathbf{x}^{t+1} = \bar{\mathbf{z}}^t - \rho \bar{\mathbf{y}}^t, \tag{4.12a}$$

$$\mathbf{z}_i^{t+1} = \mathbf{prox}_{\rho f_i}(\mathbf{x}_i^{t+1} + \rho \mathbf{y}_i^t) = \mathbf{prox}_{\rho f_i}\left(\bar{\mathbf{z}}_i^t + \rho(\mathbf{y}_i^t - \bar{\mathbf{y}}_i^t)\right), \;\; i \in \mathcal{I}_m \tag{4.12b}$$

$$\mathbf{y}^{t+1} = \mathbf{y}^t + (\mathbf{x}^{t+1} - \mathbf{z}^{t+1})/\rho = \mathbf{y}^t - \bar{\mathbf{y}}^t - (\mathbf{z}^{t+1} - \bar{\mathbf{z}}^t)/\rho, \tag{4.12c}$$

which is summarized in Algorithm 7. If the Lagrangian (L_ρ with $\rho = \infty$) has a saddle point $(x^*, \mathbf{z}^*, \mathbf{y}^*)$, then for any $(\mathbf{z}^0, \mathbf{y}^0)$ the ADMM algorithm satisfies that $\mathbf{x}^t - \mathbf{z}^t$ converges to zero, $\mathbf{y}^t$ converges to an optimal dual solution $\mathbf{y}^*$, and $\sum_i f_i(\mathbf{z}_i^t)$ converges to the optimal value of problem (4.8) (see [49]). In general, further assumptions are needed for $\mathbf{x}^t$ (or $\mathbf{z}^t$) to converge to an optimal primal solution of the problem (4.8).

Algorithm 7 Synchronous ADMM Algorithm

Initialize $\mathbf{z}^0$, $\mathbf{y}^0$, and let $t \leftarrow 0$
repeat
  for $i = 1, \ldots, m$ **do**
    $\mathbf{x}_i^{t+1} \leftarrow \bar{\mathbf{z}}_i^t - \rho \bar{\mathbf{y}}_i^t$
    $\mathbf{z}_i^{t+1} \leftarrow \mathbf{prox}_{\rho f_i}(\bar{\mathbf{z}}_i^t + \rho(\mathbf{y}_i^t - \bar{\mathbf{y}}_i^t))$
  end for
  for $i = 1, \ldots, m$ **do**
    $\mathbf{y}_i^{t+1} \leftarrow \mathbf{y}_i^t - \bar{\mathbf{y}}_i^t - (\mathbf{z}_i^{t+1} - \bar{\mathbf{z}}_i^t)/\rho$
  end for
  $t \leftarrow t + 1$
until $|\mathbf{y}^t - \mathbf{y}^{k-1}|$ and $|\mathbf{x}^t - \mathbf{z}^t|$ are sufficiently small
return $\mathbf{x}^t$ (or $\mathbf{z}^t$)

Remark 4.3.3 *It is well known [84] that the ADMM algorithm can be derived by applying the Douglas-Rachford algorithm to the dual problem. To see this explicitly in our case, let $\alpha = 1/2$ in Algorithm 6. The iteration on $\mathbf{w}^t$ then becomes:*

$$\begin{aligned} &\mathbf{z}_i^{t+1} = \mathbf{prox}_{\rho f_i}(\rho \mathbf{w}_i^t - 2\rho \bar{\mathbf{w}}_i^t), \quad i \in \mathcal{I}_m; \\ &\mathbf{w}^{t+1} = \mathbf{w}^t - \bar{\mathbf{w}}^t - \mathbf{z}^{t+1}/\rho. \end{aligned} \tag{4.13}$$

For $k = 1, 2, \ldots$, define the variables

$$\mathbf{x}^{t+1} := -\rho \bar{\mathbf{w}}^t + \bar{\mathbf{z}}^t - \bar{\mathbf{z}}^{k-1}, \tag{4.14}$$

$$\mathbf{y}^t := \mathbf{w}^t - \bar{\mathbf{w}}^t - (\bar{\mathbf{z}}^t - \bar{\mathbf{z}}^{k-1})/\rho. \tag{4.15}$$

Then we have

$$\mathbf{z}_i^{t+1} = \mathbf{prox}_{\rho f_i}(\mathbf{x}_i^{t+1} + \rho \mathbf{y}_i^t), \;\; i \in \mathcal{I}_m. \tag{4.16}$$

Applying $\Pi_{\mathcal{A}}$ *to* (4.13) *and* (4.15)*, we obtain* $\bar{\mathbf{w}}^{t+1} = -\bar{\mathbf{z}}^{t+1}/\rho$ *and* $\bar{\mathbf{y}}^t = -(\bar{\mathbf{z}}^t - \bar{\mathbf{z}}^{k-1})/\rho$*. Applying* $\Pi_{\mathcal{A}^\perp}$ *to* (4.15)*, we have* $\mathbf{y}^t - \bar{\mathbf{y}}^t = \mathbf{w}^t - \bar{\mathbf{w}}^t$*. Thus, for* $k = 1, 2, \ldots$*,*

$$\begin{aligned}
&\mathbf{y}^{t+1} = \bar{\mathbf{y}}^{t+1} + (\mathbf{y}^{t+1} - \bar{\mathbf{y}}^{t+1}) = -(\bar{\mathbf{z}}^{t+1} - \bar{\mathbf{z}}^t)/\rho + (\mathbf{w}^{t+1} - \bar{\mathbf{w}}^{t+1}) \\
&= -(\bar{\mathbf{z}}^{t+1} - \bar{\mathbf{z}}^t)/\rho + \left(\mathbf{w}^t - \bar{\mathbf{w}}^t - (\mathbf{z}^{t+1} - \bar{\mathbf{z}}^{t+1})/\rho\right) \\
&= -(\bar{\mathbf{z}}^{t+1} - \bar{\mathbf{z}}^t)/\rho + \left(\mathbf{y}^t - \bar{\mathbf{y}}^t - (\mathbf{z}^{t+1} - \bar{\mathbf{z}}^{t+1})/\rho\right) = \mathbf{y}^t - \bar{\mathbf{y}}^t - (\mathbf{z}^{t+1} - \bar{\mathbf{z}}^t)/\rho,
\end{aligned}$$

and

$$\mathbf{x}^{k+2} = -\rho \bar{\mathbf{w}}^{t+1} + (\bar{\mathbf{z}}^{t+1} - \bar{\mathbf{z}}^k) = \bar{\mathbf{z}}^{t+1} - \rho \bar{\mathbf{y}}^{t+1}.$$

These two equations together with (4.16) *are exactly the same with ADMM algorithm* (4.12)*.*

4.4 Asynchronous Algorithms

Each of the synchronous algorithms in Section 4.3 can be turned into an asynchronous one by utilizing Theorem 2.2.2. When doing so, some algorithms will have "better" asynchronous implementations compared to others. Here, the criteria for comparing different asynchronous implementations are in terms of the computation and communication overheads incurred in each iteration.

4.4.1 Asynchronous Douglas-Rachford Algorithm

By applying Theorem 2.2.2 to the Douglas-Rachford algorithm (4.7), we obtain the following asynchronous algorithm:

For $k = 0, 1, \ldots$, pick $i \in \mathcal{I}_m$ with i.i.d. probability $p_i > 0$

$\mathbf{x}_j^{t+1} = \mathbf{prox}_{\rho f_j}\left(\mathbf{z}_j^t\right), \;\; \forall j \in \mathcal{N}_i \cup (\cup_{\ell \in \mathcal{N}_i^+} \mathcal{N}_\ell^-)$;

$\mathbf{z}_i^{t+1} = \mathbf{z}_i^t + 2\alpha\left(2\bar{\mathbf{x}}_i^{t+1} - \bar{\mathbf{z}}_i^t - \mathbf{x}_i^{t+1}\right)$ for the chosen i.

Even though at each round only one agent i is activiated to carry out the update, its updated variable $\mathbf{z}_i^{t+1}$ relies on $\bar{\mathbf{x}}_i^{t+1}$ whose evaluation requires each agent in an extended (2-hop) neighborhood $\mathcal{N}_i \cup (\cup_{\ell \in \mathcal{N}_i^+} \mathcal{N}_\ell^-)$ to evaluate its proximal operator and send out its local information. This may result in large computation and communication overheads. For instance, for the coordinated optimization problem in Example 4.2.4, regardless of which agent is chosen to do the update, all the agents need to evaluate their proximal operators and communicate the results to their neighbors.

A better option is to apply Theorem 2.2.2 to Algorithm 5 to obtain:

$$\begin{aligned}
&\text{For } t = 0, 1, \ldots, \text{ pick } i \in \mathcal{I}_m \text{ with i.i.d. probability } p_i > 0 \\
&\mathbf{x}_i^{t+1} = \bar{\mathbf{z}}_i^t; && \text{(4.18a)} \\
&\mathbf{z}_i^{t+1} = \mathbf{z}_i^t + 2\alpha \left(\mathbf{prox}_{\rho f_i}(2\mathbf{x}_i^{t+1} - \mathbf{z}_i^t) - \mathbf{x}_i^{t+1} \right). && \text{(4.18b)}
\end{aligned}$$

In each round, only the activated agent i needs to evaluate its proximal operator once. By Theorem 2.2.2, starting from any $\mathbf{z}^0$, the sequence $\mathbf{z}^t$ generated by Algorithm (4.18) converges almost surely to some $\mathbf{z}^*$ for which $\bar{\mathbf{z}}^*$ is a solution to Problem (4.4).

Example 4.4.1 *By applying Algorithm* (4.18) *to Example 4.1.1 where* $f_1(\hat{x}_1) = (x_1^2 + x_2^2)/2$ *and* $f(\hat{x}_2) = -x_2$*, we arrive at the following iteration:*

$$\begin{cases}
z_1^{t+1} = (1 - 2\alpha\rho/(1+\rho))z_1^t, \\
z_{12}^{t+1} = (1-\alpha)z_{12}^t + \alpha(1-\rho)/(1+\rho)z_2^t, \\
z_2^{t+1} = z_2^t,
\end{cases}$$

if agent 1 *is activated, and*

$$z_1^{t+1} = z_1^t,\ z_{12}^{t+1} = z_{12}^t,\ z_2^{t+1} = (1-\alpha)z_2^t + \alpha z_{12}^t + 2\alpha\rho$$

if agent 2 *is activated. For* $p_1, p_2 > 0$*, with probability one* $\mathbf{z}^t$ *converges to* $\mathbf{z}^* = (0, 1-\rho, 1+\rho)$ *and* $\mathbf{x}^t$ *converges to the solution* $\bar{\mathbf{z}}^* = (0, 1, 1)$*.*

In Algorithm (4.18), to compute $\bar{\mathbf{z}}_i^t$ in step (4.18a), agent i needs to collect $z_{\ell i}^t$ from its out-neighbors ℓ and $\bar{z}_j^t$ from its in-neighbors j. The latter requires agents j to gather data from their respective out-neighbors. To avoid this, we can let each agent $i \in \mathcal{I}_m$ maintain an extra variable $\bar{z}_i \in \mathbb{R}^{n_i}$ that always has the averaged value of x_i and x_ℓ for $\ell \in \mathcal{N}_i^-$. If agent i does not have a local variable, then $\bar{z}_i$ is null. Then Algorithm (4.18) is equivalent to Algorithm 8 below.

Algorithm 8 Asynchronous Douglas-Rachford Algorithm

1: Choose any $\mathbf{z}^0$, and let $t \leftarrow 0$
2: **for** $i = 1, \ldots, m$ **do**
3: $\quad \bar{z}_i^0 \leftarrow (z_i^0 + \sum_{\ell \in \mathcal{N}_i^-} z_{\ell i}^0)/(|\mathcal{N}_i^-| + 1)$
4: **end for**
5: **repeat**
6: $\quad$ Pick $i \in \mathcal{I}_m$ with i.i.d. probability $p_i > 0$
7: $\quad x_i^{t+1} \leftarrow \bar{z}_i^t$
8: $\quad$ **for** $j \in \mathcal{N}_i^+$ **do**
9: $\quad\quad x_{ij}^{t+1} \leftarrow \bar{z}_j^t$
10: $\quad$ **end for**
11: $\quad \mathbf{z}_i^{t+1} \leftarrow \mathbf{z}_i^t + 2\alpha \left(\mathbf{prox}_{\rho f_i}(2\mathbf{x}_i^{t+1} - \mathbf{z}_i^t) - \mathbf{x}_i^{t+1}\right)$
12: $\quad \bar{z}_i^{t+1} \leftarrow \bar{z}_i^t + (z_i^{t+1} - z_i^t)/(|\mathcal{N}_i^-| + 1)$
13: $\quad$ **for** $j \in \mathcal{N}_i^+$ **do**
14: $\quad\quad \bar{z}_j^{t+1} \leftarrow \bar{z}_j^t + (z_{ij}^{t+1} - z_{ij}^t)/(|\mathcal{N}_j^-| + 1)$
15: $\quad$ **end for**
16: $\quad t \leftarrow t + 1$
17: **until** t is sufficiently large
18: **return** $\mathbf{x}^t$

In each round of Algorithm 8, the activated agent i only needs to communicate with its in-neighbors by collecting information in step 8 and sending information in step 12; its in-neighbors perform simple updates (step 12); while all other agents (even

if they are out-neighbors of agent i) can remain idle. For the coordinated optimization problem in Example 4.2.4, if the activated agent $i \in \{2, \ldots, m\}$, then agent i carries out steps 9 and 10 by itself, while all other agents including agent 1 remain idle.

Algorithm 8 has low communication and computation complexity. Specifically, in each round, the expected number of one-way transmission is $\sum_i 2p_i|\mathcal{N}_i^+| \leq \max_i 2|\mathcal{N}_i^+|$; the expected number of scalar variables transmitted is $\sum_i 2p_i(N_i - n_i) \leq \max_i 2(N_i - n_i)$; and only one proximal operator is evaluated.

Remark 4.4.1 *With the initialization step for $\bar{z}_i^0$ and the error-free iterations of Algorithm 8, the variables $\bar{z}_i^t$ at the beginning (or end) of each iteration satisfy the consistency condition $\bar{z}_i^t = (z_i^t + \sum_{\ell \in \mathcal{N}_i^-} z_{\ell i}^t)/(|\mathcal{N}_i^-| + 1)$ for all $i \in \mathcal{I}_m$. This may not be the case if there are numerical errors in the initialization step or the algorithm iterations. To account for this, re-initializations of z_i^t may be warranted from time to time.*

Corollary 4.4.1 *Suppose $\alpha \in (0, 1)$, $\rho > 0$, and $p_i > 0$ for $i \in \mathcal{I}_m$. Starting from any $\mathbf{z}^0$, with probability one the sequence $\mathbf{x}^t$ generated by Algorithm 8 converges to a solution to Problem* (4.2).

Proof Algorithm 8, or equivalently, Algorithm (4.18), is obtained by applying the randomly activated coordinate descent method in Theorem 2.2.2 to the synchronous iteration in (4.6). As the discussions preceding (4.6) show, the update of $\mathbf{z}^t$ to $\mathbf{z}^{t+1}$ in (4.6) is via the α-averaged operator $(1-\alpha)I + \alpha(2\mathbf{prox}_{\rho F} - I)(2\Pi_{\mathcal{A}} - I)$. Therefore, Theorem 2.2.2 applies. This implies that, with probability one, the sequence $\mathbf{z}^t$ obtained by Algorithm 8 converges to a fixed point $\mathbf{z}^*$ of the α-averaged operator, and hence $\mathbf{x}^t = \bar{\mathbf{z}}^t$ converges to $\bar{\mathbf{z}}^*$, a solution to Problem (4.2). ■

4.4.2 Asynchronous Dual Douglas-Rachford Algorithm

Let each agent i maintain the variables $\mathbf{w}_i, \mathbf{u}_i, \mathbf{v}_i \in \mathbb{R}^{N_i}$ and $\bar{w}_i \in \mathbb{R}^{n_i}$. The asynchronous version of Algorithm (4.10) obtained using Theorem 2.2.2 is given by

Algorithm 9 below. In each round, with the agent i activated, the proximal operator is evaluated only once (step 9). Agent i communicates only with its out-neighbors (collect in step 8 and send in step 13). The expected numbers of one-way communications and the expected number of scalar variables transmitted in each round are the same as those of Algorithm 8.

Algorithm 9 Asynchronous Dual D-R Algorithm

1: Choose any $\mathbf{w}^0$, and let $t \leftarrow 0$
2: **for** $i = 1, \ldots, m$ **do**
3: $\quad \bar{w}_i^0 \leftarrow (w_i^0 + \sum_{\ell \in \mathcal{N}_i^-} w_{\ell i}^0)/(|\mathcal{N}_i^-| + 1)$
4: **end for**
5: **repeat**
6: $\quad$ Pick $i \in \mathcal{I}_m$ with i.i.d. probability $p_i > 0$
7: $\quad u_i^{t+1} \leftarrow \bar{w}_i^t$
8: $\quad$ **for** $j \in \mathcal{N}_i^+$ **do**
9: $\quad\quad u_{ij}^{t+1} \leftarrow \bar{w}_j^t$
10: $\quad$ **end for**
11: $\quad \mathbf{v}_i^{t+1} \leftarrow \mathbf{prox}_{\eta f_i}(\eta \mathbf{w}_i^t - 2\eta \mathbf{u}_i^{t+1})$
12: $\quad \mathbf{w}_i^{t+1} \leftarrow \mathbf{w}_i^t - 2\alpha \mathbf{u}_i^{t+1} - 2\alpha\eta^{-1}\mathbf{v}_i^{t+1}$
13: $\quad \bar{w}_i^{t+1} \leftarrow \bar{w}_i^t + (w_i^{t+1} - w_i^t)/(|\mathcal{N}_i^-| + 1)$
14: $\quad$ **for** $j \in \mathcal{N}_i^+$ **do**
15: $\quad\quad \bar{w}_j^{t+1} \leftarrow \bar{w}_j^t + (w_{ij}^{t+1} - w_{ij}^t)/(|\mathcal{N}_j^-| + 1)$
16: $\quad$ **end for**
17: $\quad t \leftarrow t + 1$
18: **until** t is sufficiently large
19: **return** $\mathbf{w}^t - \bar{\mathbf{w}}^t$

Corollary 4.4.2 *Suppose $\alpha \in (0, 1)$, $\rho > 0$, and $p_i > 0$ for $i \in \mathcal{I}_m$. Starting from any initial $\mathbf{w}^0$, the sequence $\mathbf{w}^t - \bar{\mathbf{w}}^t$ obtained by Algorithm 9 converges with probability one to a solution to the dual problem* (4.9).

Proof Similar to the proof of Corollary 4.4.1, this follows directly by applying Theorem 2.2.2 to the iteration (4.10). ■

5. CONVEX OPTIMIZATION PROBLEMS WITH LOCAL AND GLOBAL COUPLINGS

The problem formulated in Chapter 4 and its corresponding distributed solutions focus on the coupling involving neighboring agents. However, the coupling in some practical problems may involve all (or at least most of) agents on networks (see, e.g., Examples 5.2.1 and 5.2.2). In this chapter, we incorporate this extra coupling and propose distributed solutions with and without a coordinator.

5.1 Problem Formulation

With the same $f(x)$ from Problem (4.1), the problem to be studied in this section is defined as follows:

$$\text{minimize} \quad f(x) = \sum_{i \in \mathcal{I}_m} f_i(\check{x}_i) \tag{5.1a}$$

$$\text{subject to} \quad \sum_{i \in \mathcal{I}_m} h_i(x_i) \leq c_0, \tag{5.1b}$$

where $h_i : \mathbb{R}^{n_i} \to \overline{\mathbb{R}}^{\ell}$ represents the local contribution of agent i to the global constraint, e.g., the local consumption function in the resource allocation problem, and $c_0 \in \mathbb{R}^{\ell}$ is a constant vector. In the above optimization problem, each agent's local objective function $f_i(\check{x}_i)$ is only coupled with its immediate neighbors while the global constraint (5.1b) depends on the local variables from all agents. Note that (5.1b) consists of ℓ scalar inequality constraints, each of which represents the total availability of a global resource (or commodity, capacity) being shared among all of the agents.

Remark 5.1.1 *In the case where h_i in* (5.1b) *also depends on the variables x_j from in-neighboring agents $j \in \mathcal{N}_i^+$, the constraint can be modified as $\sum_{i \in \mathcal{I}_m} \check{h}_i(\check{x}_j) \leq c_0$.*

The method to be developed in this section can be extended to deal with this more general case as well.

In addition to the Assumption 4.1.1, the following assumptions are made on Problem (5.1).

Assumption 5.1.1 *For each $i \in \mathcal{I}_m$, it is assumed that $h_i = (h_{i1}, \ldots, h_{i\ell}) : \mathbb{R}^{n_i} \to \overline{\mathbb{R}}^{\ell}$ where each entry h_{ij}, $j \in \mathcal{I}_l$, is an extended-real-valued, CCP function.*

Assumption 5.1.2 *A solution to Problem* (5.1) *exists.*

Explicitly, Assumption 5.1.2 implies that the optimal value of Problem (5.1) can be exactly attained by some $x^* = (x_1^*, \ldots, x_m^*)$ that satisfies the global constraints (5.1b).

To simplify the solution of Problem (5.1), we introduce a local auxiliary variable b_i at each agent i and let $z_i = (x_i, b_i)$ be the augmented local variable. Write $b = (b_1, \ldots, b_m)$. Then Problem (5.1) can be reformulated as:

$$\text{minimize}_{x,b} \quad f(x) = \sum_{i \in \mathcal{I}_m} f_i(\check{x}_i) \tag{5.2a}$$

$$\text{subject to} \quad h_i(x_i) \leq b_i, \quad \forall i \in \mathcal{I}_m \tag{5.2b}$$

$$\text{and} \quad \sum_{i \in \mathcal{I}_m} b_i \leq c_0. \tag{5.2c}$$

Note that the local constraints in (5.2b) can be incorporated into the local objective function f_i via the convex indicator functions $\imath_{Z_i}(z_i)$ of the closed convex sets $Z_i := \{z_i \,|\, h_i(x_i) \leq b_i\}$; whereas the global constraint (5.2c) is a linear inequality constraint of the form $\sum_{i \in \mathcal{I}_m} F_i z_i \leq c_0$ for some properly defined matrices F_i, $i \in \mathcal{I}_m$.

Remark 5.1.2 *When the global constraint* (5.2c) *is replaced by a linear equality constraint $\sum_{i \in \mathcal{I}_m} b_i = c_0$, the recast problem is also equivalent to Problem* (5.2) *(or* (5.1)*). We choose to adopt Problem* (5.2) *here as it will lead to a reformulation* (5.3) *below of Problem* (5.1) *that can capture many practical problems in its given form without the need of introducing auxiliary variables. When removing the orthogonal projections $\Pi_{\mathbb{R}^{\ell}_+}$ and $\Pi_{\mathbb{R}^{\ell m}_+}$, the algorithms* (5.6) *and* (5.19) *(or Algorithm 10) to be proposed later on can be directly used to solve this recast problem.*

As a result of the above observation, in the rest of this section we can focus without loss of generality on the following simplified yet equivalent version of Problem (5.1):

$$\text{minimize} \quad f(x) = \sum_{i \in \mathcal{I}_m} f_i(\check{x}_i) \tag{5.3a}$$

$$\text{subject to} \quad \sum_{i \in \mathcal{I}_m} F_i x_i \leq c_0, \tag{5.3b}$$

where $F_i \in \mathbb{R}^{\ell \times n_i}$, $\forall i \in \mathcal{I}_m$, and $c_0 \in \mathbb{R}^\ell$. With $F = \begin{bmatrix} F_1 & \cdots & F_m \end{bmatrix} \in \mathbb{R}^{\ell \times n}$, the global constraint (5.3b) can be simplified to $Fx \leq c_0$.

A straightforward distributed solution to problem (5.3) is to introduce a coordinator agent indexed by 0 to hold the global constraint. This coordinator has no local variables and is an out-neighbor of all the other agents. Thus, by denoting $\mathcal{F} := \{x \,|\, Fx \leq c_0\}$, the problem can be reduced to Problem (4.1) with $f_0 = \imath_{\mathcal{F}}(x)$ and $\text{prox}_{\rho f_0}(\mathbf{x}_0) = \Pi_{\mathcal{F}}(\mathbf{x}_0) = \mathbf{x}_0 - F^\dagger(F\mathbf{x}_0 - c_0)$. The computation of $F^\dagger$ requires the coordinator to have F_i from each agent, which is difficult to implement when F_i is private to agent i in some practical situations, as stated in Assumption 5.1.3.

Suppose c_0 can be decomposed as $c_0 = c_1 + \ldots + c_m$ with $c_i, i \in \mathcal{I}_m$, maintained by agent i. Then the global constraint can be rewritten as $\sum_{i \in \mathcal{I}_m}(F_i x_i - c_i) \leq 0$. For example, in the commodity exchange problem, c_i may represent the initial commodity owned by agent i.

Assumption 5.1.3 *For each $i \in \mathcal{I}_m$, the pair (F_i, c_i) is private to agent i and will not be shared with other agents.*

The goal of this section is to design distributed solution algorithms for Problem (5.3) consistent with the privacy requirement in Assumption 5.1.3.

5.2 Application Examples

Followed are several instances of Problem (5.3).

Example 5.2.1 *Suppose each agent $i \in \mathcal{I}_m$ has a local variable x_i and a local objective function $f_i(x_i)$, while the global objective function $f_0(x_1, \ldots, x_m) = \| \sum_{i=1}^m x_i \|^2$ is non-separable. By letting $x_0 = \sum_{i=1}^m x_i$, the problem can be transformed to an instance of Problem* (5.3)*:*

$$\textit{minimize}_{x_0, x_1, \ldots, x_m} \sum_{i=0,1,\ldots,m} f_i(x_i), \ \textit{subject to} \ \ x_0 - \sum_{i=1}^m x_i = 0$$

Example 5.2.2 (Optimal resource allocation) *Suppose ℓ types of resources are being allocated among m agents. Denote by $x_i \in \mathbb{R}^\ell$ the (stacked) amounts of resources allocated to agent $i \in \mathcal{I}_m$ and by $u_i(x_i) \in \mathbb{R}$ the resulting utility attained by agent i, where $u_i : \mathbb{R}^\ell \to \mathbb{R}$ is a concave and upper semicontinuous function. The optimal resource allocation that maximizes the total utilities of all agents can be found via solving the following problem:*

$$\max \sum_{i \in \mathcal{I}_m} u_i(x_i) - \imath_{\mathbb{R}_+^\ell}(x_i) \ \textit{such that} \ \ \textit{and} \ \sum_{i \in \mathcal{I}_m} x_i \leq c_0.$$

This problem is a special instance of Problem (5.3) *with the decoupled local objective function $f_i = -u_i + \imath_{\mathbb{R}_+^\ell}$.*

Example 5.2.3 (Building optimal control) *In the problem of comfort assurance in multi-zone buildings from Section 3.2.3, if the goal is to find the optimal control inputs to minimize the overall energy utility cost $\sum_{i=1}^m \sum_{k=0}^{N-1} g_i(x_i(k), u_i(k))$ besides assuring comfort for some convex functions g_i. With $\big(x_i(k+1), u_i(k)\big)_{k \in \{0,\ldots,N-1\}}$ being the local variable held by zone i, the building optimal control problem can be formulated as an instance of Problem* (5.3)*.*

Example 5.2.4 (Distributed model fitting) *Many model fitting problems arising in statistics and machine learning can be formulated as the following problem:*

$$\textit{minimize} \ \ g(Cx - d) + r(x),$$

where $x \in \mathbb{R}^n$ is the model parameter (or features) to be estimated; $C \in \mathbb{R}^{\ell \times n}$ is the feature matrix that contains all training examples as row vectors; $d \in \mathbb{R}^\ell$ is the output

vector; $g : \mathbb{R}^\ell \to \mathbb{R}$ *is a convex loss function; and* $r : \mathbb{R}^n \to \mathbb{R}$ *is a convex regularization function. Two common examples of regularization functions are* $r(x) = \beta\|x\|_1$ *and* $r(x) = \beta\|x\|^2$*, both of which are separable in* x*.*

In many cases, the number of model parameters far exceeds the number of training data, i.e., $n \gg \ell$*, and it is impractical for* x *to be stored in a single unit. In such cases, by adopting the partition* $C = \begin{bmatrix} C_1 & \cdots & C_m \end{bmatrix}$ *and* $x = (x_1, \ldots, x_m)$*, the data and parameter* (C_i, x_i)*,* $i \in \mathcal{I}_m$ *can be stored in* m *separate processing units. Let* $x_0 \in \mathbb{R}^\ell$ *be an auxiliary variable held at another unit. The above problem can then be equivalent formulated as*

$$\begin{aligned} \textit{minimize} \quad & g(x_0 - d) + \sum_{i \in \mathcal{I}_m} r_i(x_i) \\ \textit{subject to} \quad & x_0 - \sum_{i \in \mathcal{I}_m} C_i x_i = 0, \end{aligned}$$

which is an instance of Problem (5.3)*.*

5.3 Synchronous Algorithm with a Coordinator

A key difficulty is the presence of the global constraint (5.3b): it involves the local variables of all agents and hence cannot be enforced by any agent individually given its limited neighborhood. Such a difficulty can be alleviated by designating (or creating) a coordinator agent that can communicate with all the other agents and is specifically tasked with enforcing the global constraint. To be consistent with Assumption 5.1.3, we present a distributed algorithm with a coordinator holding the dual variable corresponding to the global constraint. Using the decomposition $c_0 = c_1 + \ldots + c_m$, the Lagrange function of problem (5.3) is

$$L_a(x, \lambda_0) = \sum_{i \in \mathcal{I}_m} \big(f_i(\check{x}_i) + \lambda_0^\top (F_i x_i - c_i) \big) - \imath_{\mathbb{R}^\ell_+}(\lambda_0), \tag{5.4}$$

Obviously, $L(x, \lambda_0)$ is a saddle function that is convex in x and concave in the dual variable λ_0.

The following assumption is made in the rest of this chapter.

Assumption 5.3.1 *$L_a(x, \lambda_0)$ has a saddle point (x^*, λ_0^*).*

Assumption 5.3.1 is in general stronger than Assumption 5.1.2 as it implies strong duality, i.e., the primal problem (5.3) and its dual problem

$$\text{maximize}_{\lambda_0 \in \mathbb{R}_+} \; d(\lambda_0) := \min_x L_a(x, \lambda_0)$$

have the same optimal value (given by $L_a(x^*, \lambda_0^*)$ in this case). Strong duality can be ensured by, e.g., Slater's conditions.

Therefore Problem (5.3) is equivalent to find a saddle point of L_a in (5.4), where the dual variable λ_0 indirectly enforces the global constraint. Since λ_0 is engaged globally, we let each agent i maintains a local copy λ_i of λ_0 in addition to the augmented variable $\mathbf{x}_i$ defined in Section 4.1. Then Lagrange function (with $\lambda_0 \in \mathbb{R}_+^\ell$) is equivalent to

$$\begin{aligned} L_b(\mathbf{x}, \lambda) &= \sum_{i \in \mathcal{I}_m} L_i + \mu_{\mathcal{A}_x \times \mathcal{A}_\lambda}(\mathbf{x}, \lambda) \\ &:= \sum_{i \in \mathcal{I}_m} \Big(f_i(\mathbf{x}_i) + \lambda_i^\top (F_i \mathbf{x}_i - c_i) - \imath_{\mathbb{R}_+^\ell}(\lambda_i) \Big) + \mu_{\mathcal{A}_x \times \mathcal{A}_\lambda}(\mathbf{x}, \lambda) \end{aligned} \tag{5.5}$$

with $\lambda = (\lambda_i)_{i \in \mathcal{I}_m}$ and $\mathcal{A}_\lambda = \{\lambda \mid \lambda_1 = \cdots = \lambda_m\}$. We designate the coordinator (indexed by 0) to reach consensus on variable λ_0, which has no local objective function. it will be a in-neighbor of every other agent. With $T_1 = \partial \sum L_i$ and $T_2 = \Pi_{\mathcal{A}_x} \times \Pi_{\mathcal{A}_\lambda}$, applying the D-R algorithm (2.8) will result in

$$\bar{\mathbf{x}}^t = \Pi_{\mathcal{A}_x}(\mathbf{x}^t), \bar{\lambda}^t = \Pi_{\mathcal{A}_\lambda}(\lambda^t), \tag{5.6a}$$

$$\left(\mathbf{x}_i^{t+1}, \lambda_i^{t+1}\right) = \left(\mathbf{x}_i^t - 2\alpha \bar{\mathbf{x}}_i^t, \lambda_i^t - 2\alpha \bar{\lambda}_i^t\right) + 2\alpha J_{\rho L_i}\left(2\bar{\mathbf{x}}_i^t - \mathbf{x}_i^t, 2\bar{\lambda}_i^t - \lambda_i^t\right), \; i \in \mathcal{I}_m, \tag{5.6b}$$

where $\bar{\lambda}^t = \Pi_{\mathcal{A}_\lambda}(\lambda^t)$ has the explicit form

$$\bar{\lambda}_1^t = \cdots = \bar{\lambda}_m^t = \frac{1}{m} \sum_{i \in \mathcal{I}_m} \lambda_i^t$$

and $(\mathbf{x}_i', \lambda_i') = J_{\rho L_i}(\mathbf{x}_i, \lambda_i)$ can be evaluated as

$$\mathbf{x}_i' = \operatorname*{argmin}_{\mathbf{x}_i'} \left(f_i(\mathbf{x}_i') + \frac{1}{2\rho} \|\rho(F_i \tilde{S}_i \mathbf{x}_i' - c_i) + \lambda_i\|^2 + \frac{1}{2\rho} \|\mathbf{x}_i' - \mathbf{x}_i\|^2 \right), \tag{5.7a}$$

$$\lambda_i' = \Pi_{\mathbb{R}_+^\ell}\Big(\lambda_i + \rho(F_i \tilde{S}_i \mathbf{x}_i' - c_i)\Big). \tag{5.7b}$$

Here, the matrix $\tilde{S}_i \in \mathbb{R}^{n_i \times N_i}$ selects agent i's own variable x_i from $\mathbf{x}_i$: $x_i = \tilde{S}_i \mathbf{x}_i$ and denote $\tilde{S} := \text{diag}\{\tilde{S}_1, \ldots, \tilde{S}_m\}$. Compared with the selection matrices S_i and S defined in Section 4.3.2, we have $x = \tilde{S}\mathbf{x} = \tilde{S}Sx$ with $\mathbf{x} \in \mathcal{A}_x$ and therefore $\tilde{S}S = I_n$. However, the reverse $S^\top \tilde{S}$ in general does not equal to I_N. The orthogonal projection operator $\Pi_{\mathbb{R}^\ell_+}$ in (5.7b) can be evaluated entry-wise where for each entry $z \in \mathbb{R}$, $\Pi_{\mathbb{R}_+}(z) = \max\{0, z\}$.

The following Theorem establishes the convergence of Algorithm (5.6), which directly follows from (2.8).

Theorem 5.3.1 *Suppose Assumptions 4.1.1, 5.1.1, and 5.3.1 hold and the coordinator can communicate bidirectionally with every other agent $i \in \mathcal{I}_m$. With $\alpha \in (0, 1)$ and $\rho > 0$, the $(\bar{\mathbf{x}}^t, \bar{\lambda}^t)$ generated by Algorithm* (5.6) *will converge to one saddle point $(\mathbf{x}^*, \lambda^*)$ of* (5.5) *and the corresponding x^* is a minimizer of Problem* (5.3)*.*

To carry out the consensus operation (5.6a), in addition to the communication between each agent $i \in \mathcal{I}_m$ and its out-neighbors to reach consensus on x_i, same as in (5), the coordinator exchanges information with every other agent to get the consensus value of the dual variable λ_0. However, in many practical problems, having a coordinator agent may not be feasible given the network structure and could increase the agent network's risk of failures or vulnerability to attacks at a single point. Thus, our focus in next section will be to develop fully distributed solution algorithms where all agents play equal role and no super agents such as coordinator agents are needed.

5.4 Synchronous Algorithm without Coordinators

In Algorithm (5.6), the only role the coordinator plays is to ensure consensus on the dual variables. In the coordinator-free algorithm to be proposed in this section, this task will be distributed among agents where each agent exchanges its own copy of the dual variable with its neighbors on a directed graph $(\mathcal{I}_m, \mathcal{E}_c)$ named *consensus graph*. The consensus graph has the vertex set $\mathcal{I}_m$ and the edge set $\mathcal{E}_c$ such that $(j, i) \in \mathcal{E}_c$ if agent i can receive the dual variable's copy λ_j held by agent j and

there are no self loops. For agent $i \in \mathcal{I}_m$, its in-neighbor set $\mathcal{N}_{ci}^+ := \{j \,|\, (j,i) \in \mathcal{E}_c\}$ consists of all agents who can send their dual variable copies to agent i while the out-neighbor set $\mathcal{N}_{ci}^- := \{k \,|\, (i,k) \in \mathcal{E}_c\}$ consists of agents that can receive λ_i from agent i. Obviously, if $j \in \mathcal{N}_{ci}^+$, then $i \in \mathcal{N}_{cj}^-$.

Assumption 5.4.1 *The consensus graph $(\mathcal{I}_m, \mathcal{E}_c)$ is strongly connected.*

Associate each edge $(j,i) \in \mathcal{E}_c$ with a constant weight $w_{ij} > 0$ and define the corresponding graph Laplacian matrix W as: for $i \neq j$, $W_{ij} = -w_{ij}$ if $(j,i) \in \mathcal{E}_c$ and $W_{ij} = 0$ if otherwise; and for $i \in \mathcal{I}_m$, $W_{ii} = \sum_{j \in \mathcal{N}_{ci}^+} w_{ij}$. By Assumption 5.4.1, W has a simple eigenvalue at 0 with the eigenvector $\mathbf{1}_m \in \mathbb{R}^m$ and the null space $\mathcal{N}(W) = \{w\mathbf{1}_m \,|\, w \in \mathbb{R}\}$. For agent $i \in \mathcal{I}_m$, its (generalized) in-degree d_{ci}^+ and out-degree d_{ci}^- are defined as follows:

$$d_{ci}^+ := \sum_{j \in \mathcal{N}_{ci}^+} w_{ij} = W_{ii}, \quad d_{ci}^- := \sum_{k \in \mathcal{N}_{ci}^-} w_{ki}, \tag{5.8}$$

which will be used later in Lemma 5.4.4 to estimate the parameters of the coordinator-free algorithm.

Let λ_i be the copy of dual variable held by agent i and, with a little abuse of notation, their concatenation be $\lambda := (\lambda_i)_{i \in \mathcal{I}_m}$. With W defined above, the equivalent condition to guarantee consensus of λ_is' is $(W \otimes I_\ell)\lambda = 0$. With $\xi \in \mathbb{R}^{\ell m}$ being the dual variable, incorporating this constraint into the Lagrange function (5.4) leads to

$$\begin{aligned} L_c(x,\xi,\lambda) &= f(x) + \lambda^\top(\tilde{F}x - c) - \imath_{\mathbb{R}_+^{\ell m}}(\lambda) + \xi^\top(W \otimes I_\ell)\lambda \\ &= \sum_{i \in \mathcal{I}_m} \left(f_i(\check{x}_i) + \lambda_i^\top(F_i x_i - c_i) - \imath_{\mathbb{R}_+^\ell}(\lambda_i) + \sum_{j \in \mathcal{N}_{ci}^+} w_{ij}\xi_i^\top(\lambda_i - \lambda_j) \right) \end{aligned} \tag{5.9}$$

with $\tilde{F} := \mathrm{diag}\{F_1, \ldots, F_m\}$ and $c := [c_1, \ldots, c_m]^\top$. Obviously, L_c is a saddle function that is convex of (x,ξ) and concave of λ.

For the indicator function $\imath_{\mathbb{R}_+^\ell}$, its subdifferential set is given by $\partial \imath_{\mathbb{R}_+^\ell}(\lambda_0) = \{q \,|\, q^\top(z - \lambda_0) \leq 0,\ \forall z \in \mathbb{R}_+^\ell\}$ if $\lambda_0 \in \mathbb{R}_+^\ell$ and $\partial \imath_{\mathbb{R}_+^\ell}(\lambda_0) = \emptyset$ if otherwise. Note that the set $\partial \imath_{\mathbb{R}_+^\ell}(\lambda_0)$ is closed under positive scalar multiplication, i.e., $\beta q \in \partial \imath_{\mathbb{R}_+^\ell}(\lambda_0)$

for any $\beta \geq 0$, $q \in \partial\imath_{\mathbb{R}^\ell_+}(\lambda_0)$, as it is a cone. The following property on $\partial\imath_{\mathbb{R}^\ell_+}$ will be used later.

Lemma 5.4.1 *For the positive integers m and ℓ, the vectors $\lambda_0, q_0 \in \mathbb{R}^\ell$, denote $\lambda := \mathbf{1}_m \otimes \lambda_0$ and $q := \mathbf{1}_m \otimes q_0$. Then the following statements are equivalent:*

(1). $q_0 \in \partial\imath_{\mathbb{R}^\ell_+}(\lambda_0)$,

(2). $q \in \imath_{\mathbb{R}^{\ell m}_+}(\lambda)$,

(3). $q_0 \in (\mathbf{1}_m^\top \otimes I_\ell)\partial\imath_{\mathbb{R}^{\ell m}_+}(\lambda) := \{(\mathbf{1}_m^\top \otimes I_\ell)y,\ y \in \partial\imath_{\mathbb{R}^{\ell m}_+}(\lambda)\}$.

Proof Suppose $q_0 \in \partial\imath_{\mathbb{R}^\ell_+}(\lambda_0)$, i.e., $\forall z_0 \in \mathbb{R}^\ell_+$, $q_0^\top(z_0 - \lambda_0) \leq 0$. Thus $\forall z \in \mathbb{R}^{\ell m}_+$, $q^\top(z-\lambda) = \sum_{i\in\mathcal{I}_m} q_0^\top(z_i-\lambda_0) \leq 0$ as each addend is nonnegative, which is exactly the conclusion (2). From the definition of q, $q_0 = (\mathbf{1}_m^\top \otimes I_\ell)(1/m)q$ for $(1/m)q \in \imath_{\mathbb{R}^{\ell m}_+}(\lambda)$, the conclusion (3) follows.

Suppose $q_0 \in (\mathbf{1}_m^\top \otimes I_\ell)\partial\imath_{\mathbb{R}^{\ell m}_+}(\lambda)$, i.e., there exists $y \in \mathbb{R}^{\ell m}$ such that $q_0 = (\mathbf{1}_m^\top \otimes I_\ell)y$ and $y^\top(z - \lambda) \leq 0$, $\forall z \in \mathbb{R}^{\ell m}_+$. As a special case of z, $y^\top(\mathbf{1}_m \otimes z_0 - \lambda) \leq 0$ always holds for every $z_0 \in \mathbb{R}^\ell_+$. Thus $\forall z_0 \in \mathbb{R}^\ell_+$, $q_0^\top(z_0 - \lambda_0) = y^\top(\mathbf{1}_m \otimes (z_0 - \lambda_0)) = y^\top(\mathbf{1}_m \otimes z_0 - \lambda) \leq 0$, that is the conclusion (1).

Suppose $q \in \imath_{\mathbb{R}^{\ell m}_+}(\lambda)$. Thus for $z = \mathbf{1}_m \otimes z_0$, $\forall z_0 \in R^\ell_+$, by definition of $\partial\imath_{\mathbb{R}^{\ell m}_+}(\lambda)$, there holds $(\mathbf{1}_m \otimes q_0)^\top(z-\lambda) = mq_0^\top(z_0-\lambda_0) \leq 0$, which leads to that $q_0^\top(z_0-\lambda_0) \leq 0$ holds for any $z_0 \in R^\ell_+$, that is the conclusion (1). ■

The following Lemma 5.4.2 establishes the one-to-one correspondence between the original Lagrange function (5.4) and the new formulation with the consensus graph introduced.

Lemma 5.4.2 *The pair (x^*, λ_0^*) is saddle point of L_a in (5.4) if and only if (x^*, ξ^*, λ^*) is saddle point of L_c in (5.9) with $\lambda^* = \mathbf{1}_m \otimes \lambda_0^*$ for some ξ^*.*

Proof The saddle differentials corresponding to L_a in (5.4) and L_c in (5.9) are:

$$T_{L_a}(x, \lambda_0) = \begin{bmatrix} \partial f(x) + \tilde{F}^\top(\mathbf{1}_m \otimes \lambda_0) \\ (\mathbf{1}_m^\top \otimes I_\ell)(c - \tilde{F}x) + \partial\imath_{\mathbb{R}^\ell_+}(\lambda_0) \end{bmatrix},$$

$$T_{L_c}(x,\xi,\lambda) = \begin{bmatrix} \partial f(x) + \tilde{F}^\top \lambda \\ (W \otimes I_\ell)\lambda \\ c - \tilde{F}x + \partial \imath_{\mathbb{R}^{\ell m}_+}(\lambda) - (W^\top \otimes I_\ell)\xi \end{bmatrix}.$$

Thus the conclusion becomes $0 \in T_{L_a}(x^*, \lambda_0^*)$ if and only if $0 \in T_{L_c}(x^*, \xi^*, \lambda^*)$.

Suppose $0 \in T_{L_c}(x^*, \xi^*, \lambda^*)$. From its second row $(W \otimes I_\ell)\lambda^* = 0$, we have $\lambda^* = \mathbf{1}_m \otimes \lambda_0^*$ for some λ_0^* and thus the first row of $0 \in T_{L_c}$ directly implies $0 \in \partial f(x^*) + \tilde{F}^\top(\mathbf{1}_m \otimes \lambda_0^*)$, which is the first row of $0 \in T_{L_a}$. With $\mathbf{1}_m^\top \otimes I_\ell$ multiplied on the left by the third row of $0 \in T_{L_c}$, we have

$$\begin{aligned} &(\mathbf{1}_m^\top \otimes I_\ell)(c - \tilde{F}x^*) + (\mathbf{1}_m^\top \otimes I_\ell)\partial \imath_{\mathbb{R}^{\ell m}_+}(\mathbf{1}_m \otimes \lambda_0^*) \\ &\quad \ni (\mathbf{1}_m^\top \otimes I_\ell)(W^\top \otimes I_\ell)\xi^* = (W\mathbf{1}_m \otimes I_\ell)^\top \xi^* = 0 \end{aligned}$$

by using the fact that W's null space $\mathcal{N}(W) = \{w\mathbf{1}_m \,|\, w \in \mathbb{R}\}$. Together with the equivalence of (1) and (3) in Lemma 5.4.1, the second row of $0 \in T_{L_a}$ directly follows from the expression above.

Suppose $0 \in T_{L_a}(x^*, \lambda_0^*)$. By letting $\lambda^* = \mathbf{1}_m \otimes \lambda_0^*$, the second row of $0 \in T_{L_c}$ trivially holds and the first row is resulted from the first row of $0 \in T_{L_a}$. From the second row of $0 \in T_{L_a}$, we have $q_0^* := (\mathbf{1}_m^\top \otimes I_\ell)(-c + \tilde{F}x^*) \in \partial \imath_{\mathbb{R}^\ell_+}(\lambda_0^*)$. Combined with the equivalence of (1) and (2) in Lemma 5.4.1, it leads to $q^* \in \partial \imath_{\mathbb{R}^{\ell m}_+}(\lambda^*)$ with $q^* := (1/m)\mathbf{1}_m \otimes q_0^*$. Then

$$(\mathbf{1}_m^\top \otimes I_\ell)(c - \tilde{F}x^* + q^*) = (\mathbf{1}_m^\top \otimes I_\ell)(c - \tilde{F}x^*) + q_0^* = 0. \tag{5.10}$$

On the other hand, because of $\mathcal{N}(W \otimes I_\ell) = \{w\mathbf{1}_m \otimes I_\ell \,|\, w \in \mathbb{R}\}$ and the space relationship $\mathcal{R}(A^\top) \perp \mathcal{N}(A)$ for any matrix A, the range space $\mathcal{R}\left(W^\top \otimes I_\ell\right) = \{y \in \mathbb{R}^{\ell m} \,|\, (\mathbf{1}_m^\top \otimes I_\ell)y = 0\}$, implying that $c - \tilde{F}x^* + q^* \in \mathcal{R}\left(W^\top \otimes I_\ell\right)$ according to (5.10). Therefore, there exists ξ^* such that $c - \tilde{F}x^* + q^* - (W^\top \otimes I_\ell)\xi^* = 0$, which implies the third row of $0 \in T_{L_c}$. ■

With the same $\mathbf{x}_i$ and $\mathbf{x}$ defined in Section 4.1, the $L_c(x,\xi,\lambda)$ in (5.9) can be transformed to the following equivalent form

$$\begin{aligned} L_c(\mathbf{x},\xi,\lambda) &= f(\mathbf{x}) + \lambda^\top(\tilde{F}\tilde{S}\mathbf{x} - c) - \imath_{\mathbb{R}_+^{\ell m}}(\lambda) + \xi^\top(W\otimes I_\ell)\lambda + \imath_{\mathcal{A}_x}(\mathbf{x}) \\ &= \sum_{i\in\mathcal{I}_m}\left(f_i(\mathbf{x}_i) + \lambda_i^\top(F_i\tilde{S}_i\mathbf{x}_i - c_i) - \imath_{\mathbb{R}_+^{\ell}}(\lambda_i) + \sum_{j\in\mathcal{N}_i} w_{ij}\xi_i^\top(\lambda_i - \lambda_j)\right) + \imath_{\mathcal{A}_x}(\mathbf{x}) \quad (5.11) \end{aligned}$$

whose saddle differential operator is

$$T_{L_c}(\mathbf{x},\xi,\lambda) = \begin{bmatrix} \partial f(\mathbf{x}) + (\tilde{F}\tilde{S})^\top\lambda + \partial\imath_{\mathcal{A}_x}(\mathbf{x}) \\ (W\otimes I_\ell)\lambda \\ -\tilde{F}\tilde{S}\mathbf{x} + c + \partial\imath_{\mathbb{R}_+^{\ell m}}(\lambda) - (W^\top\otimes I_\ell)\xi \end{bmatrix}.$$

The set of saddle points of $L_c(\mathbf{x},\xi,\lambda)$ is the same as the zero set of T_{L_c} scaled by 2, which can be split as follows:

$$\begin{aligned} 2T_{L_c} =&\ T_{L_c}^1 + T_{L_c}^2 \quad (5.12) \\ :=& \begin{bmatrix} 2\partial f(\mathbf{x}) + (\tilde{F}\tilde{S})^\top\lambda \\ (W\otimes I_\ell)\lambda \\ -\tilde{F}\tilde{S}\mathbf{x} + c + \partial\imath_{\mathbb{R}_+^{\ell m}}(\lambda) - (W^\top\otimes I_\ell)\xi \end{bmatrix} + \begin{bmatrix} 2\partial\imath_{\mathcal{A}_x}(\mathbf{x}) + (\tilde{F}\tilde{S})^\top\lambda \\ (W\otimes I_\ell)\lambda \\ -\tilde{F}\tilde{S}\mathbf{x} + c + \partial\imath_{\mathbb{R}_+^{\ell m}}(\lambda) - (W^\top\otimes I_\ell)\xi \end{bmatrix}. \end{aligned}$$

Note that except for the subdifferential term in their first rows, the terms $T_{L_c}^1$ and $T_{L_c}^2$ are the same. This feature to be shown later will help to reduce the communication requirements when distributing the computation burden among agents.

Lemma 5.4.3 *The operators $T_{L_c}^1$ and $T_{L_c}^2$ are maximal monotone. So is T_{L_c}.*

Proof The operator $T_{L_c}^1$ can be split as

$$T_{L_c}^1 = T_\partial + A\begin{bmatrix}\mathbf{x}^\top & \xi^\top & \lambda^\top\end{bmatrix}^\top := \begin{bmatrix} 2\partial f(\mathbf{x}) \\ 0 \\ c + \partial\imath_{\mathbb{R}_+^{\ell m}}(\lambda)\end{bmatrix} + \begin{bmatrix} 0 & 0 & (\tilde{F}\tilde{S})^\top \\ 0 & 0 & W\otimes I_\ell \\ -\tilde{F}\tilde{S} & -W^\top\otimes I_\ell & 0\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \xi\\ \lambda\end{bmatrix}.$$

Since both $f(\mathbf{x})$ and $\imath_{\mathbb{R}_+^{\ell m}}(\lambda)$ are convex, closed and proper (CCP) functions, the operator T_∂ is maximal monotone. As the matrix A is skew-symmetric $A + A^\top = 0$,

the second term is also maximal monotone. Therefore their sum $T^1_{L_c}$ is maximal monotone.

The same reasoning applies to $T^2_{L_c}$ with $\partial f(\mathbf{x})$ replaced by $\partial \imath_{\mathcal{A}_x}(\mathbf{x})$. Thus, T_{L_c} as their sum is maximal monotone, too. ■

Although $T^1_{L_c}$ and $T^2_{L_c}$ look mush simpler than T_{L_c}, each term could not be evaluated easily. Next we define the positive definite matrix $P = P^\top \succ 0$ to make the computation doable:

$$P = \begin{bmatrix} P_1 & 0 & (\tilde{F}\tilde{S})^\top \\ 0 & P_2 & W \otimes I_\ell \\ \tilde{F}\tilde{S} & W^\top \otimes I_\ell & P_3 \end{bmatrix}, \tag{5.13}$$

where the diagonal blocks are defined as

$$\begin{aligned} P_1 &:= \operatorname{diag}\left\{ \frac{1}{\rho_1} I_{N_1}, \ldots, \frac{1}{\rho_m} I_{N_m} \right\}, \\ P_2 &:= \operatorname{diag}\{1/\check{\epsilon}_1, \ldots, 1/\check{\epsilon}_m\} \otimes I_\ell, \\ P_3 &:= \operatorname{diag}\{1/\hat{\epsilon}_1, \ldots, 1/\hat{\epsilon}_m\} \otimes I_\ell, \end{aligned}$$

with positive constant scalars $\rho_i, \check{\epsilon}_i, \hat{\epsilon}_i, i \in \mathcal{I}_m$. The following Lemma 5.4.4 presents a sufficient but relatively conservative condition to guarantee a positive define P, where only the local information from each agent and its neighbors in the consensus graph is used.

Lemma 5.4.4 *Suppose for each $i \in \mathcal{I}_m$, the parameters ρ_i, $\check{\epsilon}_i$, and $\hat{\epsilon}_i$ satisfy the following conditions:*

$$0 < \rho_i < 1 \Big/ \|F_i\|, \tag{5.14a}$$

$$0 < \check{\epsilon}_i < 1 \Big/ \left(2 d^+_{ci}\right), \tag{5.14b}$$

$$0 < \hat{\epsilon}_i < 1 \Big/ \left(\|F_i\| + d^+_{ci} + d^-_{ci}\right). \tag{5.14c}$$

Then the symmetric matrix P defined in (5.13) *is positive definite.*

Proof The rows and columns of matrix P can be partitioned in the same way conforming to the variables $\mathbf{x}_1, \ldots, \mathbf{x}_m, \xi_1, \ldots, \xi_m, \lambda_1, \ldots, \lambda_m$ into $3m \times 3m$ blocks, and denote by P_{ij} the block in the i-th row and j-th column. Then the diagonal blocks are of the form $(1/\tau)I$ with proper dimensions for $\tau \in \{\rho_i, \check{\epsilon}_i, \hat{\epsilon}_i, i \in \mathcal{I}_m\}$ and thus are M-matrices (defined in Chapter 2).

With $\tilde{S}_i$ defined after (5.7), we have $\|F_i\| = \|F_i\tilde{S}_i\| = \|(F_i\tilde{S}_i)^\top\|$. Therefore the first condition (5.14a) results in

$$\|\frac{1}{\rho_i}I\| > \|(F_i\tilde{S}_i)^\top\|,\ i = 1, \ldots, m;$$

the second condition (5.14b) leads to

$$\|\frac{1}{\check{\epsilon}_i}I\| > 2d_{ci}^+ = W_{ii} + \sum_{j\in\mathcal{N}_i^+} \||W_{ij}|I\|,\ i = m+1, \ldots, 2m;$$

and the last one (5.14c) implies

$$\|\frac{1}{\hat{\epsilon}_i}I\| > \|F_i\| + d_{ci}^+ + d_{ci}^- = \|F_i\tilde{S}_i\| + W_{ii} + \sum_{k\in\mathcal{N}_i^-} \||W_{ki}|I\|,\ i = 2m+1, \ldots, 3m.$$

The three inequalities above together guarantee each row of blocks satisfying

$$\|P_{ii}^{-1}\|^{-1} > \sum_{j\in\mathcal{I}_{3m}, j\neq i} \|P_{ij}\|,\ i \in \mathcal{I}_{3m},$$

implying that the matrix P is block strictly diagonally dominant (see definition in Appendix). By Theorem 2.6.1 in Appendix, the matrix $P = P^\top$ is positive definite. ■

Next we show how to use P to evaluate the generalized resolvents of $T_{L_c}^1$ and $T_{L_c}^2$.

Lemma 5.4.5 *With $T_{L_c}^1$ and P defined in* (5.12) *and* (5.13)*, respectively, the generalized resolvent $(\mathbf{x}', \xi', \lambda') = J_{T_{L_c}^1}^P(\mathbf{x}, \xi, \lambda)$ can be evaluated as*

$$\lambda' = \Pi_{\mathbb{R}_+^{\ell m}}\left(\lambda + P_3^{-1}\big((\tilde{F}\tilde{S}\mathbf{x} - c + (W^\top \otimes I_\ell)\xi\big)\right), \tag{5.15a}$$

$$\mathbf{x}' = J_{\partial f}^{P_1/2}\left(\mathbf{x} - P_1^{-1}(\tilde{F}\tilde{S})^\top\Lambda\right), \tag{5.15b}$$

$$\xi' = \xi - P_2^{-1}(W \otimes I_\ell)\Lambda, \tag{5.15c}$$

with $\Lambda_i := 2\lambda_i' - \lambda_i$ *and* $\Lambda := (\Lambda_1, \ldots, \Lambda_m)$. *Equivalently, the distributed implementation is: for* $i \in \mathcal{I}_m$,

$$\lambda_i' = g_{\lambda_i}\big(\lambda_i, x_i, \xi_i, (\xi_k)_{k\in\mathcal{N}_{ci}^-}\big) := \Pi_{\mathbb{R}_+^\ell}\Big(\lambda_i + \hat{\epsilon}_i\big(F_i x_i - c_i + W_{ii}\xi_i - \sum_{k\in\mathcal{N}_{ci}^-} w_{ki}\xi_k\big)\Big), \tag{5.16a}$$

$$\mathbf{x}_i' = J_{2\rho_i \partial f_i}\left(\mathbf{x}_i - \rho_i (F_i\tilde{S}_i)^\top \Lambda_i\right), \tag{5.16b}$$

$$\xi_i' = \xi_i - \check{\epsilon}_i \sum_{j\in\mathcal{N}_{ci}^+} w_{ij}\left(\Lambda_i - \Lambda_j\right). \tag{5.16c}$$

For $T_{L_c}^2$ *in* (5.12) *and the same* P, $J^P_{T^2_{L_c}}$ *can be computed in the same way but with* (5.15b) *and* (5.16b) *replaced by the following* (5.17) *and* (5.18), *respectively,*

$$\mathbf{x}' = J^{P_1/2}_{\partial \imath_{\mathcal{A}_x}}\left(\mathbf{x} - P_1^{-1}(\tilde{F}\tilde{S})^\top \Lambda\right), \tag{5.17}$$

$$x_{ik}' = x_i' = \frac{(x_i - \rho_i F_i^\top \Lambda_i)/\rho_i + \sum_{k\in\mathcal{N}_i^-} x_{ik}/\rho_k}{1/\rho_i + \sum_{k\in\mathcal{N}_i^-} 1/\rho_k}, \forall k \in \mathcal{N}_i^-, \tag{5.18}$$

Proof By definition of the preconditioned resolvent J_T^P in (2.7), we have $P(\mathbf{x}, \xi, \lambda) \in P(\mathbf{x}', \xi', \lambda') + T_{L_c}^1(\mathbf{x}', \xi', \lambda')$, which can be simplified as

$$\begin{bmatrix} \mathbf{x} - P_1^{-1}(\tilde{F}\tilde{S})^\top(2\lambda' - \lambda) \\ \xi - P_2^{-1}(W\otimes I_\ell)(2\lambda' - \lambda) \\ \lambda + P_3^{-1}\Big(\tilde{F}\tilde{S}\mathbf{x} - c + (W^\top \otimes I_\ell)\xi\Big) \end{bmatrix} \in \begin{bmatrix} \mathbf{x}' + (P_1/2)^{-1}\partial f(\mathbf{x}') \\ \xi' \\ \lambda' + P_3^{-1}\partial \imath_{\mathbb{R}_+^{\ell m}}(\lambda') \end{bmatrix}.$$

According to the definition of preconditioned resolvent, the third row results in

$$\lambda' = J^{P_3}_{\partial \imath_{\mathbb{R}_+^{\ell m}}}\Big(\lambda + P_3^{-1}\big(\tilde{F}\tilde{S}\mathbf{x} - c + (W^\top \otimes I_\ell)\xi\big)\Big).$$

Since $\imath_{\mathbb{R}_+^n}(x) = \sum_{i\in\mathcal{I}_n} \imath_{R_+}(x_i)$, the expression (5.15a) follows from Remark 2.3.1(b). Using the definition of preconditioned resolvent, the first row leads to the conclusion (5.15b). And (5.15c) can be derived from the second row. The distributed implementations (5.16) and (5.18) follow from Lemma 2.3.1. ■

From the distributed implementation (5.16) and (5.18) in Lemma 5.4.5, the resolvents $J^P_{T^1_{L_c}}$ and $J^P_{T^2_{L_c}}$ can be computed using only information from neighboring agents

in the dependency graph $(\mathcal{I}_m, \mathcal{E})$ as well as the consensus graph $(\mathcal{I}_m, \mathcal{E}_c)$. Particularly, to compute ξ_i in (5.16c), agent i receives the information $\Lambda_j = 2\lambda'_j - \lambda_j$ from every in-neighbor $j \in \mathcal{N}_{ci}^+$ in the consensus graph. However, for its in-neighbor $j \in \mathcal{N}_{ci}^+$ to update its own λ_j in (5.16a), agent i as one of agent j's out-neighbor needs to send back ξ_i to agent j. Therefore, in addition to the bidirectional communication Assumption 4.1.3 on $(\mathcal{I}_m, \mathcal{E})$, we impose the following Assumption 5.4.2 about $(\mathcal{I}_m, \mathcal{E}_c)$.

Assumption 5.4.2 *The communication in the consensus graph $(\mathcal{I}_m, \mathcal{E}_c)$ is bidirectional, i.e., the communication graph is the union of $(\mathcal{I}_m, \mathcal{E}_c)$ and its transpose $(\mathcal{I}_m, \mathcal{E}_c^\top)$.*

With Assumption 5.4.2, we are able to adopt the generalized Douglas-Rachford algorithm (2.8) to obtain our distributed implementation (in a compact form) as follows:

$$\bar{\mathbf{z}}^t = J^P_{T^2_{L_c}}(\mathbf{z}^t), \tag{5.19a}$$

$$\mathbf{z}^{t+1} = \mathbf{z}^t + 2\alpha\left(J^P_{T^1_{L_c}}(2\bar{\mathbf{z}}^t - \mathbf{z}^t) - \bar{\mathbf{z}}^t\right), \tag{5.19b}$$

with the constant $\alpha \in (0,1)$, $\mathbf{z}^t = (\mathbf{x}^t, \xi^t, \lambda^t)$ and $\bar{\mathbf{z}}^t = (\bar{\mathbf{x}}^t, \bar{\xi}^t, \bar{\lambda}^t)$ for $t \geq 0$. Using the distributed computation of $J^P_{T^1_{L_c}}$ and $J^P_{T^2_{L_c}}$ in Lemma 5.4.5, the detailed implementation of (5.19) is summarized in the following Algorithm 10, where $\bar{\bar{\mathbf{z}}}^t = (\bar{\bar{\mathbf{x}}}^t, \bar{\bar{\xi}}^t, \bar{\bar{\lambda}}^t) := 2\bar{\mathbf{z}}^t - \mathbf{z}^t$ and $\tilde{\mathbf{z}}^t = (\tilde{\mathbf{x}}^t, \tilde{\xi}^t, \tilde{\lambda}^t) := J^P_{T^1_{L_c}}(\bar{\bar{\mathbf{z}}}^t)$ denote the intermediate values.

In Algorithm 10, the first two **for** loops carry out the computation (5.19a) plus the step $\bar{\bar{\mathbf{z}}}^t = 2\bar{\mathbf{z}}^t - \mathbf{z}^t$ and the left two loops implement the rest of (5.19b). These four loops must be executed in sequence while within each loop, the computation is completed by each agent in parallel. The information is exchanged: a) in steps 6, 8 between each agent and its out-neighbors in the dependency graph $(\mathcal{I}_m, \mathcal{E})$, same as that in Algorithm 5; b) in steps 4, 12 between each agent and its out-neighbors in the consensus graph $(\mathcal{I}_m, \mathcal{E}_c)$, and in steps 9, 17 between each agent and its in-neighbors also in $(\mathcal{I}_m, \mathcal{E}_c)$. Therefore the first two **for** loops are relatively communication-extensive.

The convergence of Algorithm 10 is established in the following Theorem 5.4.1.

Algorithm 10 Synchronous Coordinate-Free Algorithm

Initialize $\mathbf{x}^0$, ξ^0, λ^0, and let $t \leftarrow 0$.
repeat
  for $i = 1, \ldots, m$ **do**
    $\bar{\lambda}_i^t \leftarrow g_{\lambda_i}\big(\lambda_i^t, x_i^t, \xi_i^t, (\xi_k^t)_{k\in\mathcal{N}_{ci}^-}\big)$
    $\bar{\bar{\lambda}}_i^t = \bar{\Lambda}_i^t \leftarrow 2\bar{\lambda}_i^t - \lambda_i^t$
    $\bar{x}_i^t \leftarrow \frac{(x_i - \rho_i F_i^\top \Lambda_i)/\rho_i + \sum_{k\in\mathcal{N}_i^-} x_{ik}/\rho_k}{1/\rho_i + \sum_{k\in\mathcal{N}_i^-} 1/\rho_k}$
  end for
  for $i = 1, \ldots, m$ **do**
    $\bar{x}_{ik}^t \leftarrow \bar{x}_i^t,\ \forall k \in \mathcal{N}_i^-$
    $\bar{\xi}_i^t \leftarrow \xi_i^t - \check{\epsilon}_i \sum_{j\in\mathcal{N}_{ci}^+} w_{ij}\left(\bar{\Lambda}_i^t - \bar{\Lambda}_j^t\right)$
    $(\bar{\bar{\mathbf{x}}}_i^t, \bar{\bar{\xi}}_i^t) \leftarrow 2(\bar{\mathbf{x}}_i^t, \bar{\xi}_i^t) - (\mathbf{x}_i^t, \xi_i^t)$
  end for
  for $i = 1, \ldots, m$ **do**
    $\tilde{\lambda}_i^t \leftarrow g_{\lambda_i}\big(\bar{\bar{\lambda}}_i^t, \bar{\bar{x}}_i^t, \bar{\bar{\xi}}_i^t, (\bar{\bar{\xi}}_k^t)_{k\in\mathcal{N}_{ci}^-}\big)$
    $\tilde{\Lambda}_i^t \leftarrow 2\tilde{\lambda}_i^t - \bar{\bar{\lambda}}_i^t$
    $\tilde{\mathbf{x}}_i^t \leftarrow J_{2\rho_i \partial f_i}\left(\bar{\bar{\mathbf{x}}}_i - \rho_i (F_i \tilde{S}_i)^\top \tilde{\Lambda}_i\right)$
    $(\mathbf{x}_i^{t+1}, \lambda_i^{t+1}) \leftarrow (\mathbf{x}_i^t, \lambda_i^t) + 2\alpha\big((\tilde{\mathbf{x}}_i^t, \tilde{\lambda}_i^t) - (\bar{\mathbf{x}}_i^t, \bar{\lambda}_i^t)\big)$
  end for
  for $i = 1, \ldots, m$ **do**
    $\tilde{\xi}_i^t \leftarrow \bar{\bar{\xi}}_i^t - \check{\epsilon}_i \sum_{j\in\mathcal{N}_{ci}^+} w_{ij}\left(\tilde{\Lambda}_i^t - \tilde{\Lambda}_j^t\right)$
    $\xi_i^{t+1} \leftarrow \xi_i^t + 2\alpha(\tilde{\xi}_i^t - \bar{\xi}_i^t)$
  end for
  $t \leftarrow t + 1$
until $\|(\bar{\mathbf{x}}^t, \bar{\xi}^t, \bar{\lambda}^t) - (\bar{\mathbf{x}}^{t-1}, \bar{\xi}^{t-1}, \bar{\lambda}^{t-1})\|$ is sufficiently small
return $\bar{\mathbf{x}}^t$

Theorem 5.4.1 *Suppose Assumptions 4.1.1, 5.1.1, 5.3.1 and 5.4.1 hold, and the positive constant scalars* $\rho_i, \check{\epsilon}_i, \hat{\epsilon}_i, i \in \mathcal{I}_m$, *are chosen such that the matrix* P *defined in* (5.13) *is positive definite. The sequence* $\{(\bar{\mathbf{x}}^t, \bar{\xi}^t, \bar{\lambda}^t)\}$ *generated by Algorithm 10 (or* (5.19)*) will converge to a saddle point* $(\bar{\mathbf{x}}^*, \bar{\xi}^*, \bar{\lambda}^*)$ *of* L_c *in* (5.9) *and the corresponding* x^* *is a solution to Problem* (5.3).

Proof The Algorithm (5.19) is derived by applying the Douglas-Rachford splitting method. As stated after (2.8), for $P \succ 0$, the generated sequence will converge to one saddle point of $L_c(\mathbf{x}, \xi, \lambda)$. Following from Lemma 5.4.2, the corresponding x^* is a minimizer of Problem (5.3). ■

6. NUMERICAL EXAMPLES

This chapter summarizes the simulation results of the proposed algorithms applied to the linear equations/programs and network localization problems. In all these examples, the initial value x_{ij}^0 are set to be x_i^0 for all $i \in \mathcal{I}_m$ and $j \in \mathcal{N}_i^-$.

6.1 Linear Programs/Equations

We firstly apply Algorithms 1 and 3 to solve Example 3.2.1 with $\varepsilon = 0, 0.01, 0.5$, respectively, which is a linear equation when $\varepsilon = 0$ and linear programs otherwise. For comparison, in Algorithm 3 all agents are assumed to take part in both the projection and consensus operations for all rounds. The parameter α_i in the relaxed projection (3.11) is $\alpha_i = 1.5$, $i \in \{1, 2, 3\}$ in both Algorithms 1 and 3 and the weight matrices in (3.17) of Algorithm 3 are, for all $t = 1, 2, \ldots$

$$W_1^t = \begin{bmatrix} 1 \end{bmatrix}, \; W_2^t = \begin{bmatrix} 0.1 & 0.9 \\ 0.9 & 0.1 \end{bmatrix}, \; W_3^t = \begin{bmatrix} 0.04 & 0.48 & 0.48 \\ 0.48 & 0.04 & 0.48 \\ 0.48 & 0.48 & 0.04 \end{bmatrix}.$$

The results are shown in Fig. 6.1, where $\mathbf{x}^*$ is the augmented variable corresponding to the unique solution $x^* = (0, 2, -1)$ when $\varepsilon = 0$, and the converged feasible solution when $\varepsilon \neq 0$. As can be seen, Algorithm 3 with proper assigned weights converges significantly faster than Algorithm 1 in all cases.

When ϵ increases, it generates a larger feasible set for each agent and also a larger set of common feasible points. As can be seen in Fig. 6.1, when ϵ is larger, the convergence is slightly faster since it gets easier for agents to satisfy their local constraints. This feature is more obvious in Fig. 6.7 for network localization problem.

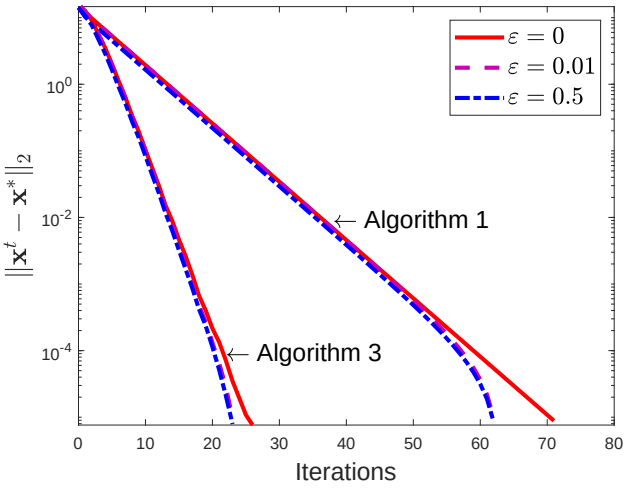


Fig. 6.1. Results of Example 3.2.1: plots of $\|\mathbf{x}^t - \mathbf{x}^*\|$ vs iterations t for Algorithms 1 and 3.

6.2 Network Localization Using Accurate AOA Information

In this section we consider the network localization problem using AOA information introduced in Section 3.2.2. A number of agents are randomly placed inside a planar region. Among them, some are anchors indexed by $\mathcal{I}_a$ who know their exact locations $(x_i^*)_{i\in\mathcal{I}_a}$, while the other are free agents indexed by $\mathcal{I}_f$ who need to estimate their positions $(x_i^*)_{i\in\mathcal{I}_f}$ based on the relative orientation measurements between pairs of agents within a certain measurement range. This is an instance of the convex feasibility problem and therefore a special case of convex optimization problems. If there are at least two anchors and the whole network has an infinitesimally rigid graph, then the network is localizable, i.e., there is a unique solution $(x_i^*)_{i\in\mathcal{I}_f}$ satisfying all the relative orientation constraints [10, Thm. 15].

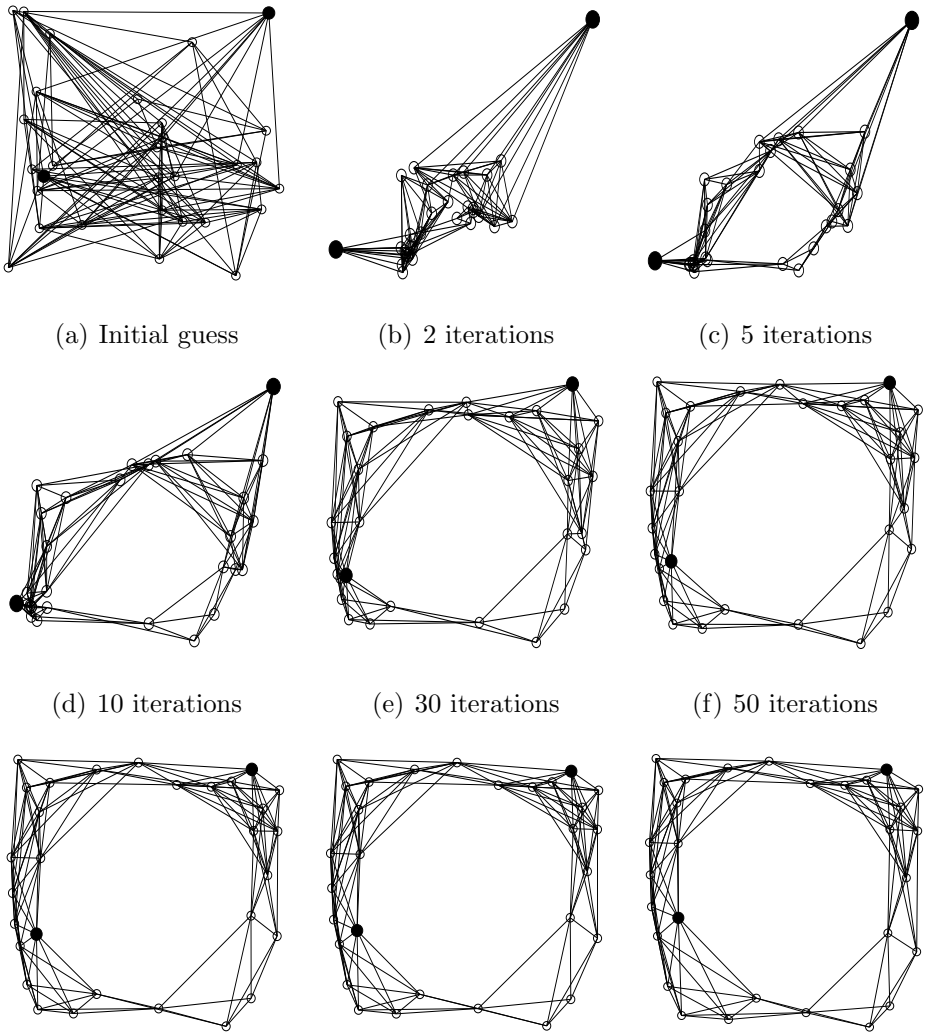
When the AOA information is accurate, i.e. $\delta = 0$, the whole network is infinitesimally rigid and therefore the solution is unique, which is named as the ground truth in the sequel.

In this case study, there are 30 agents with two anchors (the minimum number of necessary anchors). The initial guesses randomly generated for the free agents are shown in Fig. 6.2 (a) while the ground truth is depicted in sub-figure (i). In each sub-figure, edges represent the constraints couplings resulting from the relative orientation measurements and solids dots and small circles are anchors and free agents, respectively. In the rest of this section, Algorithms 1, 5 and 8 are applied to solve this problem and the iterative results will be compared with that of the well known algorithm Pro-Con from [27, 28] and ADMM.

Fig. 6.2 shows the iterative result of Algorithm 1. As can be seen, the algorithms converges after about 50 iterations to the ground truth.

Fig. 6.3 plots the convergence rates of Algorithm 1 with three different settings of α_i: $\alpha_i \equiv 0.5$, $\alpha_i \equiv 1$, $\alpha_i \equiv 1.9$ and Pro-Con algorithm in [27, 28]. For a fair comparison, the Pro-Con algorithm adopts equal weights as that in the consensus operation of Algorithm 1. At least for this example, regardless of α_i being used, Algorithm 1 converges much faster than the Pro-Con algorithm despite the fact that the later one demands each agent to store and exchange with neighbors a whole copy of the variable x, resulting in more information storage and communication for all agents. An intuitive explanation of the performance difference is as follows. In the Pro-Con algorithm, agent i maintains a copy of x. However, in the copy, only the part involved in the local constraint $\mathcal{F}_i$ will be updated/improved via the local projection step, while the other part remains unchanged but still gets delivered to neighboring agents for their consensus step, potentially hindering the algorithm. Regarding to Algorithm 1, at least in this case study, a larger value of α_i leads to a faster convergence rate. It is unclear whether this observation holds true for general problems. Our experience seems to suggest that using $\alpha_i \in (1, 2)$ generally induces faster convergence than using $\alpha_i \in (0, 1)$.

In Fig. 6.4, we compare the convergence rates of three synchronous algorithms, Algorithms 1, 5 and the ADMM algorithm (4.12). For Algorithm 5, the parameter α is set to 0.5, 0.7, 0.9, 0.98, respectively, while the parameter ρ has no effect on the



(a) Initial guess (b) 2 iterations (c) 5 iterations

(d) 10 iterations (e) 30 iterations (f) 50 iterations

(g) 80 iterations (h) 100 iterations (i) Ground truth

Fig. 6.2. Results of applying Algorithm 1 to the network localization problem with 2 anchors among 30 agents.

algorithm as each f_i is an indicator function. For the ADMM algorithm (4.12), $\rho = 0.01, 0.1, 1, 10, 1000$ are tested and the best result ($\rho = 0.1$) is plotted in Fig. 6.4. For Algorithm 1, the result with the best parameter value $\alpha_i \equiv 1.9$ based on experiments is included here. For this example, the Algorithm 5 converges at a similar rate as the ADMM algorithm (4.12) with less performance oscillations, but faster than Algorithm 1.

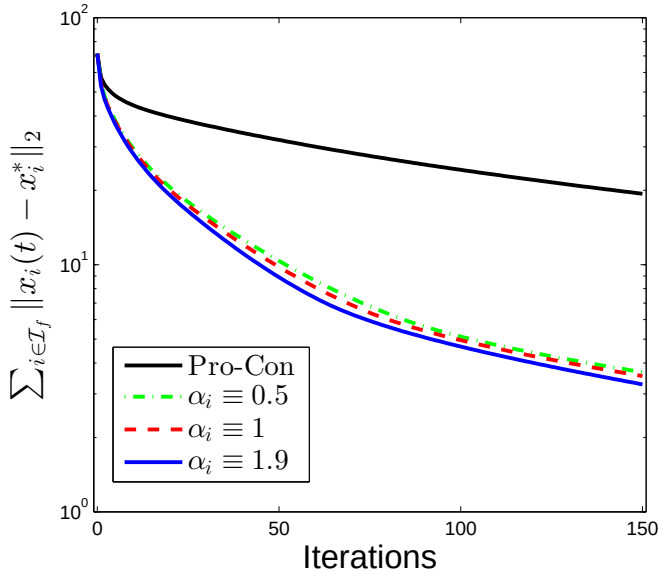


Fig. 6.3. Comparison of the convergence rates of Algorithm 1 with different α_i and the Pro-Con algorithm for the network localization problem. The value $\sum_{i\in\mathcal{I}_f} \|x_i^t - x_i^*\|_2$ versus iteration number t is plotted.

Fig. 6.5 shows representative random outcomes of applying Algorithm 8 to the same localization problem with $\alpha = 0.5$ and two different sets of probabilities: $p_i = 1/30$, and $p_i = D_i / \sum D_i$ where D_i is the degree of node i in the dependency graph. It is observed that the iterative results indeed converge to the optimal solution. Further, updating highly connected nodes more frequently does not seem to speed up convergence in this example. Note that, compared to Algorithm 5, the number of iterations needed for achieving the same convergence performance is much larger due to the fact that at each round only one agent is performing computation as opposed to 28 agents in Algorithm 5.

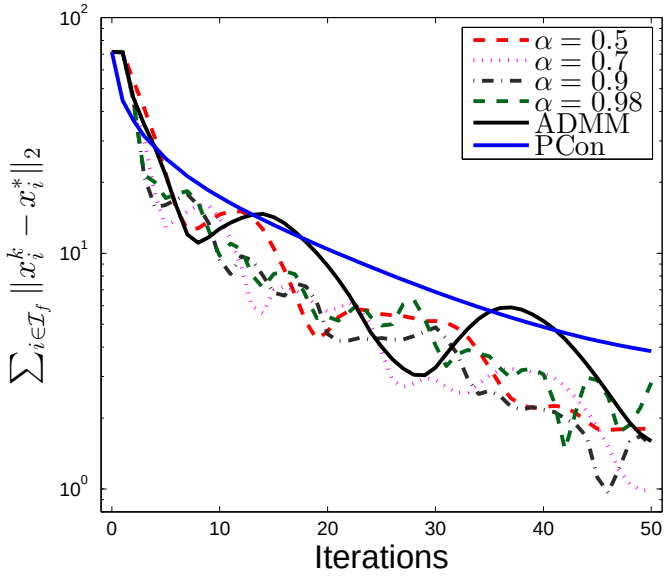


Fig. 6.4. Comparison of the convergence rates of Algorithm 5 with different α_i, ADMM algorithm (4.12) and Algorithm 1 (PCon in figure) for the network localization problem.

6.3 Network Localization Using Inaccurate AOA Information

In this section, consider the same setting as last section but the relative orientation measurements are not accurate ($\delta \neq 0$). With setting of $\delta = 0°$, $2.5°$, $4°$, $8°$, respectively, Fig. 6.7 provides the convergence rate of applying Algorithm 1 by plotting the sum of constraint violations $d\{\angle(x_j(k) - x_i(k)), \Theta_{ij}\}$ vs. iteration number k, where $d\{\cdot,\cdot\}$ denotes the angle difference between the estimation and its feasible sets (see Fig. 6.6). As can be seen in Fig. 6.7, with a larger error range δ and hence a larger feasible set, the algorithm converges faster to a feasible solution.

However, the converged feasible solution above is not the ground truth in general. When $\delta = 8°$, the evolvement of network localization results is shown in Fig. 6.8. With about 50 iterations, the localization results have converged to a common feasible point, which, however, is not the one that represents the ground truth.

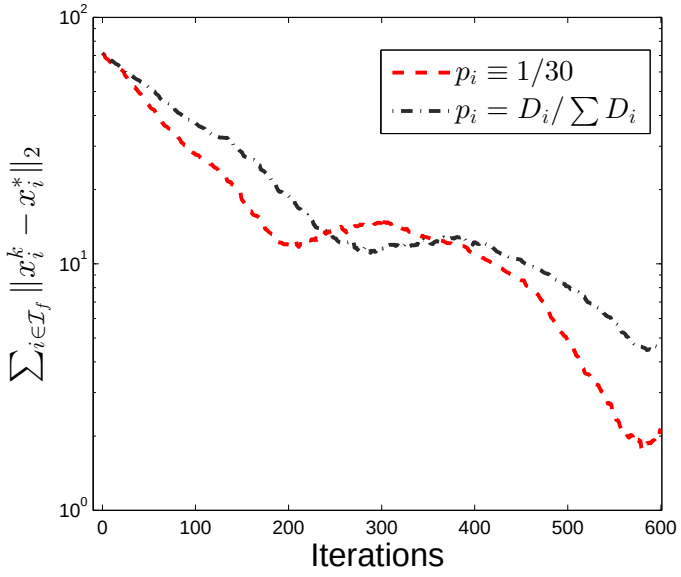


Fig. 6.5. Random outcomes of Algorithm 8 with $\alpha = 0.5$ and two different sets of probabilities p_i's.

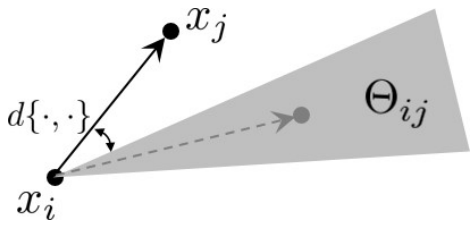


Fig. 6.6. Angle difference between the intermediate estimation and its feasible set. With accurate measurement, the constraint is a singleton (dashed line) which will expand to a cone (shading area) if the measurement has errors.

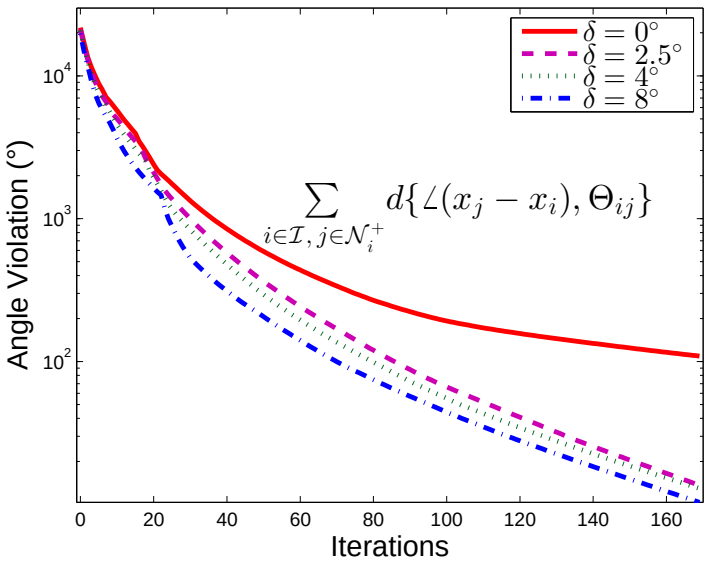
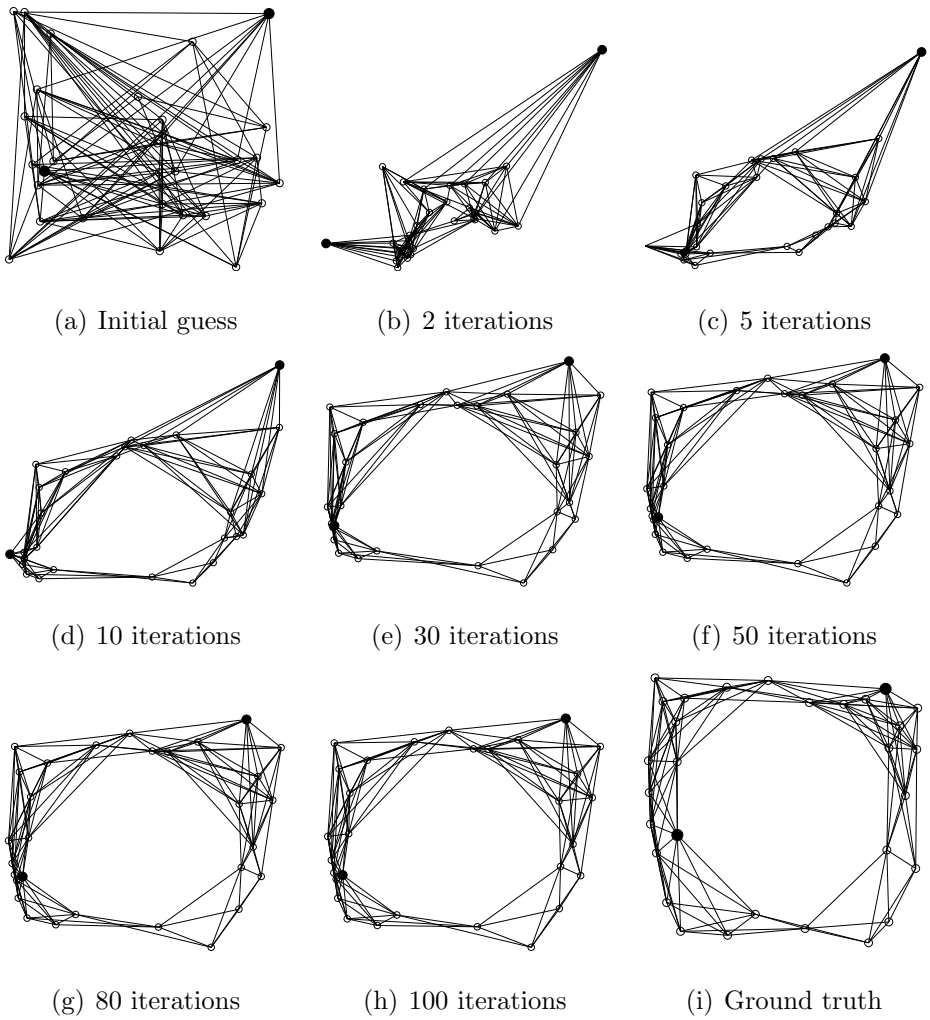


Fig. 6.7. Comparison of the convergence rates of Algorithm 1 for the network localization problem with measurement errors.



(a) Initial guess (b) 2 iterations (c) 5 iterations

(d) 10 iterations (e) 30 iterations (f) 50 iterations

(g) 80 iterations (h) 100 iterations (i) Ground truth

Fig. 6.8. Results of applying Algorithm 1 to the network localization problem using inaccurate AOA information with 2 anchors among 30 agents.

7. CONCLUSIONS

7.1 Main Results

This thesis studies the convex feasibility and optimization problems on agent networks. A series of distributed synchronous or asynchronous solution algorithms are developed based on paracontractions and operator splitting methods. The proposed algorithms have the following desired features:

a. Except for Algorithm (5.6) (the synchronous algorithm with a coordinator for optimization problems with local and global couplings), all the other algorithms are iterative and distributed in that all agents play an equal tole and no super nodes are needed. Compared to centralized algorithms, **distributed solutions** have better scalability as the size of networks grows and better resilience to single-point failures and attacks.

b. All of the algorithms adopt **fixed step sizes**. Therefore there is no need to tune step sizes or choose the proper scheme for diminishing step sizes.

c. As stated multiple times in this thesis, the variables maintained by each agent and the information exchanged between each agent and its in-neighbors (resp. out-neighbors) only include this agent's own variable and its desired values for in-neighboring agents (reps. the desired values on its own variable from out-neighboring agents). When the couplings across agents are sparse (i.e., each agent is coupled with only a few of agents), the **storage and communication** requirements for individual agents will be significantly reduced compared to the common setting where each agent holds and exchanges with neighbors a copy of the whole variable. This is the key feature of the algorithms developed in this thesis.

d. Following from the feature c. above, for agent $i \in \mathcal{I}_m$, each of its out-neighbors has only access to agent i's own variable x_i. Compared to the case where each agent exchanges the local copy of the whole variable x with neighbors, it will be relatively difficult for the out-neighbors to infer agent i's private information, e.g., the local constraints and objective functions. On the other hand, as each of its in-neighbor $j \in \mathcal{N}_i^+$ can only obtain the desired value x_{ji} from agent i, the in-neighbors will also have difficulty to reconstruct agent i's private information unless all of the in-neighbors cooperate as a group. This helps to protect the **privacy** of each agent, should such needs arise.

7.2 Future Works

The approach proposed in this thesis can be extended in multiple directions. Following are a few of important ones.

- **Networked convex-concave games with coupling constraints**

  Since the Lagrange functions of the constrained optimization problems studied in Chapters 4 and 5 are special cases of saddle functions, the proposed distributed solutions can be modified to solve the networked convex-concave games with coupling constraints where the coupling relations between agents are modeled by the dependency graph. Interested readers can refer to our work [85, 86] for details.

- **Asynchronous algorithms with bounded delay**

  The synchronous solutions discussed in this thesis require multiple rounds of synchronization within each iteration where each agent waits until all agents complete their computation for that round. This may lead to a large amount of computation resource waste when some of the agents are slow. In the case of asynchronous algorithms based on randomized implementation, only one agent is allowed to carry out its update at each iteration, resulting in each iteration

being relatively efficient. However, much more iterations are needed. Asynchronous algorithms taking bounded delay into account could be a good candidate to remedy the situation, where each agent can update utilizing the latest available information whenever it is ready.

- **Robustness to the absence of agents**

  In the synchronous algorithms, it is assumed that all agents remain functional all the time to carry out updates while the asynchronous algorithms are able to accommodate the temporary absence of some agents. The study on the algorithms' convergence behaviors when one or some agents are permanently lost (due to communication or breakdown) will be helpful to explore algorithms' robustness to the absence of agents.

- **Convergence rate**

  With the convergence analysis established so far, the algorithms can converge asymptotically to a solution. However, it remains to be established how fast the convergence can be. Especially, we would like to quantify the convergence rate of the proposed algorithms in terms of the convexity of the local constraints and objective functions as well as the connectivity of the dependency and consensus graphs.

REFERENCES

REFERENCES

[1] J. Ash and L. Potter, "Sensor network localization via received signal strength measurements with directional antennas," in *Proceedings of the 2004 Allerton Conference on Communication, Control, and Computing*, 2004, pp. 1861–1870.

[2] L. Doherty, L. El Ghaoui *et al.*, "Convex position estimation in wireless sensor networks," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3. IEEE, 2001, pp. 1655–1663.

[3] D. Blatt and A. O. Hero, "Energy-based sensor network source localization via projection onto convex sets," *IEEE Transactions on Signal Processing*, vol. 54, no. 9, pp. 3614–3619, 2006.

[4] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "A collaborative training algorithm for distributed learning," *IEEE Transactions on Information Theory*, vol. 55, no. 4, pp. 1856–1871, 2009.

[5] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[6] C. S. Raghavendra, K. M. Sivalingam, and T. Znati, *Wireless sensor networks*. Springer, 2006.

[7] D. Mellinger, M. Shomin, N. Michael, and V. Kumar, "Cooperative grasping and transport using multiple quadrotors," in *Distributed autonomous robotic systems*. Springer, 2013, pp. 545–558.

[8] G. Zhu and J. Hu, "Distributed network localization using angle-of-arrival information part i: Continuous-time protocol," in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 1006–1011.

[9] ——, "Distributed network localization using angle-of-arrival information part ii: Discrete-time algorithm and error analysis," in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 1000–1005.

[10] ——, "A distributed continuous-time algorithm for network localization using angle-of-arrival information," *Automatica*, vol. 50, no. 1, pp. 53–63, 2014.

[11] M. Cao, A. S. Morse, and B. D. Anderson, "Reaching a consensus in a dynamically changing environment: A graphical approach," *SIAM Journal on Control and Optimization*, vol. 47, no. 2, pp. 575–600, 2008.

[12] M. Martin, P. Klupar, S. Kilberg, and J. Winter, "Techsat 21 and revolutionizing space missions using microsatellites," 2001.

[13] H. Fiedler and G. Krieger, "Close formation flight of passive receiving microsatellites," in *18th International Symposium on Space Flight Dynamics*, vol. 548, 2004, p. 47.

[14] C. Sueur, J.-L. Deneubourg, and O. Petit, "From social network (centralized vs. decentralized) to collective decision-making (unshared vs. shared consensus)," *PLoS one*, vol. 7, no. 2, p. e32566, 2012.

[15] B. Buechel, T. Hellmann, and S. Klößner, "Opinion dynamics and wisdom under conformity," *Journal of Economic Dynamics and Control*, vol. 52, pp. 240–257, 2015.

[16] D. Acemoğlu, G. Como, F. Fagnani, and A. Ozdaglar, "Opinion fluctuations and disagreement in social networks," *Mathematics of Operations Research*, vol. 38, no. 1, pp. 1–27, 2013.

[17] S. Yang, S. Tan, and J.-X. Xu, "Consensus based approach for economic dispatch problem in a smart grid," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4416–4426, 2013.

[18] S. D. McArthur, E. M. Davidson, V. M. Catterson, A. L. Dimeas, N. D. Hatziargyriou, F. Ponci, and T. Funabashi, "Multi-agent systems for power engineering applicationspart i: Concepts, approaches, and technical challenges," *IEEE Transactions on Power systems*, vol. 22, no. 4, pp. 1743–1752, 2007.

[19] A. L. Dimeas and N. D. Hatziargyriou, "Operation of a multiagent system for microgrid control," *IEEE Transactions on Power systems*, vol. 20, no. 3, pp. 1447–1455, 2005.

[20] G. Tesauro, D. M. Chess, W. E. Walsh, R. Das, A. Segal, I. Whalley, J. O. Kephart, and S. R. White, "A multi-agent systems approach to autonomic computing," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*. IEEE Computer Society, 2004, pp. 464–471.

[21] A. M. Talib, R. Atan, R. Abdullah, and M. Azrifah, "Cloudzone: Towards an integrity layer of cloud data storage based on multi agent system architecture," in *2011 IEEE Conference on Open Systems*. IEEE, 2011, pp. 127–132.

[22] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-point algorithms for inverse problems in science and engineering*. Springer, 2011, pp. 185–212.

[23] M. Jiang and G. Wang, "Convergence studies on iterative algorithms for image reconstruction," *IEEE Transactions on Medical Imaging*, vol. 22, no. 5, pp. 569–579, 2003.

[24] A. Cichocki and S.-i. Amari, *Adaptive blind signal and image processing: learning algorithms and applications*. John Wiley & Sons, 2002, vol. 1.

[25] A. Bemporad, D. Bernardini, and P. Patrinos, "A convex feasibility approach to anytime model predictive control," *arXiv preprint arXiv:1502.07974*, 2015.

[26] X. Hou, Y. Xiao, J. Cai, J. Hu, and J. E. Braun, "Distributed model predictive control via proximal jacobian admm for building control applications," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 37–43.

[27] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[28] A. Nedić and J. Liu, "On convergence rate of weighted-averaging dynamics for consensus problems," *IEEE Transactions on Automatic Control*, vol. 62, no. 2, pp. 766–781, 2017.

[29] H. H. Bauschke and J. M. Borwein, "On projection algorithms for solving convex feasibility problems," *SIAM review*, vol. 38, no. 3, pp. 367–426, 1996.

[30] Y. Censor, W. Chen, P. L. Combettes, R. Davidi, and G. T. Herman, "On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints," *Computational Optimization and Applications*, vol. 51, no. 3, pp. 1065–1088, 2012.

[31] A. J. Zaslavski, "Convex feasibility problems," in *Approximate Solutions of Common Fixed-Point Problems*. Springer, 2016, pp. 341–384.

[32] 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.

[33] 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.

[34] W. Shi, Q. Ling, G. Wu, and W. Yin, "A proximal gradient algorithm for decentralized composite optimization," *IEEE Trans. Signal Processing*, vol. 63, no. 22, pp. 6013–6023, 2015.

[35] 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 Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[36] 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.

[37] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Processing*, vol. 58, no. 10, pp. 5262–5276, 2010.

[38] S. Mou and A. S. Morse, "A fixed-neighbor, distributed algorithm for solving a linear algebraic equation," in *2013 European Control Conference (ECC)*. IEEE, 2013, pp. 2269–2273.

[39] S. Mou, J. Liu, and A. S. Morse, "A distributed algorithm for solving a linear algebraic equation," *IEEE Transactions on Automatic Control*, vol. 60, no. 11, pp. 2863–2878, 2015.

[40] G. Shi and B. D. Anderson, "Distributed network flows solving linear algebraic equations," in *American Control Conference (ACC), 2016*. IEEE, 2016, pp. 2864–2869.

[41] H.-T. Cao, T. E. Gibson, S. Mou, and Y.-Y. Liu, "Impacts of network topology on the performance of a distributed algorithm solving linear equations," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 1733–1738.

[42] X. Wang, S. Mou, and D. Sun, "Improvement of a distributed algorithm for solving linear equations," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 4, pp. 3113–3117, 2017.

[43] X. Wang, J. Zhou, S. Mou, and M. J. Corless, "A distributed algorithm for least squares solutions," *IEEE Transactions on Automatic Control*, 2019.

[44] D. Fullmer, L. Wang, and A. S. Morse, "A distributed algorithm for computing a common fixed point of a family of paracontractions," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 552–557, 2016.

[45] D. Fullmer, J. Liu, and A. S. Morse, "An asynchronous distributed algorithm for computing a common fixed point of a family of paracontractions," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 2620–2625.

[46] J. Liu, D. Fullmer, A. Nedić, T. Başar, and A. S. Morse, "A distributed algorithm for computing a common fixed point of a family of strongly quasi-nonexpansive maps," in *American Control Conference (ACC), 2017*. IEEE, 2017, pp. 686–690.

[47] S. Khoshfetrat Pakazad, M. S. Andersen, and A. Hansson, "Distributed solutions for loosely coupled feasibility problems using proximal splitting methods," *Optimization Methods and Software*, vol. 30, no. 1, pp. 128–161, 2015.

[48] K. Lu, G. Jing, and L. Wang, "Distributed algorithms for solving convex inequalities," *IEEE Transactions on Automatic Control*, 2017.

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

[50] N. Parikh, S. Boyd *et al.*, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[51] Y. Lou, G. Shi, K. H. Johansson, and Y. Hong, "Approximate projected consensus for convex intersection computation: Convergence analysis and critical error angle," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1722–1736, 2014.

[52] R. Aharoni and Y. Censor, "Block-iterative projection methods for parallel computation of solutions to convex feasibility problems," *Linear Algebra Appl.*, vol. 120, pp. 165–175, 1989.

[53] Y. Censor, D. Gordon, and R. Gordon, "Component averaging: An efficient iterative parallel algorithm for large and sparse unstructured problems," *Parallel computing*, vol. 27, no. 6, pp. 777–808, 2001.

[54] S.-S. Chang, J. Kim, and X. Wang, "Modified block iterative algorithm for solving convex feasibility problems in banach spaces," *Journal of Inequalities and Applications*, vol. 2010, no. 1, p. 869684, 2010.

[55] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," *IEEE Trans. Robot. Automat.*, vol. 17, no. 6, pp. 947–951, 2001.

[56] X. Gao, J. Liu, and T. Başar, "Stochastic communication-efficient distributed algorithms for solving linear algebraic equations," in *2016 IEEE Conference on Control Applications (CCA)*. IEEE, 2016, pp. 380–385.

[57] S. Mou, Z. Lin, L. Wang, D. Fullmer, and A. S. Morse, "A distributed algorithm for efficiently solving linear equations and its applications (special issue jcw)," *Systems & Control Letters*, vol. 91, pp. 21–27, 2016.

[58] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert Spaces*, 2nd ed. Springer, 2017.

[59] E. K. Ryu and S. Boyd, "A primer on monotone operator methods," *Appl. Comput. Math*, vol. 15, no. 1, pp. 3–43, 2016.

[60] E. Wei and A. Ozdaglar, "On the O(1/k)convergence of asynchronous distributed alternating direction method of multipliers," in *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, 2013, pp. 551–554.

[61] P. Bianchi, W. Hachem, and F. Iutzeler, "A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization," *IEEE Trans. Automatic Control*, vol. 61, no. 10, pp. 2947–2957, 2016.

[62] Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin, "Coordinate friendly structures, algorithms and applications," *arXiv Preprint 1601.00863*, 2016.

[63] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous distributed optimization using a randomized alternating direction method of multipliers," in *IEEE Int. Conf. Decision and Control*. IEEE, 2013, pp. 3671–3676.

[64] T. Wu, K. Yuan, Q. Ling, W. Yin, and A. H. Sayed, "Decentralized consensus optimization with asynchrony and delays," in *50th Asilomar Conf. Signals, Systems and Computers*. IEEE, 2016, pp. 992–996.

[65] 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.

[66] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization – Part I: algorithm and convergence analysis," *IEEE Trans. Signal Processing*, vol. 64, no. 12, pp. 3118–3130, 2016.

[67] X. Hou, "Distributed solution for a class of multi-agent optimization problems," Ph.D. dissertation, Purdue University, May 2019.

[68] P. L. Combettes*, "Solving monotone inclusions via compositions of nonexpansive averaged operators," *Optimization*, vol. 53, no. 5-6, pp. 475–504, 2004.

[69] L. Elsner, I. Koltracht, and M. Neumann, "Convergence of sequential and asynchronous nonlinear paracontractions," *Numerische Mathematik*, vol. 62, no. 1, pp. 305–319, 1992.

[70] C. L. Byrne, *Applied iterative methods*. AK Peters Wellesley, 2008.

[71] L. Fang and P. J. Antsaklis, "Asynchronous consensus protocols using nonlinear paracontractions theory," *IEEE Transactions on Automatic Control*, vol. 53, no. 10, pp. 2351–2355, 2008.

[72] C. W. Wu, *Synchronization in complex networks of nonlinear dynamical systems*. World Scientific, 2007.

[73] L. Xiao, S. Boyd, and S. Lall, "Distributed average consensus with time-varying metropolis weights," *Automatica*, 2006.

[74] A. Cegielski, *Iterative methods for fixed point problems in Hilbert spaces*. Springer, 2012, vol. 2057.

[75] D. Davis and W. Yin, "A three-operator splitting scheme and its optimization applications," *Set-valued and variational analysis*, vol. 25, no. 4, pp. 829–858, 2017.

[76] D. W. Peaceman and H. H. Rachford, Jr, "The numerical solution of parabolic and elliptic differential equations," *Journal of the Society for industrial and Applied Mathematics*, vol. 3, no. 1, pp. 28–41, 1955.

[77] P.-L. Lions and B. Mercier, "Splitting algorithms for the sum of two nonlinear operators," *SIAM Journal on Numerical Analysis*, vol. 16, no. 6, pp. 964–979, 1979.

[78] J. Douglas and H. H. Rachford, "On the numerical solution of heat conduction problems in two and three space variables," *Transactions of the American mathematical Society*, vol. 82, no. 2, pp. 421–439, 1956.

[79] R. T. Rockafellar, "Monotone operators associated with saddle-functions and minimax problems," *Nonlinear functional analysis*, vol. 18, no. Part 1, pp. 397–407, 1970.

[80] M. Sion *et al.*, "On general minimax theorems." *Pacific Journal of mathematics*, vol. 8, no. 1, pp. 171–176, 1958.

[81] A. Ostrowski, "Über die determinanten mit überwiegender hauptdiagonale," *Commentarii Mathematici Helvetici*, vol. 10, no. 1, pp. 69–96, 1937.

[82] D. G. Feingold, R. S. Varga *et al.*, "Block diagonally dominant matrices and generalizations of the gerschgorin circle theorem." *Pacific Journal of Mathematics*, vol. 12, no. 4, pp. 1241–1250, 1962.

[83] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.* IEEE, 2005, pp. 63–70.

[84] J. Eckstein and D. P. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Math. Program.*, vol. 55, no. 1, pp. 293–318, 1992.

[85] Y. Xiao, X. Hou, and J. Hu, "Distributed solutions of convex-concave games on networks," in *2019 American Control Conference (ACC).* IEEE, 2019, pp. 1189–1194.

[86] J. Hu, Y. Xiao, and X. Hou, "Distributed solution of networked convex-concave games with coupling constraints," *IEEE Trans. Automatica Control*, submitted.