FAST TRACKING ADMM FOR DISTRIBUTED OPTIMIZATION AND CONVERGENCE UNDER TIME VARYING NETWORKS

by

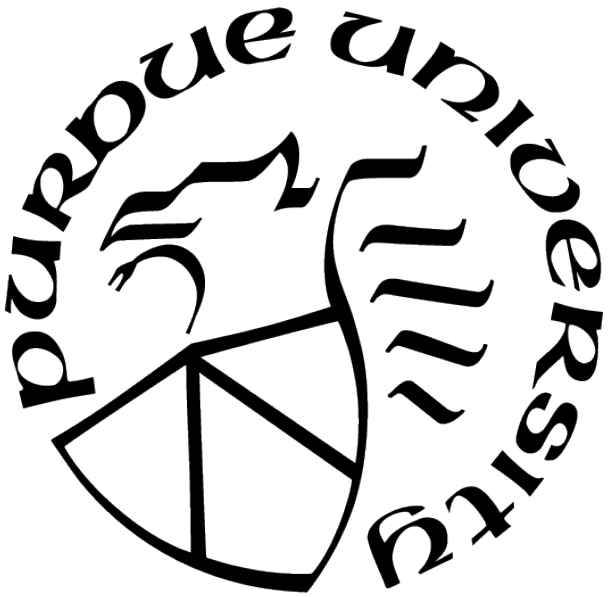
Shreyansh Shethia

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Aeronautics and Astronautics



School of Aeronautics and Astronautics

West Lafayette, Indiana

May 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Inseok Hwang, Chair

School of Aeronautics and Astronautics

Dr. Martin Corless

School of Aeronautics and Astronautics

Dr. Shaoshuai Mou

School of Aeronautics and Astronautics

Approved by:

Dr. Gregory A. Blaisdell

To my parents

ACKNOWLEDGMENTS

First and foremost, I am extremely grateful to my supervisor and graduate advisor, Prof. Inseok Hwang for his invaluable advice, continuous support, and patience during my thesis study. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life. Besides my major advisor, I would like to thank Prof. Martin Corless and Prof. Shaoshuai Mou, for their time and advice on my research. Additionally, I would also like to the appreciate the support provided by my PhD mentor, Akshita Gupta. She has spent countless hours advising me on my research and my thesis composition. I also like to thank my colleague Sesha Charla for helping me whenever I ask. Finally, I would like to express my gratitude to my family for their constant encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Due to the increase in the advances in wireless communication, there has been an increase in the use of multi-agents systems to complete any given task. In various applications, multi-agent systems are required to solve an underlying optimization problem to obtain the best possible solution within a feasible region. Solving such multi-agent optimization problems in a distributed framework preferable over centralized frameworks as the former ensures scalability, robustness, and security. Further distributed optimization problem becomes challenging when the decision variables of the individual agents are coupled.

In this thesis, a distributed optimization problem with coupled constraints is considered, where a network of agents aims to cooperatively minimize the sum of their local objective functions, subject to individual constraints. This problem setup is relevant to many practical applications like formation flying, sensor fusion, smart grids, etc. For practical scenarios, where agents can solve their local optimal solution efficiently and require fewer assumptions on objective functions, the Alternating Direction Method of Multipliers(ADMM)-based approaches are preferred over gradient-based approaches. For such a constraint coupled problem, several distributed ADMM algorithms are present that guarantee convergence to optimality but they do not discuss the complete analysis for the rate of convergence. Thus, the primary goal of this work is to improve upon the convergence rate of the existing state-of-the-art Tracking-ADMM (TADMM) algorithm to solve the above-distributed optimization problem. Moreover, the current analysis in literature does not discuss the convergence in the case of a time-varying communication network.

The first part of the thesis focuses on improving the convergence rate of the Tracking-ADMM algorithm to solve the above-distributed optimization problem more efficiently. To this end, an upper bound on the convergence rate of the TADMM algorithm is derived in terms of the weight matrix of the network. To achieve faster convergence, the optimal weight matrix is computed using a semi-definite programming (SDP) formulation. The improved convergence rate of this Fast-TADMM (F-TADMM) is demonstrated with a simple yet illustrative, coupled constraint optimization problem. Then, the applicability of F-TADMM

is demonstrated to the problem of distributed optimal control for trajectory generation of aircraft in formation flight.

In the second part of the thesis, the convergence analysis for TADMM is extended while considering a time-varying communication network. The modified algorithm is named as Time-Varying Tracking (TV-TADMM). The formal guarantees on asymptotic convergence are provided with the help of control system analysis of a dynamical system that uses Lyapunov-like theory. The convergence of this TV-TADMM is demonstrated on a simple yet illustrative, coupled constraint optimization problem with switching topology and is compared with the fixed topology setting.

1. INTRODUCTION

1.1 Background and Motivation

Large-scale networks, such as the Internet, handheld ad hoc networks, and cellular sensor networks, have gained considerable attention because of recent developments in wireless communication technology. Data-based networks, autonomous networks, unmanned aerial vehicle systems, social and economic networks, smart power networks, and disease networks are among the emerging network technology areas that have emerged because of these networks. To support multiple operations, such as resource allocation, teamwork, learning, and prediction, such systems often include decentralized in-network control and optimization techniques.

As a result, new models and tools for the architecture and performance analysis of massive, complex networked systems are urgently needed. The issues that arise in such networks are primarily caused by two factors: a lack of a central authority or coordinator (master node) and the network's underlying complex communication structure. A network system's lack of central control necessitates a modular architecture for network operations. In many applications, the decentralized framework is often preferred over a centralized one for mainly three reasons: (a) (scalability) The size of the network (the number of agents) and the resources needed to coordinate (i.e., communicate with) many agents make a centralized architecture impractical, (b) (robustness) a centralized network architecture is not robust to the failure of the central entity, and (c) (privacy) the privacy of agent information often cannot be preserved in a centralized system.

Over the last decade, there has been a lot of interest in distributed computing and decision-making problems, particularly consensus and flocking problems [1], [2], multiagent coverage problems [3], rendezvous problems [4], sensor localization in a multi-sensor network [5], and distributed management of multi-agent formations [6], to name a few. Often the underlying problem in all these applications is an optimization problem where the decision variables, objective function, constraints, and other problem parameters are distributed among the agents of the network.

More recently, significant efforts have been made to create distributed optimization algorithms that can be implemented in such networks without the need for a central coordinator and that can take advantage of the network access to reach a global success. Such distributed algorithms are characterized by the following properties: (a) They rely only on local information and observations (i.e., the agents can exchange some limited information with their one-hop neighbors only), (b) they are robust to the changes in the network topology (the topology is not necessarily static, as the communication links may not function perfectly), and (c) they are easily implementable in the sense that the local computations performed by the agents are not expensive.

In this thesis, we study a specific distributed optimization set-up that is relevant to several practical network control applications like formation flying and sensor network estimation. We consider that the agents which can communicate only with their neighbors in a network. Each agent only knows its local cost function and constraints, that depend on its individual decision variables. Moreover, the agents' decision variables are coupled by a linear constraint. Each agent knows only how its local decision variables affect the coupling constraint. The agents collectively aim at minimizing the sum of the local cost functions subject to both local and coupling constraints. The presence of this coupling element makes the problem solution challenging, especially in the considered distributed context in which no central authority (communicating with all the agents) is present. To explain the practical significance of constraint coupled problem consider the application of distributed optimization in smart grids [7]. Here, each sub-station need to optimally redistribute the power and can only communicate the power load to neighbouring substations. Since the total incoming power to substations is conserved, thus the individual decisions of the agents are required to satisfy a coupling constraint.

In the literature of distributed optimization, there have been many algorithms proposed that tackle the above-described problem. These are generally classified into two types. The first type is based on using the (sub)gradient of the local objective function to approach the optimal solution at each step, followed by using the neighboring nodes estimates to update the decision variable. The computation at each step can be very inexpensive and lead to distributed implementations [8]. The best known rate of convergence for sub-gradient based

distributed methods is $O\left(\frac{1}{\sqrt{k}}\right)$ [9], where k represents the iteration count. However, these methods require strong assumptions on objective functions and the fine tuning of step-size to ensure convergence [10].

The second type relies on dual methods where at each step, for a fixed value of the dual variable, the primal variables are solved to minimize some Lagrangian related function, then the dual variables are updated accordingly. This type of methods is preferred when each agent can solve the local optimization problem efficiently. One of the well known method of this kind is the Alternating Direction Method of Multipliers (ADMM), which decomposes the original problem into two sub-problems, sequentially solves them and updates the dual variables associated with a coupling constraint at each iteration. The best known rate of convergence for the classic ADMM algorithm is $O\left(\frac{1}{k}\right)$ [11], where k represents the iteration count. The ADMM-based approach is vastly preferred for practical implementation over competing methods because it can be easily formulated into a distributed problem and due to its robustness and extremely mild assumptions for convergence [12], [13], [10], [14]. Thus, in this work we focus on distributed ADMM-based approach to solve the problem of couple-constrained problem.

Although, empirically, ADMM has great performance, on a theoretical level as well as on practical level, its convergence rate properties are still not fully understood, especially in distributed settings. As shown in [15], [16], the convergence rate analysis of distributed ADMM is studied for the problem with unconstrained optimization. Their analysis further provides optimal penalty parameter for faster convergence. In [17], the convergence rate analysis considers the network weights and but again gives guarantees for an unconstrained optimization problem. In this work, we specifically consider a more general problem with local constraints. Further for such problem setting, in current literature there are no ADMM-based algorithms that guarantee convergence to optimality under the case of time varying communication network.

In this work, we are focused on improving the converge rate of distributed ADMM algorithms for couple constrained problems with local constraints with a specific emphasis on the network topology which is characterized by the edge weights. Particularly, we build upon the asymptotic convergence results of the existing Tracking ADMM (TADMM) algorithm [18] to

demonstrate faster convergence. The TADMM algorithm is preferred over other variants of ADMM because it can handle both local constraints and coupled constraints. Also as compared to other methods, TADMM does not require any parameter tuning and considers less assumptions on initialization of variables. Each agent in the TADMM algorithm maintains a local estimate of the constraint violations and collectively work to minimize the violation. Falsone et. al. [18] have shown that the TADMM algorithm converges asymptotically. But to improve the convergence speed, in this work, we analytically derive the convergence rate in terms of the communication weight matrix and compute the optimal weights to guarantee faster convergence. Lastly we build up on the analysis of TADMM to provide formal guarantees for convergence and optimality under the special case of time varying communication network.

1.2 Objectives and Contributions

In this thesis, we consider a system of multiple agents that cooperatively solve the problem of couple-constrained optimization problem. We aim to solve the problem in a distributed framework. We also consider the network weights. We specifically focus on the state-of-the-art algorithm, TADMM, for solving such problem. The main contributions of this work are: (1) computing and analyzing the convergence rate of the distributed TADMM algorithm in terms of the weight matrix of the communication graph; (2) computing the optimal weight matrix to minimize the convergence time; (3) demonstrating faster convergence of the proposed F-TADMM algorithm via an illustrative formation flying example, and (4) computing and analyzing the convergence and optimality of the distributed TADMM algorithm with respect to time-varying communication graph.

1.3 Organization

This thesis is organized as follows. In Chapter 2, we present the problem formulation and introduce the fixed topology for the distributed setting. In Chapter 3, we give a brief overview of TADMM algorithm and then derive and optimize its convergence rate to propose the F-TADMM algorithm. Here, we also demonstrate faster convergence rate of the proposed

F-TADMM algorithm with a sample problem. In Chapter 4, we demonstrate a real-world application of the F-TADMM algorithm via an illustrative example of distributed trajectory planning for formation flight. In Chapter 5, we introduce the distributed setting for time varying communication network. Then, we extend the convergence analysis of TADMM to propose Time Varying TADMM (TV-TADMM) with formal guarantees. We also demonstrate the convergence of the proposed TV-TADMM algorithm with a sample problem and present a comparison with the fixed topology setting. Finally, in Chapter 6, we provide a few concluding remarks along with our future plan to extend this work.

2. PROBLEM FORMULATION

We consider a multi-agent system comprising of N agents. These agents are indexed by the set $\mathcal{V} = \{1, 2, \dots N\}$. The goal of the agents is to solve a global optimization problem, with the decision variables coupled by a linear constraint. Further, the agents are only allowed to communicate with the neighboring agents. Here, we first define the global optimization problem and state the assumptions. Then we describe the underlying communication network among the agents.

2.1 Coupled-Constrained Optimization Problem

Let $f_{\mathrm{i}} : \mathbb{R}^n \to \mathbb{R}$ represent the local objective function and let $x_{\mathrm{i}} \in \mathbb{R}^n$ represent the local decision variable of agent i. Further let the linear coupling constraint between the agents represented by,

$$\sum_{\mathrm{i}\in\mathcal{V}} A_{\mathrm{i}} x_{\mathrm{i}} = b \tag{2.1}$$

where $A_{\mathrm{i}} \in \mathbb{R}^{m\times n}$, be the coupling coefficients available only to the agent i, and $b \in \mathbb{R}^m$, is the constant of the linear coupling constraint. It is noted that, the agent i only knows the partial information of this coupling vector given by b_{i}, such that $\sum_{\mathrm{i}\in\mathcal{V}} b_{\mathrm{i}} = b$. The problem also considers that the local decision variable x_{i} is constrained to the local set $\mathcal{X}_{\mathrm{i}} \subseteq \mathbb{R}^n$.

Then, the optimization problem is that all the agents $\mathrm{i} \in \mathcal{V}$ must reach the optimal values of their decision variables, such that the overall system minimizes the sum of their individual objective functions while satisfying local and the linearly coupled constraint. Mathematically, the global optimization problem can be stated as,

$$\begin{aligned} &\min_{x_1,\dots,x_N} \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}\left(x_{\mathrm{i}}\right) \\ &\text{subj. to: } \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} x_{\mathrm{i}} = b, \quad x_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \{\mathrm{i} = 1, \dots, N\}, \end{aligned} \tag{2.2}$$

Assumption 2.1.1 (Local Objectives)**.** *The local objective function, f_{i}, is convex, and the local constraint set, $\mathcal{X}_{\mathrm{i}}$, is convex and compact, for all* $\mathrm{i} \in \{1, 2, \dots, N\}$.

Problem (2.2) is rewritten in it's dual form by forming the Lagrangian function. Let $\mathbf{x}_k = [x_1^T, x_2^T, \dots x_N^T]^T$ and $\lambda \in \mathbb{R}^p$ denote the Lagrangian multipliers. Then,

$$L(\mathbf{x}, \lambda) = \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(x_{\mathrm{i}}) + \lambda^T (\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} x_{\mathrm{i}} - b). \tag{2.3}$$

The formulation for the dual problem follows as,

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^p} \min_{\boldsymbol{x} \in \tilde{\mathbf{X}}} L(\boldsymbol{x}, \lambda) &= \max_{\lambda \in \mathbb{R}^p} \sum_{\mathrm{i}=1}^{N} \varphi_{\mathrm{i}}(\lambda) \\ \text{where : } \varphi_{\mathrm{i}}(\lambda) &= \min_{x_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}} f_{\mathrm{i}}\left(x_{\mathrm{i}}\right) + \lambda^{\top}\left(A_{\mathrm{i}} x_{\mathrm{i}} - b_{\mathrm{i}}\right) \end{aligned} \tag{2.4}$$

where $\tilde{\mathbf{X}} = \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \mathcal{X}_N$ and $\sum_{\mathrm{i}=1}^{N} b_{\mathrm{i}} = b$.

Assumption 2.1.2 (Existence of Solution)**.** *The solution to the primal problem 2.3,* $\mathbf{x}^*$*, exists and the solution to the dual problem 2.4,* λ^**, also exists within the domain.*

The above assumption is common in the literature, [15], [17], [18], as it ensures feasibility of the given problem.

2.2 Distributed Computation Framework

Here we describe the network topology and state the necessary assumptions. The underlying, undirected communication graph of the network is denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ is the set of nodes, representing the agents, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, representing the communication links. An edge $(\mathrm{i}, \mathrm{j}) \in \mathcal{E}$ if agent i receives information from agent j and vice versa. It is assumed that the communication graph is time-invariant. The neighborhood of agent i in $\mathcal{G}$ is denoted by $\mathcal{N}_{\mathrm{i}} = \{\mathrm{j} \in \mathcal{V} \mid (\mathrm{i}, \mathrm{j}) \in \mathcal{E}\}$, with $(\mathrm{i}, \mathrm{i}) \in \mathcal{E}, \forall \mathrm{i} \in \{1, \dots, N\}$. Each edge $(\mathrm{i}, \mathrm{j}) \in \mathcal{E}$ has an associated weight w_{ij}, which measures how much agent i values the information received from agent j. The weight matrix of the entire network is denoted by $W \in \mathbb{R}^{N \times N}$, where w_{ij} represents the $(\mathrm{i}, \mathrm{j})^{th}$ element of W, such that,

$$w_{\mathrm{ij}} = \begin{cases} 0, & (\mathrm{i}, \mathrm{j}) \notin \mathcal{E}, \\ \geq 0, & \text{otherwise.} \end{cases} \tag{2.5}$$

Additionally, we consider the following assumption on these network weights,

Assumption 2.2.1 (Network Properties)**.** *The communication graph $\mathcal{G}$ is undirected and connected. The associated weight matrix W has the following properties:*

1. *W is balanced, i.e., $w_{ij} = w_{ji}, \forall (i,j) \in \mathcal{E}$,*
2. *W is a doubly stochastic matrix, i.e.,*

$$W\hat{\mathbf{1}}_N = \hat{\mathbf{1}}_N, \quad W^T\hat{\mathbf{1}}_N = \hat{\mathbf{1}}_N,$$

   where $\hat{\mathbf{1}}_N^T \coloneqq [1, 1, \ldots, 1]_{1\times N}$

3. *W is a positive semidefinite, ie, $W \succeq 0$*

From Assumption 2.2.1, the weight matrix W belongs to the following set of matrices

$$\Gamma = \{W \in \mathbb{R}^{N\times N} | w_{i,j} = w_{j,i}, W\hat{\mathbf{1}} = \hat{\mathbf{1}}, W^T\hat{\mathbf{1}} = \hat{\mathbf{1}}, W \succeq 0\}$$

Based on many applications, it is assumed that this underlying network topology is provided. Such practical scenarios include, a network of agents in formation flight [19], large scale networks for distributed machine learning [12], power grid networks [20] and sensor networks [5]. However many of these applications consider standard adjacency weight matrix. It is noted that if $w_{i,j} = 1/d_{i,i}, \quad \forall\{(i,j) \in \mathcal{E}\} \cap \{i \neq j\}$ and $w_{i,i} = 1$, where deg_i is degree of agent i then W coorespond to the standard adjacency matrix of the given network topology. But here we consider a more general case where these weights can vary between $(0, 1)$. And, as shown in this thesis, based on the network topology, we have more flexibility to maximize the convergence speed of our distributed algorithm through choosing these weights.

Moreover it is noted that the problem setting can be applied to a wide range of practical problems, including formation flight, and it can handle the commonly used linear and quadratic objective functions. In Chapter 4, we describe one such approach on how the formation flight problem can be modeled as a coupled constraint convex optimization problem as described in problem (2.2).

3. ALGORITHM DEVELOPMENT FOR FAST-TRACKING ADMM

In this chapter, we develop the Fast Tracking ADMM (F-TADMM) algorithm for iteratively solving problem (2.2). Here we begin our approach by introducing Tracking ADMM (TADMM) algorithm which is the state of the art algorithm to handle distributed problems with coupled constraints. TADMM [18] is a fully distributed optimization algorithm used to solve the coupled-constraint problem (2.2), over a network by means of an ADMM-based approach. On comparison, it is observed that the TADMM algorithm requires no parameter tuning, unlike the distributed ADMM algorithm in [15], and it requires less restrictions on the initialization of decision variables when compared to the algorithms in [12], [17]. In [18], the authors also proved asymptotic convergence of the TADMM algorithm to the optimal solution. Therefore, given the above advantages, in this paper, we focus on improving the convergence rate of the TADMM algorithm for solving problem (2.2).

Essentially, the TADMM algorithm 11, creates local copies of the dual variable, λ_{i}, as well as local copies of the coupling constraint violation, defined as $d_{\mathrm{i}} = A_{\mathrm{i}}x_{\mathrm{i}} - b_{\mathrm{i}}$. At each step k, after receiving the constraint violation and dual variable (step 2) from neighbours, each agent computes the weighted average of d_{i}^k and λ_{i}^k, represented as Δd_{j}^k and $\Delta\lambda_{\mathrm{j}}^k$, respectively (steps 3, 4). Using these averages, each agent performs the local optimization to obtain x_{i}^{k+1} (step 5). And finally, it enforces the consensus-based approach (steps 6, 7) to ensure the total coupling constraint is satisfied.

Optimality and Convergence: For the primal and dual problems, (2.2) and (2.4), respectively, under Assumptions 2.1.2,and 2.2.1, it has been proved that the TADMM algorithm achieves asymptotic convergence (see Theorem 1, [18]) to the pair of optimal solutions $(\mathbf{x}^*, \lambda^*)$ (see [18], Theorem 2).

However, in real world problems, a stronger analysis of the convergence rate is required to ensure that a solution will reached in practical time. Therefore, in the next section, we extend the analysis of the above algorithm to provide some guarantees on the convergence rate.

Algorithm 1 TADMM

procedure INITIALIZATION
    $x_{\mathrm{i}}^0 \in \mathcal{X}_{\mathrm{i}}$, $\lambda_{\mathrm{i}}^0 \in \mathcal{X}_{\mathrm{i}}$, $d_{\mathrm{i}}^0 = A_{\mathrm{i}} x_{\mathrm{i}}^0 - b_{\mathrm{i}}$
end procedure
procedure REPEAT TILL CONVERGENCE
    Communicate to neighbors: $(d_{\mathrm{i}}^k, \lambda_{\mathrm{i}}^k)$
    Compute Deviation: $\Delta d_{\mathrm{i}}^k = \sum_{\mathrm{j} \in \mathcal{N}_{\mathrm{i}}} w_{\mathrm{i,j}} d_{\mathrm{j}}^k$
    Compute Dual Variable: $\Delta \lambda_{\mathrm{i}}^k = \sum_{\mathrm{j} \in \mathcal{N}_{\mathrm{i}}} w_{\mathrm{i,j}} \lambda_{\mathrm{j}}^k$
    Local Optimization: $x_{\mathrm{i}}^{k+1} = \arg\min_{x \in \mathcal{X}_{\mathrm{i}}} \{ f_{\mathrm{i}}(x) + (\Delta \lambda_{\mathrm{i}}^k)^T A_{\mathrm{i}} x + \frac{\rho}{2} ||A_{\mathrm{i}} x - A_{\mathrm{i}} x_{\mathrm{i}}^k + \Delta d_{\mathrm{i}}^k||^2 \}$
    Update Deviation: $d_{\mathrm{i}}^{k+1} = \Delta d_{\mathrm{i}}^k + A_{\mathrm{i}}(x_{\mathrm{i}}^{k+1} - x_{\mathrm{i}}^k)$
    Update Dual Variable: $\lambda_{\mathrm{i}}^{k+1} = \lambda_{\mathrm{i}}^k + \rho d_{\mathrm{i}}^{k+1}$
end procedure

3.1 Convergence Rate Analysis for TADMM

While TADMM does discuss the convergence analysis to the optimal solution, it does not share any information on the convergence rate. In this section, we analytically derive the convergence rate of the TADMM algorithm and show that it depends on the weights of the communication network. Further, we present the network weight optimization for better convergence using SDP and finally present our Fast-Tracking ADMM (F-TADMM) algorithm.

To analyze the convergence rate, we use a compact representation of the update rules of the TADMM algorithm, by stacking the vectors from each node. Let

$$\mathbf{x}_k \coloneqq [x_{1,k}^T, x_{2,k}^T, \dots x_{N,k}^T]^T, \quad \mathbf{d}_k \coloneqq [d_{1,k}^T, d_{2,k}^T, \dots d_{N,k}^T]^T, \quad \lambda_k \coloneqq [\lambda_{1,k}^T \lambda_{2,k}^T, \dots \lambda_{N,k}^T]^T \tag{3.1}$$

$$\boldsymbol{\Delta}\mathbf{d}_k \coloneqq [\Delta d_{1,k}^T, \Delta d_{2,k}^T, \dots \Delta d_{N,k}^T]^T. \quad \boldsymbol{\Delta}\lambda_k \coloneqq [\Delta\lambda_{1,k}^T, \Delta\lambda_{2,k}^T, \dots \Delta\lambda_{N,k}^T]^T \tag{3.2}$$

Therefore, the update rules in Algorithm 1, [18], can be written compactly as:

$$\begin{aligned}
&(1)\, \mathbf{x}_{k+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \{\boldsymbol{f}(\mathbf{x}) + \boldsymbol{\Delta}\mathbf{d}_k^T \mathbf{A}_d \mathbf{x} + \frac{\rho}{2} ||\mathbf{A}_d\mathbf{x} - \mathbf{A}_d\mathbf{x}_k + \boldsymbol{\Delta}\lambda_k||^2\} \\
&(2)\, \mathbf{d}_{k+1} = \mathbf{W}\mathbf{d}_k + \mathbf{A}_d\mathbf{x}_{k+1} - \mathbf{A}_d\mathbf{x}_k \\
&(3)\, \boldsymbol{\lambda}_{k+1} = \mathbf{W}\boldsymbol{\lambda}_k + \rho\mathbf{d}_{k+1}
\end{aligned} \tag{3.3}$$

where $\boldsymbol{f}(x) = \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(x_{\mathrm{i}})$ is the global objective function. The block diagonal matrix $\mathbf{A}_d = \mathrm{blkdiag}(A_1, A_2, \dots, A_N)$ is the compact matrix representation of the coupled constraints, while $\mathbf{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_N$ represents the concatenation of the local constraint sets. The global weight matrix $\mathbf{W} \coloneqq W \otimes I_p$, where $\otimes$ represents the Kronecker product. Therefore, the update rules to compute the constraint violations and changes in dual variables are now given as $\boldsymbol{\Delta}\mathbf{d}_k = \mathbf{W}\mathbf{d}_k$ and $\boldsymbol{\Delta}\boldsymbol{\lambda}_k = \mathbf{W}\boldsymbol{\lambda}_k$, respectively.

Now, let the network average of the consensus variables be given as: $\bar{d}_k \coloneqq \frac{1}{N}\sum_{\mathrm{i}=1}^{N} d_{\mathrm{i}}^k$ and $\bar{\lambda}_k \coloneqq \frac{1}{N}\sum_{\mathrm{i}=1}^{N} \lambda_{\mathrm{i}}^k$.

From Assumption 2.2.1, it follows that $\forall k \geq 0$,

$$\bar{d}_k = \frac{1}{N}\left(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} x_{\mathrm{i}}^k - b_{\mathrm{i}}\right), \quad \bar{\lambda}_{k+1} = \bar{\lambda}_k + \rho \bar{d}_{k+1}. \tag{3.4}$$

The introduction of variables $\bar{d}_k$ and $\bar{\lambda}_k$ is useful to analyze the behavior of d_{i}^k and λ_{i}^k. Specifically, to analyze the convergence rate, we first model the error dynamics of the TADMM algorithm by studying the distance between the agents' local estimates d_{i}^k and λ_{i}^k and the respective network averages $\bar{d}_k$ and $\bar{\lambda}_k$.

Let $\bar{\mathbf{d}}_k \coloneqq \hat{\mathbf{1}}_N \otimes \bar{d}_k$ and $\bar{\boldsymbol{\lambda}}_k \coloneqq \hat{\mathbf{1}}_N \otimes \bar{\lambda}_k$. Then, using the fact that W is doubly stochastic and the properties of Kronecker product,

$$\mathbf{W}\bar{\mathbf{d}}_k = (W \otimes I_p)(\hat{\mathbf{1}}_N \otimes \bar{d}_k) = \bar{\mathbf{d}}_k \tag{3.5}$$

Defining the error vectors as:

$$\mathbf{e}_k^d = \mathbf{d}_k - \bar{\mathbf{d}}_k, \quad \mathbf{e}_k^\lambda = \boldsymbol{\lambda}_k - \bar{\boldsymbol{\lambda}}_k \tag{3.6}$$

For clarity, we define an additional term $\mathbf{z}_k \coloneqq \mathbf{A}_d \mathbf{x}_k - \bar{\mathbf{d}}_k$. Now, using the update (2) in eq. 5.2, the error dynamics for $\mathbf{e}^d$ is derived as:

$$\begin{aligned}
\mathbf{e}_{k+1}^d &= \boldsymbol{d}_{k+1} - \overline{\boldsymbol{d}}_{k+1} \\
&= \mathbf{W}\boldsymbol{d}_k + \mathbf{A}_d \boldsymbol{x}_{k+1} - \mathbf{A}_d \boldsymbol{x}_k - \overline{\boldsymbol{d}}_{k+1} \pm \overline{\boldsymbol{d}}_k \\
&\overset{(a)}{=} \mathbf{W}\boldsymbol{d}_k - \overline{\boldsymbol{d}}_k + [\boldsymbol{z}_{k+1} - \boldsymbol{z}_k] \\
&\overset{(b)}{=} \mathbf{W}\mathbf{e}_k^d + [\boldsymbol{z}_{k+1} - \boldsymbol{z}_k]
\end{aligned} \tag{3.7}$$

Where the equality (a) is due to the the definition of $\boldsymbol{z}_k$ and in (b) we use equation 3.5. With similar arguments, using the update (3) in eq. 5.2, the error dynamics for $\mathbf{e}^{\lambda}$ is derived as:

$$\begin{aligned}\mathbf{e}^{\boldsymbol{\lambda}}_{k+1} &= \boldsymbol{\lambda}_{k+1} - \overline{\boldsymbol{\lambda}}_{k+1} \\ &= \mathbf{W}\boldsymbol{\lambda}_k + c\boldsymbol{d}_{k+1} - \left(\overline{\boldsymbol{\lambda}}_k + c\overline{\boldsymbol{d}}_{k+1}\right) \\ &= \mathbf{W}\mathbf{e}^{\boldsymbol{\lambda}}_k + \rho\mathbf{e}^{\boldsymbol{d}}_{k+1} \\ &= \mathbf{W}\mathbf{e}^{\boldsymbol{\lambda}}_k + \rho\mathbf{e}^{\boldsymbol{d}}_{k+1}\end{aligned} \tag{3.8}$$

Now let the limiting matrix $\mathcal{O} = \frac{1}{N}\hat{\mathbf{1}}_N\hat{\mathbf{1}}_N^T$. And as discussed in [21], based on network assumptions 2.2.1, the network weight matrix converges to this limiting matrix,

$$\lim_{k\to\infty} W^k = \mathcal{O} \tag{3.9}$$

Further the following properties on the error vecotrs are introduced as

$$\begin{aligned}(\mathcal{O}\otimes I_p)\mathbf{e}^d_k &= (\frac{1}{N}\hat{\mathbf{1}}_N\hat{\mathbf{1}}_N^T \otimes I_p)\left[\mathbf{d}_k - \bar{\mathbf{d}}_k\right] \\ &= \hat{\mathbf{1}}_N \otimes \frac{1}{N}\sum_{\mathrm{i}=1}^{N}[d^k_{\mathrm{i}} - \bar{d}^k] = 0.\end{aligned} \tag{3.10}$$

Similarly,

$$\begin{aligned}(\mathcal{O}\otimes I_p)\mathbf{e}^{\lambda}_k &= (\frac{1}{N}\hat{\mathbf{1}}_N\hat{\mathbf{1}}_N^T \otimes I_p)\left[\boldsymbol{\lambda}_k - \bar{\boldsymbol{\lambda}}_k\right] \\ &= \hat{\mathbf{1}}_N \otimes \frac{1}{N}\sum_{\mathrm{i}=1}^{N}[\lambda^k_{\mathrm{i}} - \bar{\lambda}^k] = 0\end{aligned} \tag{3.11}$$

Defining $\tilde{\mathbf{W}} = (W - \mathcal{O}) \otimes I_p$ and using the above properties, the overall error dynamics is given as:

$$\mathrm{e}_{k+1} = \mathbf{A}\mathrm{e}_k + \mathbf{B}\left(\boldsymbol{z}_{k+1} - \boldsymbol{z}_k\right) \tag{3.12}$$

where

$$\mathrm{e}_k = \begin{bmatrix} \mathrm{e}^{\boldsymbol{\lambda}}_k \\ \rho\mathrm{e}^{\boldsymbol{d}}_k \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} \tilde{\mathbf{W}} & \tilde{\mathbf{W}} \\ 0 & \tilde{\mathbf{W}} \end{bmatrix}, \quad \mathbf{B} = \rho\begin{bmatrix} I \\ I \end{bmatrix}$$

From Assumption 2.2.1, $\left\|W - \mathcal{O}\right\| < 1$. Therefore, all the eigenvalues of $\tilde{\mathbf{W}}$ lie within the unit circle. This implies that the dynamical system describing the evolution of the consensus

errors in (3.12) is asymptotically stable. Also, it is proven [18] that the sequences, $\{\mathrm{e}_k\}_{k\geq 0}$ and $\{\boldsymbol{d}_k\}_{k\geq 0}$, are bounded.

Theorem 3.1.1. *The convergence rate factor, γ, of the TADMM algorithm is bounded by*

$$\gamma < \left\|\mathbf{A}\right\| + \sup \frac{\left\|\mathbf{B}\left(\boldsymbol{z}_{k+1} - \boldsymbol{z}_k\right)\right\|}{\left\|\mathbf{e}_k\right\|}. \tag{3.13}$$

Proof: Using the standard definition of per-step convergence factor, defined in [21], we have

$$\gamma = \sup_{\mathbf{e}_k \neq \bar{\mathbf{e}}} \frac{\left\|\mathbf{e}_{k+1} - \bar{\mathbf{e}}\right\|}{\left\|\mathbf{e}_k - \bar{\mathbf{e}}\right\|}. \tag{3.14}$$

Also, from Theorem 1 in [18], since the consensus updates converge asymptotically as $\lim_{k\to\infty} \left\|\mathrm{e}_k\right\| = 0$ and $\lim_{k\to\infty} \left\|\bar{\boldsymbol{d}}_k\right\| = 0$, it implies the error itself converges to $\bar{\mathbf{e}} = 0$. Hence,

$$\gamma = \sup \frac{\left\|\mathbf{e}_{k+1}\right\|}{\left\|\mathbf{e}_k\right\|} \tag{3.15}$$

$$\overset{(a)}{=} \sup \frac{\left\|\mathbf{A}\mathrm{e}_k + \mathbf{B}\left(\boldsymbol{z}_{k+1} - \boldsymbol{z}_k\right)\right\|}{\left\|\mathbf{e}_k\right\|} \tag{3.16}$$

$$\overset{(b)}{<} \sup \frac{\left\|\mathbf{A}\mathrm{e}_k\right\|}{\left\|\mathrm{e}_k\right\|} + \frac{\left\|\mathbf{B}\left(\boldsymbol{z}_{k+1} - \boldsymbol{z}_k\right)\right\|}{\left\|\mathbf{e}_k\right\|} \tag{3.17}$$

$$\overset{(c)}{<} \left\|\mathbf{A}\right\| + \sup \frac{\left\|\mathbf{B}\left(\boldsymbol{z}_{k+1} - \boldsymbol{z}_k\right)\right\|}{\left\|\mathbf{e}_k\right\|}. \quad \square \tag{3.18}$$

where in (a), the error dynamics equation 3.12 is used and in (b) and (c), following matrix norm properties are used.

$$\left\|Q + R\right\| \leq \left\|Q\right\| + \left\|R\right\|$$

$$\left\|Q\right\| := \sup_{\|x\|\neq 0} \frac{\left\|Qx\right\|}{\left\|x\right\|}$$

where Q and R, are matrices of arbitrary size.

3.2 Optimal Weights for Fast Convergence

Now, since the convergence time is given by, [21],

$$\tau_{step} = \frac{1}{\log 1/\gamma},$$

minimizing the convergence time corresponds to minimizing the convergence rate factor γ. From (3.18), it is also noted that the optimal value of γ is achieved by optimizing the network weight matrix W. This optimization problem is given as:

$$\begin{aligned} \min_{W} \quad & \gamma(W) \\ \text{subject to: } & W \in \Gamma. \end{aligned} \tag{3.19}$$

Now $\min_W \gamma(W) \implies \min_W \left\|\mathbf{A}\right\|$, because only the first term depend on the network weights. Further, using the definition of $\mathbf{A}$,

$$\left\|\mathbf{A}\right\| = \left\| \begin{bmatrix} \tilde{\mathbf{W}} & \tilde{\mathbf{W}} \\ 0 & \tilde{\mathbf{W}} \end{bmatrix} \right\|.$$

It is also noted that the eigenvalues of $\mathbf{A}$ are the same as the eigenvalues of $\tilde{\mathbf{W}}$ because it is a block diagonal matrix. Since the matrix norm is related to the eigenvalues, we have $\left\|\mathbf{A}\right\| \propto \left\|\mathbf{W}\right\| = \left\|W - \mathcal{O}\right\|\left\|I_p\right\|$. Hence, the objective function in (3.19) becomes $\min_W \gamma(W) \implies \min_W \left\|W - \mathcal{O}\right\|$.

Therefore, problem (3.19) is now modeled as the spectral norm minimization problem, which can be expressed as an semi-definite programming (SDP) [21], by introducing a scalar variable s to bound the norm $\left\|W - \mathcal{O}\right\|$, and expressing the norm bound constraint as a linear matrix inequality (LMI). Thus, we have

$$\begin{aligned} \min \quad & s \\ \text{subject to: } & -sI \preceq W - \frac{1}{N}\hat{\mathbf{1}}_N\hat{\mathbf{1}}_N \preceq sI, \\ & W \in \Gamma, \end{aligned} \tag{3.20}$$

where $s \in \mathbb{R}$, $W \in \mathbb{R}^{N\times N}$. Also let the optimal weights are given by $[W^*]_{\mathrm{i,j}} = w^*_{\mathrm{i,j}}$. Note that the process of weight optimization is offline. This approach is practical in formation flying scenarios where we are given that certain agents are paired by communication constraints. Therefore, we are tasked with finding the best possible network weights to ensure faster convergence. Finally, the F-TADMM algorithm is given in Algorithm 2.

Algorithm 2 F-TADMM

procedure INITIALIZATION
  $x^0_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}$, $\lambda^0_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}$, $d^0_{\mathrm{i}} = A_{\mathrm{i}}x^0_{\mathrm{i}} - b_{\mathrm{i}}$
  find W^* using equation (3.20)
end procedure
procedure REPEAT TILL CONVERGENCE
  Communicate to neighbors: $(d^k_{\mathrm{i}}, \lambda^k_{\mathrm{i}})$
  Compute Deviation: $\Delta d^k_{\mathrm{i}} = \sum_{\mathrm{j}\in\mathcal{N}_{\mathrm{i}}} w^*_{\mathrm{i,j}} d^k_{\mathrm{j}}$
  Compute Dual Variable: $\Delta\lambda^k_{\mathrm{i}} = \sum_{\mathrm{j}\in\mathcal{N}_{\mathrm{i}}} w^*_{\mathrm{i,j}} \lambda^k_{\mathrm{j}}$
  Local Optimization: $x^{k+1}_{\mathrm{i}} = \arg\min_{x\in\mathcal{X}_{\mathrm{i}}}\{f_{\mathrm{i}}(x) + (\Delta\lambda^k_{\mathrm{i}})^T A_{\mathrm{i}}x + \frac{\rho}{2}||A_{\mathrm{i}}x - A_{\mathrm{i}}x^k_{\mathrm{i}} + \Delta d^k_{\mathrm{i}}||^2\}$
  Update Deviation: $d^{k+1}_{\mathrm{i}} = \Delta d^k_{\mathrm{i}} + A_{\mathrm{i}}(x^{k+1}_{\mathrm{i}} - x^k_{\mathrm{i}})$
  Update Dual Variable: $\lambda^{k+1}_{\mathrm{i}} = \lambda^k_{\mathrm{i}} + \rho d^{k+1}_{\mathrm{i}}$
end procedure

3.3 Comparison with TADMM

The performance of the proposed F-TADMM algorithm is compared with that of the standard TADMM algorithm on a sample problem with the same form as that of (2.2). More specifically, a random communication graph is generated for $N = 20$ agents, as shown in Fig. 3.1(a). Let M represents the edge-node incidence matrix of the undirected communication graph. Each row of M corresponds to the edge $\mathcal{E}(\mathrm{i},\mathrm{j})$ in the graph without repetition and each column an agent, such that

$$[M]_k^{\mathcal{E}(\mathrm{i},\mathrm{j})} = \begin{cases} 1, & \text{if } k = \mathrm{i} \\ -1, & \text{if } k = \mathrm{j} \\ 0, & \text{otherwise.} \end{cases}$$

The consensus constraint $\{x_1 = x_2 = \ldots = x_N\}$ can be written as: $\{\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} x_{\mathrm{i}} = \mathbf{0}\}$, where the coupling matrix A_{i} corresponds to the $\mathrm{i}-th$ column of incidence matrix M as following:

$$A_{\mathrm{i}} = M_{\mathrm{i}}^T \otimes I_{n\times n}.$$

For the simulation problem, $x_{\mathrm{i}} \in \mathbb{R}^2, \forall \mathrm{i} \in \mathcal{V}$ and are initialized randomly. The local objective functions are designed as quadratic functions, i.e., $f_{\mathrm{i}}(x) = x^T P_{\mathrm{i}} x + q_{\mathrm{i}}^T x$, where P_{i} are random positive definite matrices. The local constraints are given as $x_{min}^{\mathrm{i}} \leq x_{\mathrm{i}} \leq x_{max}^{\mathrm{i}}$. One such random instance of the problem setting is visualized in Fig. 3.1(b), where the red points represent the initial positions of the agents. The local constraints are represented using blue rectangular boxes and the intersection of these constraints are represented by the green rectangle.

Both the standard TADMM and F-TADMM algorithms are implemented to solve the problem and both the algorithms converge at a consensus. According to [18], TADMM network weights are randomly generated such that they satisfy $W \in \Gamma$. However, Fig. 3.2 shows that the F-TADMM algorithm converges much faster than the standard TADMM algorithm to the same threshold. we also compute $\left\|W - \mathcal{O}\right\|$ both algorithms. For optimal weights (OW), it is 0.6440 whereas for standard randomized weights (SW) used by TADMM, it is 0.7787. This matches the convergence behavior of OW and SW shown in Fig 3.2 and justifies that the minimization of $\left\|W - \mathcal{O}\right\|$ has indeed improved the convergence time. Note that since the optimal weights are computed by solving an offline optimization problem, the number of computations in each iteration of both the algorithms are the same. Hence, comparing the convergence rate in terms of number of iterations is justified in this paper.

To test the scalability of the proposed F-TADMM algorithm, we conduct an empirical study of the number of iterations required for convergence with increasing number of nodes in Table 3.1. To reduce the effect of changing graph topology, 10 random simulations were performed for different values of N, with both the standard TADMM and proposed F-TADMM algorithms. The mean and standard deviation of the number of iterations are documented in Table 3.1. It is observed that for all the different values of N, the proposed F-TADMM algorithm requires fewer iterations than the standard TADMM algorithm to converge. Ad-

ditionally, the standard deviation of the number of iterations for standard TADMM is higher than that for F-TADMM. This implies that the performance of the proposed algorithm is less susceptible to the topology of the underlying communication graph network than the TADMM algorithm. These simulation results shows an improvement in convergence rate by the proposed F-TADMM algorithm.

Further, we perform another set of simulations with fixed topology and network weights but different F-TADMM penalty factor $\rho = \{0.01, 0.1, 0.5, 1, 5\}$. The results are shown in 3.5. It clear from simulations regardless of ρ, the F-TADMM algorithm converges to the optimal value ie consensus. Also it is observed that the choosing an optimal ρ also changes the convergence rate of the algorithm. This study of convergence rate with respect to ρ is left for future work.

In the next chapter, we demonstrate how the F-TADMM algorithm can be applied to solve the practical problem of formation flight.

Table 3.1. Scalability analysis of TADMM algorithm with standard weights (SW) and optimized weights (OW).

N (# of agents)		**5**	**10**	**20**	**30**	**40**	**50**
SW (Standard Weights)	Mean	226.30	489.70	602.20	537.80	459.00	445.70
	Variance	31.78	112.67	199.69	253.04	87.84	58.23
OW (Optimal Weights)	Mean	154.90	398.60	371.00	258.60	265.80	308.00
	Variance	27.27	140.61	99.38	53.75	12.98	22.86

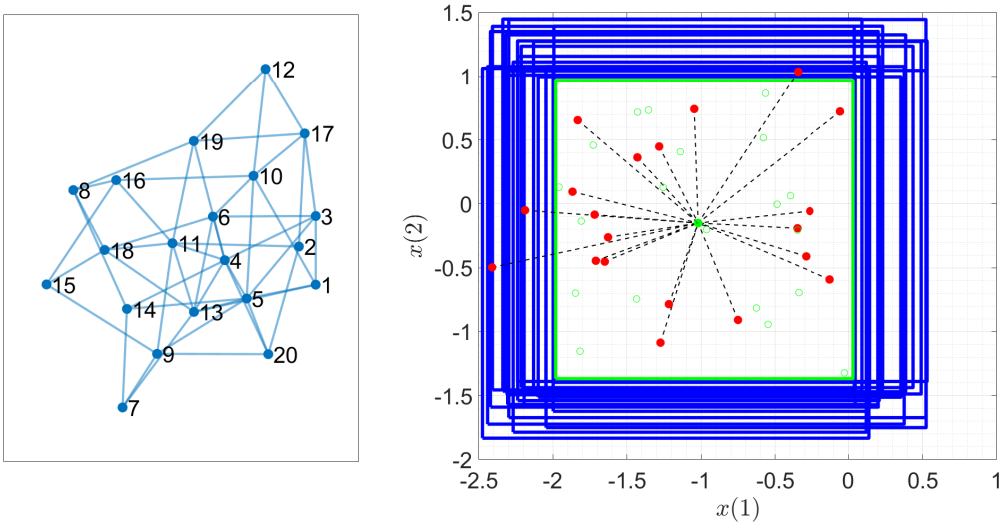


Figure 3.1. (a) The random graph network for $N = 20$ agents; (b) Consensus is achieved in the feasible region by all the agents starting from random initial points (in red)

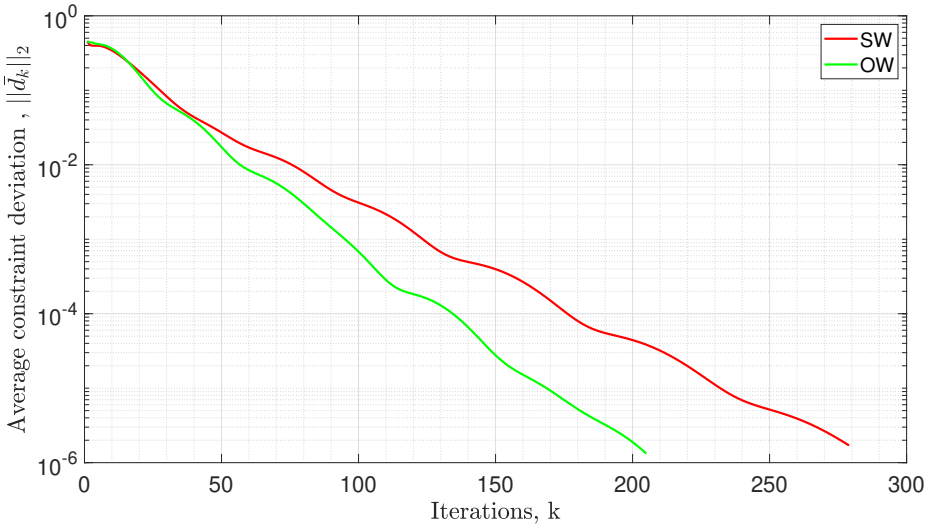


Figure 3.2. Comparison of convergence trend between the proposed F-TADMM and the baseline algorithm with $N = 20$.

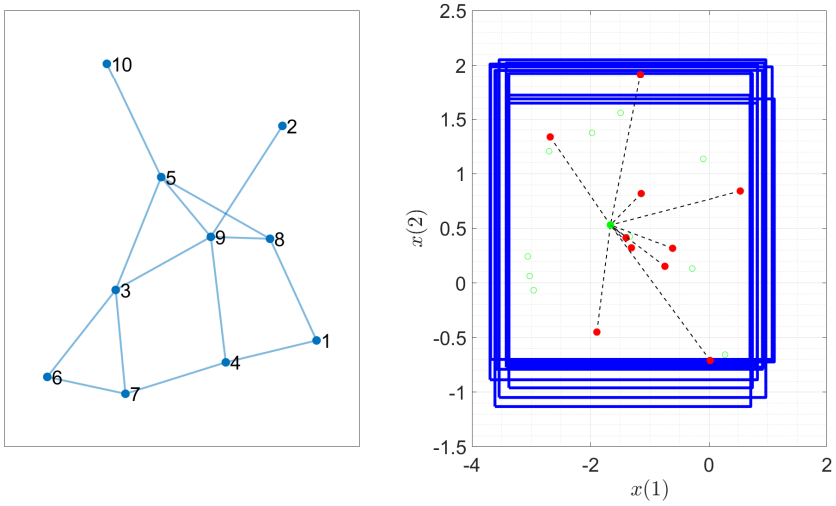


Figure 3.3. (a) The random graph network for $N = 10$ agents; (b) Consensus is achieved in the feasible region by all the agents starting from random initial points (in red)

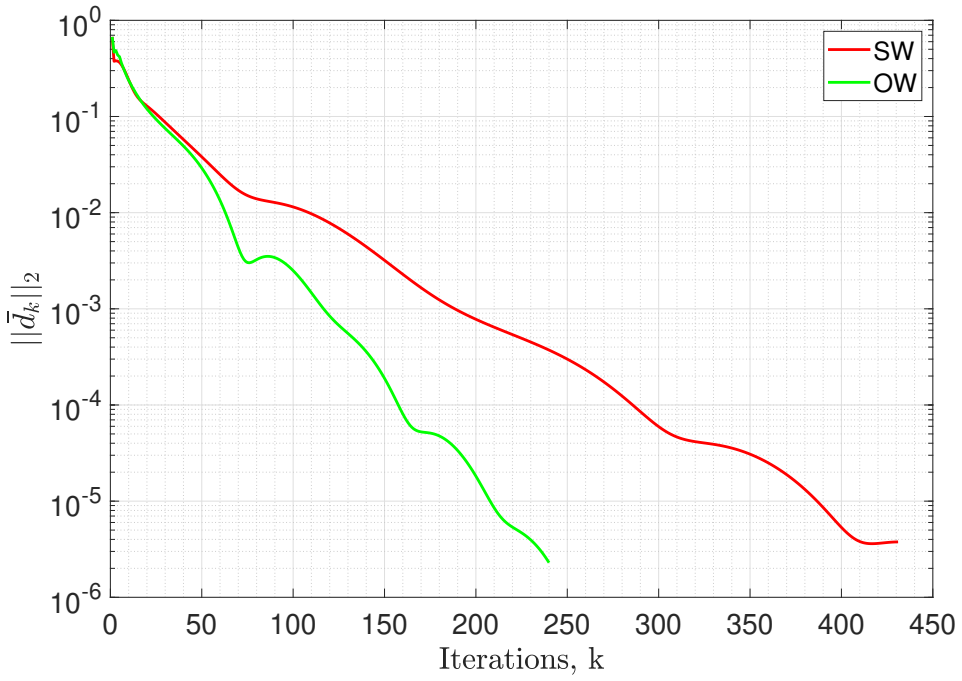


Figure 3.4. Comparison of convergence trend between the proposed F-TADMM and the baseline algorithm with $N = 10$.

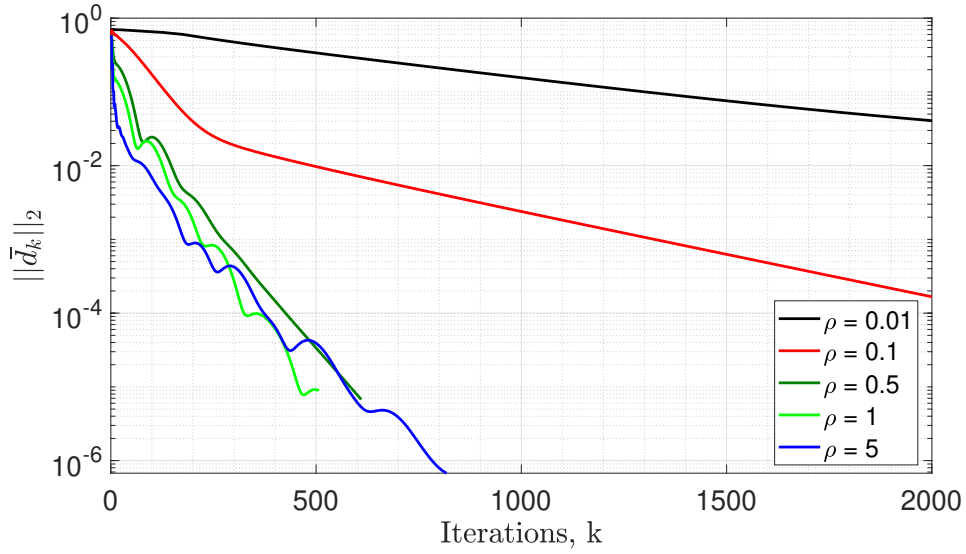


Figure 3.5. Comparison of convergence trend between the proposed F-TADMM with different ADMM penalty factor $\rho = \{0.01, 0.1, 0.5, 1, 5\}$.

4. APPLICATION TO FORMATION FLYING

In this section, the trajectory planning of multiple vehicle coordination problem is considered. Here each vehicle pursues private objectives as well as a linearly coupled formation constraints. To apply F-TADMM to formation flying, we propose the following problem formulation. Let $\hat{x}_{\mathrm{i}}(k) \in \mathbb{R}^n$ and $\hat{u}_{\mathrm{i}}(k) \in \mathbf{R}^m$, respectively, represent the n-dimensional state vector and m-dimensional input vector of agent i at time step k. Then for each vehicle $\mathrm{i} \in \{1, \ldots, N\}$, the variables, $X_{\mathrm{i}} \in \mathbb{R}^{nK}$ and $U_{\mathrm{i}} \in \mathbb{R}^{dK}$, are introduced as the trajectory of state and the control input of the vehicle over the next K time steps. Hence,

$$X_{\mathrm{i}} = \begin{bmatrix} \hat{x}_{\mathrm{i}}^T(1) \\ \hat{x}_{\mathrm{i}}^T(2) \\ \vdots \\ \hat{x}_{\mathrm{i}}^T(K+1) \end{bmatrix}, \quad U_{\mathrm{i}} = \begin{bmatrix} \hat{u}_{\mathrm{i}}^T(1) \\ \hat{u}_{\mathrm{i}}^T(2) \\ \vdots \\ \hat{u}_{\mathrm{i}}^T(K+1) \end{bmatrix}.$$

The trajectory of position of each vehicle is designated by $P_{\mathrm{i}} = CX_{\mathrm{i}}$ such that $C \in \mathbb{R}^{\frac{nK}{2} \times n}$

4.1 F-TADMM to Formation flying

To demonstrate the application of F-TADMM to formation flying, we first make the decision variable vector as the input trajectory U_{i} for all agents. Therefore, using the system dynamics, we identify the relationship between U_{i} and X_{i}, and write the optimization problem in terms of U_{i} alone.

4.1.1 Agent Prediction Model

Here, the individual agent dynamics is considered as a second-order system given by,

$$\hat{x}_{\mathrm{i}}(k+1) = A_d \hat{x}_{\mathrm{i}}(k) + B_d \hat{u}_{\mathrm{i}}(k) \tag{4.1}$$

where the system matrices are given as:

$$A_d = \begin{bmatrix} I & dtI \\ 0 & I \end{bmatrix}, \quad B_d = \begin{bmatrix} \frac{dt^2}{2} I \\ dtI \end{bmatrix}$$

and dt is the time step of prediction. These are standard assumptions in many practical applications where agents are quadcopters [22] or aircraft [23]. Knowing the definition of state trajectory,

$$X_{\mathrm{i}} = H\hat{x}_{\mathrm{i}}(0) + GU_{\mathrm{i}} \tag{4.2}$$

where,

$$H = \begin{bmatrix} A_d \\ A_d^2 \\ \vdots \\ A_d^T \end{bmatrix}, \quad G = \begin{bmatrix} B_d & 0 & \dots & 0 \\ A_d B_d & B_d & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ A_d^{T-1} B_d & A_d^{T-2} B_d & \dots & B_d \end{bmatrix}$$

4.1.2 Desired Formation as Equality Constraints

Further, as generally assumed in the literature, [23], [19] that the agents know the desired formation locally represented by

$$\Delta P_{\mathrm{ij}} = P_{\mathrm{i}} - P_{\mathrm{j}}, \quad \forall \mathrm{j} \in \mathcal{N}_{\mathrm{i}}$$

These constraints are rewritten as:

$$\begin{aligned} P_{\mathrm{i}} - P_{\mathrm{j}} &= \Delta P_{\mathrm{ij}} \\ CX_{\mathrm{i}} - CX_{\mathrm{j}} &= \frac{\Delta P_{\mathrm{ij}}}{2} - \frac{\Delta P_{\mathrm{ji}}}{2} \\ C(GU_{\mathrm{i}} + H\hat{x}_{\mathrm{i}}(0)) - C(GU_{\mathrm{j}} + H\hat{x}_{\mathrm{i}}(0)) &= \frac{\Delta P_{\mathrm{ij}}}{2} - \frac{\Delta P_{\mathrm{ji}}}{2} \\ CGU_{\mathrm{i}} - CGU_{\mathrm{j}} &= \underbrace{\left[\frac{\Delta P_{\mathrm{ij}}}{2} - H\hat{x}_{\mathrm{i}}(0)\right]}_{1} - \underbrace{\left[\frac{\Delta P_{\mathrm{ji}}}{2} - H\hat{x}_{\mathrm{j}}(0)\right]}_{2}. \end{aligned} \tag{4.3}$$

It is noted that the terms on the RHS are available to local agents only. Also, we use the fact that $\Delta P_{\mathrm{ij}} = -\Delta P_{\mathrm{ji}}$. Thus, the equality constraints for the desired formation are converted into the coupling constraints for F-TADMM using the definition of edge-node incidence matrix M_{i} as follows,

$$A_{\mathrm{i}} = M_{\mathrm{i}}^T \otimes CG,$$

$$[b_{\mathrm{i}}]_{\mathrm{j}th-block} = \begin{cases} 0, & (\mathrm{i},\mathrm{j}) \notin \mathcal{E}, \\ \left[\frac{\Delta P_{\mathrm{ij}}}{2} - H\hat{x}_{\mathrm{i}}(0)\right] & \text{otherwise.} \end{cases}$$

4.1.3 Local objective function

In practice, private objectives range from fuel consumption minimization to target tracking to trajectory optimization along a specified path. Here, we consider the target tracking and define

$$f_{\mathrm{i}}(X_{\mathrm{i}}, U_{\mathrm{i}}) = \left\|CX_{\mathrm{i}} - P_{\mathrm{i},target}\right\|_2^2 + \left\|U_{\mathrm{i}}\right\|_2^2 \tag{4.4}$$

where CX_{i} and $P_{\mathrm{i},target}$ are, respectively, the position and the desired final position of agent i. The objective function is rewritten in terms of the decision variable U_{i} alone as:

$$f_{\mathrm{i}}(U_{\mathrm{i}}) = \left\|C(GU_{\mathrm{i}} + H\hat{x}_{\mathrm{i}}(0)) - P_{\mathrm{i},target}\right\|_2^2 + \left\|U_{\mathrm{i}}\right\|_2^2. \tag{4.5}$$

Since the optimization problem is to minimize a positive definite quadratic form objective function over a linear set, it is a convex problem. Again, the network weights are calculated using (3.20).

4.1.4 Input Constraints

In practical conditions, it is not always possible to get the arbitrarily large input. Therefore, we consider that the agent input is constrained as

$$\hat{u}_{\mathrm{i}}^{min} \leq \hat{u}_{\mathrm{i}}(k) \leq \hat{u}_{\mathrm{i}}^{max} \quad \forall k \in [0, K]$$

4.2 Simulation Results

Using the above formulation, the F-TADMM algorithm is applied to a multi-agent system of 10 agents with a randomly generated graph topology. This random communication graph and the desired formation are represented in Fig. 4.1. In Fig. 4.2, the formation flying scenario is presented. The agent trajectories are considered to be in the 2-d plane (p_x, p_y). Each agent's state is randomly initialized as seen in Fig. 4.2. Also from Fig. 4.2, it is observed that the proposed F-TADMM algorithm makes the network of agents successfully attain the desired formation.

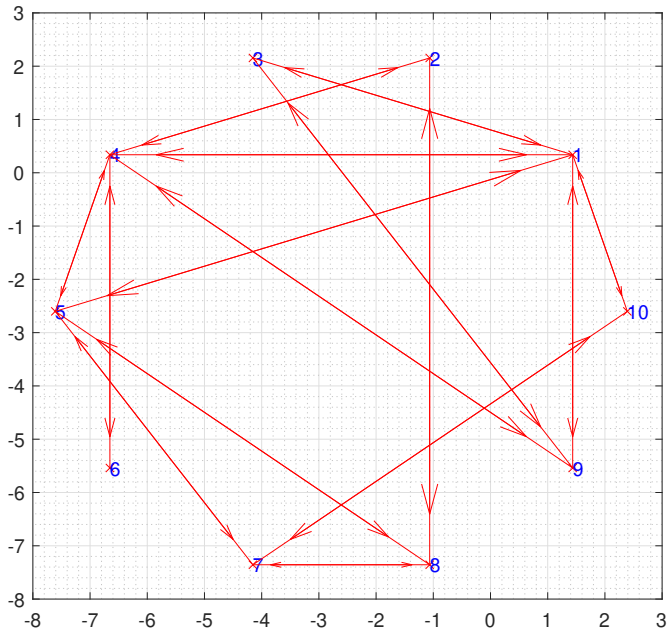


Figure 4.1. Desired Formation and randomly generated communication graph topology for $N = 10$ agents.

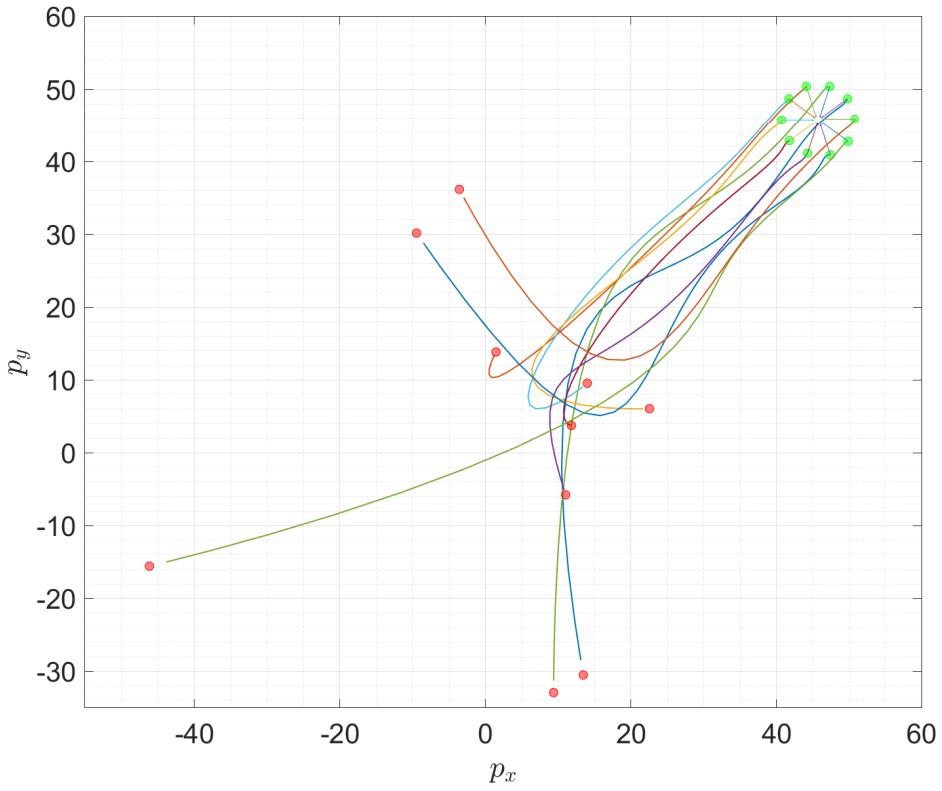


Figure 4.2. Individual agent trajectories show that the proposed F-TADMM algorithm successfully achieves the desired formation.

5. CONVERGENCE ANALYSIS FOR TIME-VARYING TOPOLOGY

In this chapter, we extend the TADMM algorithm for iteratively solving problem ((2.2)) but with time-varying network topology. We begin our analysis after re-stating the Distributed Computation Framework as described in Section (2.2).

5.1 Distributed Computation Framework with Time-Varying Topology

Here we describe the network topology with time varying network and state the necessary assumptions. The underlying, undirected communication graph of the network at the instant k is denoted by $\mathcal{G}^k = (\mathcal{V}, \mathcal{E}^k)$, where $\mathcal{V} = \{1, \ldots, N\}$ is the set of nodes, representing the agents, which do not change with time, and $\mathcal{E}^k \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, representing the communication links. An edge $(\mathrm{i}, \mathrm{j}) \in \mathcal{E}^k$ if agent i receives information from agent j and vice versa at instant k. The neighborhood of agent i in $\mathcal{B}^k$ is denoted by $\mathcal{N}_\mathrm{i}^k = \{\mathrm{j} \in \mathcal{V} \mid (\mathrm{i}, \mathrm{j}) \in \mathcal{E}^k\}$, with $(\mathrm{i}, \mathrm{i}) \in \mathcal{E}^k, \forall \mathrm{i} \in \{1, \ldots, N\}$. Each edge $(\mathrm{i}, \mathrm{j}) \in \mathcal{E}^k$ has an associated weight w_ij^k, which measures how much agent i values the information received from agent j. The time varying weight matrix of the entire network is denoted by $W_k \in \mathbb{R}^{N \times N}$, where w_ij^k represents the $(\mathrm{i}, \mathrm{j})^{th}$ element of W_k, such that

$$w_\mathrm{ij}^k = \begin{cases} 0, & (\mathrm{i}, \mathrm{j}) \notin \mathcal{E}^k, \\ \geq 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

Additionally, we consider the following assumption on these network weights,

Assumption 5.1.1 (Network Properties for each $\mathcal{B}^k$)**.** *The communication graph $\mathcal{B}^k$ is undirected and connected $\forall k$. The associated weight matrix W_k has the following properties $\forall k$:*

1. *W_k is balanced, i.e., $w_\mathrm{ij}^k = w_\mathrm{ji}^k, \forall (\mathrm{i}, \mathrm{j}) \in \mathcal{E}^k$,*

2. W_k is a doubly stochastic matrix, i.e.,

$$W_k \hat{\mathbf{1}}_N = \hat{\mathbf{1}}_N, \quad W_k^T \hat{\mathbf{1}}_N = \hat{\mathbf{1}}_N.$$

3. W_k is a positive semidefinite, ie, $W_k \succeq 0$

5.2 Modified TADMM updates

We rewrite the TADMM algorithm with time varying network topology as shown below in Algorithm 3 as Time Varying Tracking ADMM (TV-ADMM). It is noted that the case of time varying network topology implies time varying network weights.

Algorithm 3 TV-TADMM with Time varying weights

procedure INITIALIZATION
  $x_{\mathrm{i}}^0 \in \mathcal{X}_{\mathrm{i}}$, $\lambda_{\mathrm{i}}^0 \in \mathcal{X}_{\mathrm{i}}$, $d_{\mathrm{i}}^0 = A_{\mathrm{i}} x_{\mathrm{i}}^0 - b_{\mathrm{i}}$
end procedure
procedure REPEAT TILL CONVERGENCE
  Communicate to neighbors: $(d_{\mathrm{i}}^k, \lambda_{\mathrm{i}}^k)$
  Compute Deviation: $\Delta d_{\mathrm{i}}^k = \sum_{\mathrm{j} \in \mathcal{N}_{\mathrm{i}}} w_{\mathrm{i,j}}^k d_{\mathrm{j}}^k$
  Compute Dual Variable: $\Delta \lambda_{\mathrm{i}}^k = \sum_{\mathrm{j} \in \mathcal{N}_{\mathrm{i}}} w_{\mathrm{i,j}}^k \lambda_{\mathrm{j}}^k$
  Local Optimization: $x_{\mathrm{i}}^{k+1} = \arg\min_{x \in \mathcal{X}_{\mathrm{i}}} \{ f_{\mathrm{i}}(x) + (\Delta \lambda_{\mathrm{i}}^k)^T A_{\mathrm{i}} x + \frac{\rho}{2} || A_{\mathrm{i}} x - A_{\mathrm{i}} x_{\mathrm{i}}^k + \Delta d_{\mathrm{i}}^k ||^2 \}$
  Update Deviation: $d_{\mathrm{i}}^{k+1} = \Delta d_{\mathrm{i}}^k + A_{\mathrm{i}}(x_{\mathrm{i}}^{k+1} - x_{\mathrm{i}}^k)$
  Update Dual Variable: $\lambda_{\mathrm{i}}^{k+1} = \lambda_{\mathrm{i}}^k + \rho d_{\mathrm{i}}^{k+1}$
end procedure

After re-introducing the concatenated vectors for the required from (3.1) and (3.2), we get the following updates, written compactly as,:

$$\begin{aligned}
&(1)\, \mathbf{x}_{k+1} = \arg\min_{\mathbf{x} \in \mathbf{X}} \{ \boldsymbol{f}(\mathbf{x}) + \boldsymbol{\Delta} \mathbf{d}_k^T \mathbf{A}_d \mathbf{x} + \frac{\rho}{2} || \mathbf{A}_d \mathbf{x} - \mathbf{A}_d \mathbf{x}_k + \boldsymbol{\Delta} \lambda_k ||^2 \} \\
&(2)\, \mathbf{d}_{k+1} = \mathbf{W}_k \mathbf{d}_k + \mathbf{A}_d \mathbf{x}_{k+1} - \mathbf{A}_d \mathbf{x}_k \\
&(3)\, \boldsymbol{\lambda}_{k+1} = \mathbf{W}_k \boldsymbol{\lambda}_k + \rho \mathbf{d}_{k+1}
\end{aligned} \tag{5.2}$$

5.3 Convergence Analysis for Time Varying topology

We begin our analysis with similar way as shown in [18]. We show that the Lemma 1, 2, and 3 and Preposition 1, [18], will still be true for a time varying network topology as described in (5.1).

Lemma 5.3.1. *(Tracking property). Under Assumption (5.1.1) it holds that*

$$\bar{d_k} = \frac{1}{N}\left(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} x_{\mathrm{i},k} - b\right) \tag{5.3}$$

for all $k \geq 0$

Proof See Proof for Lemma 1 in [18], but where $\mathbf{W}$ is mentioned we consider $\mathbf{W}_k$ and using the column stochastic of each $W_k \forall k$ on step 3, we conclude to the same result. □

Lemma 5.3.2. *(Average dual update). Under Assumption (5.1.1), it holds that*

$$\bar{\lambda}_{k+1} = \bar{\lambda}_k + \rho \bar{d}_{k+1} \tag{5.4}$$

for all $k \geq 0$.

Proof See Proof for Lemma 2 in [18], but again where $\mathbf{W}$ is mentioned we consider $\mathbf{W}_k$ and using the column stochastic of $W_k \forall k$ on step 2, we conclude to the same result. □

Now we rewrite the error dynamics in equation (3.12) with W_k. Defining $\tilde{\mathbf{W}}_k = (W_k - \mathcal{O}) \otimes I_p$ and using the analogous math as described in section (3.1), the overall error dynamics for time varying case is given as:

$$\mathrm{e}_{k+1} = \mathbf{A}_k \mathrm{e}_k + \mathbf{B}\left(\boldsymbol{z}_{k+1} - \boldsymbol{z}_k\right) \tag{5.5}$$

where

$$\mathrm{e}_k = \begin{bmatrix} \mathrm{e}_k^{\boldsymbol{\lambda}} \\ \rho \mathrm{e}_k^{\boldsymbol{d}} \end{bmatrix}, \quad \mathbf{A}_k = \begin{bmatrix} \tilde{\mathbf{W}}_k & \tilde{\mathbf{W}}_k \\ 0 & \tilde{\mathbf{W}}_k \end{bmatrix}, \quad \mathbf{B} = \rho \begin{bmatrix} I \\ I \end{bmatrix}$$

From Assumption (2.2.1), $\left\|W_k - \mathcal{O}\right\| < 1 \ \forall k$. Therefore, all the eigenvalues of $\tilde{\mathbf{W}}_k$ lie within the unit circle at each instant. This implies that the dynamical system describing the evolution of the consensus errors in (5.5) is asymptotically stable.

Since the proof of Lemma 3 and Preposition 1 [18] do not any require the network properties, they still hold true under the assumption (5.1.1). Therefore we borrow their proof to state the following lemmas.

Lemma 5.3.3. *(Bounded sequences). Under Assumptions (2.1.1) and (5.1.1) we have that (i) the sequences* $\{\boldsymbol{x}_k\}_{k\geq 0}$ *and* $\{\boldsymbol{z}_k\}_{k\geq 0}$ *are bounded; (ii) the sequences* $\left\{\mathrm{e}_k^d\right\}_{k\geq 0}$ *and* $\left\{\mathrm{e}_k^{\boldsymbol{\lambda}}\right\}_{k\geq 0}$ *are bounded.*

Proof See Lemma 3, [18]. □

Lemma 5.3.4. *(Local optimality). Under Assumptions (2.1.1) and (2.1.2) we have that*

$$\begin{aligned} &\|\overline{\boldsymbol{\lambda}}_{k+1} - \boldsymbol{\lambda}^\star\|^2 + 2\rho\left[\boldsymbol{z}_{k+1} - \mathbf{A}_d\boldsymbol{x}^\star\right]^\top \mathbf{e}_{k+1}^{\boldsymbol{\lambda}} \\ &\qquad \leq \left\|\overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star\right\|^2 - \left\|\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k\right\|^2, \end{aligned} \tag{5.6}$$

for any optimal solution pair $(\boldsymbol{x}^\star, \lambda^\star)$ *for (2.3) and (2.4), where we set* $\boldsymbol{\lambda}^\star = \hat{\mathbf{1}}_N \otimes \lambda^\star$.

Proof See Preposition 1, [18]. □.

Now we have all the preliminaries required to prove that the modified updates in Algorithm 3 converge as summarized in theorem below.

Theorem 5.3.5. *Under Assumptions (2.1.1), (2.1.2), (5.1.1) the sequences generated by algorithm (3) satisfy:*

(i) $\lim_{k\to\infty} \left\|\mathbf{e}_k^{\boldsymbol{d}}\right\| = 0$,

(ii) $\lim_{k\to\infty} \left\|\mathbf{e}_k^{\boldsymbol{\lambda}}\right\| = 0$,

(iii) $\lim_{k\to\infty} \left\|\bar{d}_k\right\| = 0$,

(iv) $\left\{\left\|\overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star\right\|^2 + \rho^2 \left\|\boldsymbol{z}_k - A_d\boldsymbol{x}^\star\right\|^2\right\}_{k\geq 0}$ *is convergent,*

for any optimal solution pair $(\boldsymbol{x}^\star, \lambda^\star)$ *for (2.3) and (2.4), with* $\boldsymbol{\lambda}^\star = \hat{\mathbf{1}}_N \otimes \lambda^\star$.

The main idea behind the proof is as following. We start by considering (5.5) together as a discrete-time varying dynamical system for the consensus updates. Since the consensus system is asymptotically stable, we then build a piecewise positive definite quadratic

Lyapunov function which after each iteration guarantees contraction of solution. However, since the consensus system is not autonomous, the inequality describing the variation of such Lyapunov function across each iteration contains terms that are not defined in sign, so that it is no longer necessarily decreasing. The idea is to properly combine such the inequality in (5.6) so as to "balance out" the terms that are not defined in sign. In this way, we obtain an aggregate descent condition that allows us to prove the asymptotic stability of the overall (nonlinear) dynamical system modeling algorithm (3).

Proof of Theorem (5.3.5) Let a positive definite symmetric matrix $P_k = P_k^T \succ 0$,

$$\|B\boldsymbol{u}_{k+1} - \mathbf{e}_{k+1}\|_{P_k}^2 \tag{5.7}$$

$$= \|B\boldsymbol{u}_k - \mathrm{e}_k + (I - \mathbf{A}_k)\mathrm{e}_k\|_{P_k}^2 \tag{5.8}$$

$$= \|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 + 2\left[B\boldsymbol{u}_k - \mathrm{e}_k\right]^\top P_k(I - \mathbf{A}_k)\mathrm{e}_k + \mathrm{e}_k^\top \left(I - \mathbf{A}_k^\top\right) P_k(I - \mathbf{A}_k)\mathrm{e}_k \tag{5.9}$$

$$= \|B\boldsymbol{u}_k - \mathrm{e}_k\Big\|_{P_k}^2 - \Big\|\mathrm{e}_k\|_{P_k - F_k^T P_k F_k}^2 + 2\boldsymbol{u}_k^\top B^T P_k(I - F_k)\mathrm{e}_k \tag{5.10}$$

Now use this equality both sides of inequality in lemma (5.3.4).

$$\begin{aligned}\left\|\overline{\boldsymbol{\lambda}}_{k+1} - \boldsymbol{\lambda}^\star\right\|^2 + 2\rho\boldsymbol{u}_{k+1}^\top H\mathbf{e}_{k+1} + \|B\boldsymbol{u}_{k+1} - \mathbf{e}_{k+1}\|_{P_k}^2 \\ \leq \underbrace{\left\|\bar{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star\right\|^2}_{a} - \underbrace{\left\|\bar{\boldsymbol{\lambda}}_{k+1} - \boldsymbol{\lambda}_k\right\|^2}_{b} + \underbrace{\left\|B\boldsymbol{u}_k - \mathbf{e}_k\right\|_{P_k}^2}_{c} \\ - \underbrace{\left\|\mathbf{e}_k\right\|_{P_k - \mathbf{A}_k^T P_k \mathbf{A}_k}^2}_{d} + \underbrace{2\boldsymbol{u}_k^\top B^\top P_k(I - \mathbf{A}_k)\mathrm{e}_k}_{\mathrm{e}}\end{aligned} \tag{5.11}$$

where $H = [I \quad 0]$. Now it is observed that if we choose appropriate P_k, such that $P_k - \mathbf{A}_k^T P_k \mathbf{A}_k \succ 0$, then term (d) will be always positive. Further term (b) is always positive. Also if $H = B^T P_k(I - \mathbf{A}_k)$, the overall inequality in (5.30), guarantees a descent condition for all $k \to k + 1$.

These conditions are rewritten as

$$H = B^T P_k (I - \mathbf{A}_k) \tag{5.12}$$

$$P_k - \mathbf{A}_k^T P_k \mathbf{A}_k \succ 0 \tag{5.13}$$

$$P_k \succ 0 \tag{5.14}$$

Take P_k partitioned in blocks as follows

$$P_k = \begin{bmatrix} P_{k,1} & P_{k,2} \\ P_{k,2}^\top & P_{k,3} \end{bmatrix}$$

Owing to the fact that all eigenvalues of $\tilde{\mathbf{W}}_k$ lies in the open unit circle, thus $(I - \tilde{\mathbf{W}}_k)$ is invertible and, from the equality $MM^{-1} = M^{-1}M = I$ with $M = (I - \tilde{\mathbf{W}}_k)$, we have the following identities

$$\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} = (I - \tilde{\mathbf{W}}_k)^{-1} - I \tag{5.15}$$

$$(I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k = (I - \tilde{\mathbf{W}}_k)^{-1} - I \tag{5.16}$$

Noticing that

$$(I - \mathbf{A}_k)^{-1} = \begin{bmatrix} (I - \tilde{\mathbf{W}}_k)^{-1} & (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \\ 0 & (I - \tilde{\mathbf{W}}_k)^{-1} \end{bmatrix}$$

condition (5.12) can be rewritten as $B^T P_k = H(I - \mathbf{A}_k)^{-1}$ and translates into the following constraints on the blocks of P_k :

$$P_{k,1} + P_{k,2}^T = (I - \tilde{\mathbf{W}}_k)^{-1} \tag{5.17}$$

$$P_{k,2} + P_{k,3} = (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tag{5.18}$$

$$\overset{(a)}{=} (I - \tilde{\mathbf{W}}_k)^{-2} - (I - \tilde{\mathbf{W}}_k)^{-1} \tag{5.19}$$

$$P_{k,1} + P_{k,2}^T + P_{k,2} + P_{k,3} = (I - \tilde{\mathbf{W}}_k)^{-2} \tag{5.20}$$

where in (a) we used (5.15), while follows by simply summing (5.17) and (5.19). Note that, from (5.17), we deduce that $P_{k,2} = P_{k,2}^T$ since both $P_{1,k}$ and $(I - \tilde{\mathbf{W}}_k)^{-1}$ will be symmetric. Now we can write LHS of (5.13) as a function of $P_{k,1}$ alone, using blocks and above properties,

$$\begin{bmatrix} P_{k,1} - \tilde{\mathbf{W}}_k P_{k,1} \tilde{\mathbf{W}}_k & P_{k,1} - \tilde{\mathbf{W}}_k \left(P_{k,1} + P_{k,2}\right) \tilde{\mathbf{W}}_k \\ P_{k,2}^\top - \tilde{\mathbf{W}}_k \left(P_1 + P_{k,2}^\top\right) \tilde{\mathbf{W}}_k & P_{k,3} - \tilde{\mathbf{W}}_k \left(P_{k,1} + P_{k,2}^\top + P_{k,2} + P_{k,3}\right) \tilde{\mathbf{W}}_k \end{bmatrix} \tag{5.21}$$

$$\overset{\text{(a)}}{=} \begin{bmatrix} P_{k,1} - \tilde{\mathbf{W}}_k P_{k,1} \tilde{\mathbf{W}}_k & P_{k,2} - \tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k \\ P_{k,2}^\top - \tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k & P_{k,3} - \tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-2} \tilde{\mathbf{W}}_k \end{bmatrix} \tag{5.22}$$

$$\overset{\text{(b)}}{=} \begin{bmatrix} P_{k,1} - \tilde{\mathbf{W}}_k P_{k,1} \tilde{\mathbf{W}}_k & \tilde{\mathbf{W}}_k - (P_{k,1} - I) \\ \tilde{\mathbf{W}}_k - (P_{k,1} - I) & P_{k,1} - I \end{bmatrix} \tag{5.23}$$

where in (a) we used (5.17) and in (b) we leveraged identities in (5.15) together with (5.17) and (5.19) to express $P_{k,2}$ and $P_{k,3}$ as a function of $P_{k,1}$ only. Next, we find a value for $P_{k,1} \succ 0$ ensuring that $P_k - \mathbf{A}_k^T P_k \mathbf{A}_k F$ is positive definite. This requirement can be posed in terms of its blocks by means of the Schur complement, i.e., $P_k - \mathbf{A}_k^T P_k \mathbf{A}_k \succ 0$ if and only if

$$P_{k,1} - I \succ 0, \tag{5.24}$$

$$P_{k,1} - \tilde{\mathbf{W}}_k P_{k,1} \tilde{\mathbf{W}}_k - \left(\tilde{\mathbf{W}}_k - (P_{k,1} - I)\right) (P_{k,1} - I)^{-1} \left(\tilde{\mathbf{W}}_k - (P_{k,1} - I)\right) \tag{5.25}$$

$$= 2\tilde{\mathbf{W}}_k + I - \tilde{\mathbf{W}}_k \left(P_{k,1} + (P_{k,1} - I)^{-1}\right) \tilde{\mathbf{W}}_k \succ 0 \tag{5.26}$$

Now if we choose $P_{k,1} = 2I$. And letting $V^T \Lambda_k V = \tilde{\mathbf{W}}_k$ be the eigenvalue decomposition of $\tilde{\mathbf{W}}_k$, we have

$$\begin{aligned} 2\tilde{\mathbf{W}}_k + I - \tilde{\mathbf{W}}_k (2I + I) \tilde{\mathbf{W}}_k &= 2\tilde{\mathbf{W}}_k + I - 3\tilde{\mathbf{W}}_k^2 \\ &= V \left(I + 2\Lambda - 3\Lambda^2\right) V^T \succ 0 \end{aligned} \tag{5.27}$$

which is equivalent to $I + 2\Lambda - 3\Lambda^2 \succ 0$ and always holds true when all the eigenvalues of $\tilde{\mathbf{W}}_k$ lies within $(-1/3, 1)$ based on assumption (5.1.1). Finally, using again Schur's com-

plement, it is also easy to prove that $P_{k,1} = 2I$ satisfies condition $P_k \succ 0$. In fact, we have that

$$P_k = \begin{bmatrix} 2I & (I - \tilde{\mathbf{W}}_k)^{-1} - 2I \\ (I - \tilde{\mathbf{W}}_k)^{-1} - 2I & (I - \tilde{\mathbf{W}}_k)^{-2} - 2(I - \tilde{\mathbf{W}}_k)^{-1} + 2I \end{bmatrix} \succ 0 \tag{5.28}$$

if and only if

$$\begin{aligned} &2I \succ 0, \\ &(I - \tilde{\mathbf{W}}_k)^{-2} - 2(I - \tilde{\mathbf{W}}_k)^{-1} + 2I - \tfrac{1}{2}\left((I - \tilde{\mathbf{W}}_k)^{-1} - 2I\right)^2 \\ &= \tfrac{1}{2}(I - \tilde{\mathbf{W}}_k)^{-2} \succ 0 \end{aligned} \tag{5.29}$$

Above condition (5.29) is satisfied since $\tilde{\mathbf{W}}_k$ has all eigenvalues in the open unit circle and hence $I - \tilde{\mathbf{W}}_k \succ 0$.

Remark: It is noted that this analysis will provide that same result regardless of the iteration count k as far as each W_k satisfies assumption (5.1.1).

From above results we can rewrite (5.30) as

$$\begin{aligned} &\left\|\overline{\boldsymbol{\lambda}}_{k+1} - \boldsymbol{\lambda}^\star\right\|^2 + 2\boldsymbol{u}_{k+1}^\top H \mathrm{e}_{k+1} + \|B\boldsymbol{u}_{k+1} - \mathrm{e}_{k+1}\|_{P_k}^2 \\ &\qquad \le \left\|\overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star\right\|^2 + 2\boldsymbol{u}_k^\top H \mathrm{e}_k + \|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 \quad - \left\|\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k\right\|^2 - \|\mathrm{e}_k\|_{Q_k}^2 \end{aligned} \tag{5.30}$$

where $Q_k = P_k - \mathbf{A}_k^T P_k \mathbf{A}_k$. Now rearranging the terms on and summing from $k = 0$ to $s-1$, for any $s \in \mathbb{N}$, we have

$$\sum_{k=0}^{s-1} \|\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k\|^2 + \|\mathrm{e}_k\|_{Q_k}^2 \tag{5.31}$$

$$\le \left\|\overline{\boldsymbol{\lambda}}_0 - \boldsymbol{\lambda}^\star\right\|^2 + 2\boldsymbol{u}_0^\top H \mathrm{e}_0 + \sum_{k=0}^{s-1} \|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 - \|B\boldsymbol{u}_{k+1} - \mathrm{e}_{k+1}\|_{P_k}^2 \tag{5.32}$$

$$- \left\|\overline{\boldsymbol{\lambda}}_s - \boldsymbol{\lambda}^\star\right\|^2 - 2\boldsymbol{u}_s^\top H \mathrm{e}_s \tag{5.33}$$

$$\le \left\|\overline{\boldsymbol{\lambda}}_0 - \boldsymbol{\lambda}^\star\right\|^2 + \underbrace{\sum_{k=0}^{s-1} \|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 - \|B\boldsymbol{u}_{k+1} - \mathrm{e}_{k+1}\|_{P_k}^2}_{a} + 2\mathcal{C} \tag{5.34}$$

where in the second inequality we neglected the terms $-\left\|\overline{\boldsymbol{\lambda}}_M - \boldsymbol{\lambda}^\star\right\|^2$ and $-\|B\boldsymbol{u}_s - \mathrm{e}_s\|_P^2$ in the RHS since they are non-positive (recall $P_k \succ 0$) and we used the fact that $|\ 2\boldsymbol{u}_k^T H\mathrm{e}$

$_k$ $|\leq \mathcal{C}$ for all $k \geq 0$ owing to the results of Lemma 5.3.4. Also the term (a) in last inequality, is

$$\begin{aligned}
&\sum_{k=0}^{s-1} \|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 - \|B\boldsymbol{u}_{k+1} - \mathrm{e}_{k+1}\|_{P_k}^2 \\
&\leq \sum_{k=0}^{s-1} \|B\boldsymbol{u}_k - \mathrm{e}_k - B\boldsymbol{u}_{k+1} + \mathrm{e}_{k+1}\|_{P_k}^2 \\
&\leq \sum_{k=0}^{s-1} \|(I - \mathbf{A}_k)\mathbf{e}_k\|_{P_k}^2 \\
&= \sum_{k=0}^{s-1} \|\mathbf{e}_k\|_{(I-\mathbf{A}_k)^T P_k (I-\mathbf{A}_k)}^2
\end{aligned} \tag{5.35}$$

where we use (3.12) and norm inequality $\big\|a\big\| - \big\|b\big\| \leq \big\|a-b\big\|$. Thus the LHS on 5.34 is upper bounded by

$$\sum_{k=0}^{s-1} \|\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k \Big\|^2 + \Big\| \mathrm{e}_k\|_{Q_k}^2 \leq \Big\|\overline{\boldsymbol{\lambda}}_0 - \boldsymbol{\lambda}^\star\Big\|^2 + \sum_{k=0}^{s-1} \|\mathbf{e}_k\|_{(I-\mathbf{A}_k)^T P_k (I-\mathbf{A}_k)}^2 + 2\mathcal{C} \tag{5.36}$$

$$\sum_{k=0}^{s-1} \|\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k \Big\|^2 + \Big\| \mathrm{e}_k\|_{Q_k - (I-\mathbf{A}_k)^T P_k (I-\mathbf{A}_k)}^2 \leq \Big\|\overline{\boldsymbol{\lambda}}_0 - \boldsymbol{\lambda}^\star\Big\|^2 + 2\mathcal{C} \tag{5.37}$$

Now we focus our attention to $Q_k - (I - \mathbf{A}_k)^T P_k (I - \mathbf{A}_k)$ as,

$$\begin{aligned}
Q_k - (I - \mathbf{A}_k)^T P_k (I - \mathbf{A}_k) &= P_k - \mathbf{A}_k^T P_k \mathbf{A}_k - (I - \mathbf{A}_k)^T P_k (I - \mathbf{A}_k) \\
&= \mathbf{A}_k^T P_k + P_k \mathbf{A}_k - 2\mathbf{A}_k^T P_k \mathbf{A}_k
\end{aligned}$$

Using the derived value of P_k and $\mathbf{A}_k$, we have

$$\mathbf{A}_k^T P_k + P_k \mathbf{A}_k = \begin{bmatrix} 4\tilde{\mathbf{W}}_k & 2\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k \\ 2\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k & 2\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-2} \tilde{\mathbf{W}}_k \end{bmatrix} \tag{5.38}$$

$$2\mathbf{A}_k^T P_k \mathbf{A}_k = \begin{bmatrix} 4\tilde{\mathbf{W}}_k^2 & 2\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k \\ 2\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-1} \tilde{\mathbf{W}}_k & 2\tilde{\mathbf{W}}_k (I - \tilde{\mathbf{W}}_k)^{-2} \tilde{\mathbf{W}}_k \end{bmatrix} \tag{5.39}$$

$$\mathbf{A}_k^T P_k + P_k \mathbf{A}_k - 2\mathbf{A}_k^T P_k \mathbf{A}_k = \begin{bmatrix} 4\tilde{\mathbf{W}}_k - 4\tilde{\mathbf{W}}_k^2 & 0 \\ 0 & 0 \end{bmatrix} \tag{5.40}$$

Now it is noted that $\tilde{\mathbf{W}}_k - \tilde{\mathbf{W}}_k^2 = \tilde{\mathbf{W}}_k(I - \tilde{\mathbf{W}}_k) \succ 0$ which implies that the $Q_k - (I - \mathbf{A}_k)^T P_k (I - \mathbf{A}_k) \succeq 0$. This implies that all the terms in LHS of 5.37 is always positive. Let $\bar{Q} = \min_k \sigma_{min,2}(Q_k - (I - \mathbf{A}_k)^T P_k (I - \mathbf{A}_k))I$ where $\sigma_{min,2}(.)$ corresponds to the second minimum eigenvalue. And it is noted that $\bar{Q} \succeq 0$ based on above analysis. Hence we get,

$$\sum_{k=0}^{\infty} \left(\left\| \overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k \right\|^2 + \|\mathrm{e}_k\|_{\bar{Q}}^2 \right) \leq \sum_{k=0}^{\infty} \left(\left\| \overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k \right\|^2 + \|\mathrm{e}_k\|_{Q_k - (I-\mathbf{A}_k)^T P_k (I-\mathbf{A}_k)}^2 \right) \tag{5.41}$$

Also it is noted that based on lemma 5.3.3, the RHS (5.37) shows that the sum of the series in finite because the $\left\| \overline{\boldsymbol{\lambda}}_0 - \boldsymbol{\lambda}^\star \right\|^2$ and $2\mathcal{C}$ are always bounded. Therefore,

$$\sum_{k=0}^{\infty} \left(\left\| \overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k \right\|^2 + \|\mathrm{e}_k\|_{\bar{Q}}^2 \right) \leq \infty \tag{5.42}$$

which, recalling that $\bar{Q} \succeq 0 \quad \forall k \geq 0$, implies that $\lim_{k\to\infty} \|\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k\| = 0$ and $\lim_{k\to\infty} \|\mathrm{e}_k\| = 0$. Hence, $\lim_{k\to\infty} \left\|\mathrm{e}_k^d\right\| = 0$ and $\lim_{k\to\infty} \left\|\mathrm{e}_k^\lambda\right\| = 0$, thus proving points (i) and (ii), respectively. Since $\overline{\boldsymbol{d}}_{k+1} = \frac{1}{\rho}\left(\overline{\boldsymbol{\lambda}}_{k+1} - \overline{\boldsymbol{\lambda}}_k\right)$, we also have that $\lim_{k\to\infty} \left\|\overline{\boldsymbol{d}}_k\right\| = 0$ and therefore $\lim_{k\to\infty} \left\|\bar{d}_k\right\| = 0$, thus proving point (iii). From equation (5.29), we also have that the sequence

$$\left\{ \left\| \overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star \right\|^2 + 2\boldsymbol{u}_k^\top H \mathrm{e}_k + \|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 \right\}_{k\geq 0}$$

is non-increasing, bounded below since $\left\|\boldsymbol{u}_k^\top H \mathrm{e}_k\right\| \leq \mathcal{C}$ and $\|B\boldsymbol{u}_k - \mathrm{e}_k\|_{P_k}^2 \geq 0$ (recall $P_k \succ 0$), and, therefore, convergent. Since $\|\mathrm{e}_k\|$ is vanishing and $\{\boldsymbol{u}_k\}_{k\geq 0}$ is bounded, we have that also the sequence $\left\{ \left\| \overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star \right\|^2 + \|B\boldsymbol{u}_k\|_{P_k}^2 \right\}_{k\geq 0}$ is convergent. A straightforward computation using (5.20) shows that $B^\top P_k B = (I - \tilde{\mathbf{W}}_k)^{-2} \succ 0$ which, recalling the definition of $\boldsymbol{u}_k = \rho\left(\boldsymbol{z}_k - \mathbf{A}_d \boldsymbol{x}^\star\right)$, implies that the sequence

$$\left\{ \left\| \overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star \right\|^2 + \rho^2 \left\| \boldsymbol{z}_k - A_d \boldsymbol{x}^\star \right\|_{(I-\tilde{\mathbf{W}}_k)^{-2}}^2 \right\}_{k\geq 0}$$

is convergent, which finally implies that

$$\left\{\left\|\overline{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}^\star\right\|^2 + \rho^2 \left\|\boldsymbol{z}_k - A_d \boldsymbol{x}^\star\right\|^2\right\}_{k\geq 0}$$

is convergent due to norm equivalence, thus proving point (iv) and concluding the proof. □

Now we prove that this converged value is actually the optimal solution. For the proof we observe that the same analysis as Theorem 2, [18] gives the desired result for a time varying topology, therefore only the theorem is stated here.

Theorem 5.3.6. *(Optimality) Under Assumptions (2.1.1), (2.1.2), (5.1.1) the sequences generated by algorithm (3) satisfy:*

(i) any limit point of the primal sequence $\{\boldsymbol{x}_k\}_{k\geq 0}$ *is an optimal solution* $\boldsymbol{x}^\star$ *of (2.3);*

(ii) each dual sequence $\{\lambda_{\mathrm{i},k}\}_{k\geq 0}, \mathrm{i} = 1, \ldots, N$, *converges to the same optimal solution* $\lambda^\star$ *of (2.4).*

Proof: See Theorem 2, [18]. □

5.4 Numerical Study under Switching Topology

Here we demonstrate the effectiveness of our approach, we again consider the problem of distributed optimization with linear coupling as consensus constraint. We test algorithm (3) on the same problem as discussed in section (2.2) but with switching between two strongly connected topologies. Here for both the cases we consider that the network weights for each topology is derived using Metropolis weights.

More specifically, two random communication graph are generated for $N = 20$ agents, as shown in Fig. (5.1). While simulations we consider that at every second time instant the graph to the other one and therefore we the weights are

$$w_{\mathrm{i,j}} = \begin{cases} \frac{1}{\max\{deg_{\mathrm{i}}^1, deg_{\mathrm{j}}^1\}}, & (\mathrm{i},\mathrm{j}) \in \mathcal{E}^1, mod\{k,2\} == 0 \\ \frac{1}{\max\{deg_{\mathrm{i}}^2, deg_{\mathrm{j}}^2\}}, & (\mathrm{i},\mathrm{j}) \in \mathcal{E}^2, else \end{cases} \tag{5.43}$$

where deg_{i}^k represents the degree of the agent i in the $k-th$ graph excluding the agent itself.

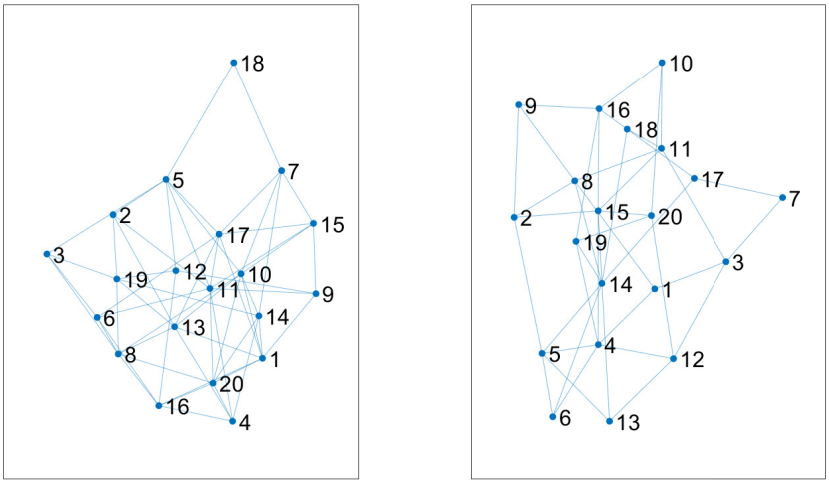


Figure 5.1. (a) The random network $\mathcal{G}^1$ and (b) the $\mathcal{G}^2$ for $N = 20$ agents;

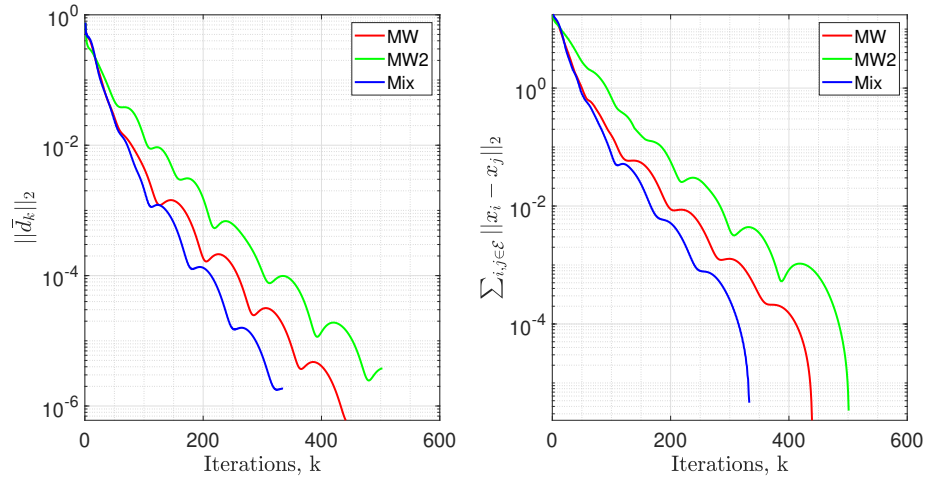


Figure 5.2. Comparison of convergence trend between the proposed Switching topology, Fixed topology with metropolis weights of $\mathcal{G}_1$ and $\mathcal{G}_2$, (a) iterations vs $\left\|\bar{d}_k\right\|_2$ (b) iterations vs consensus error;

Similar to earlier case the consensus constraint $\{x_1 = x_2 = \ldots = x_N\}$ can be written as: $\{\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} x_{\mathrm{i}} = \mathbf{0}\}$, where the coupling matrix A_{i} corresponds to the $\mathrm{i} - th$ column of the incidence matrix M of either of the graph as following:

$$A_{\mathrm{i}} = M_{\mathrm{i}}^T \otimes I_{n\times n}.$$

For the simulation problem, $x_{\mathrm{i}} \in \mathbb{R}^2, \forall \mathrm{i} \in \mathcal{V}$ and are initialized randomly. The local objective functions are designed as quadratic functions, i.e., $f_{\mathrm{i}}(x) = x^T P_{\mathrm{i}} x + q_{\mathrm{i}}^T x$, where P_{i} are random positive definite matrices. The local constraints are given as $x_{min}^{\mathrm{i}} \leq x_{\mathrm{i}} \leq x_{max}^{\mathrm{i}}$.

As it can be seen from the (5.2), the algorithm (3) reaches feasibility and optimality under a switching network topology. It is noted that the switching method outperforms the rate of convergence of as compared to individual network topology. Intuitively, this may be due to the fact that switching topology helps in the distribution of information in a better way as compared to any fixed topology. However further analysis on convergence rate with time varying communication graph is required to formally explain this behaviour.

6. CONCLUSION AND FUTURE WORK

In this work, we considered a general distributed optimization problem with coupling constraints. We presented formal guarantees on the convergence rate of the tracking alternating direction method of multipliers (TADMM) algorithm to solve this problem. Specifically, we showed that the convergence rate can be bounded by the norm of the communication weight matrix. Further, we optimized this weight matrix using a semi-definite programming (SDP) approach, thereby ensuring a faster convergence rate. This result was demonstrated on a sample problem and the results were compared with the baseline algorithm. We demonstrated the performance of the proposed Fast-TADMM (F-TADMM) algorithm via an illustrative example of distributed trajectory planning for formation flight. As part of further analysis we prove that under certain assumption on time-varying communication network, the modified algorithm TV-TADMM converges to optimal solution.

We also plan to extend the convergence rate analysis presented in this thesis for the case of a time-varying network topology and network weights. Additionally, we are looking into the problem of computing the optimal weight matrix in real time, to further improve the convergence rate of the proposed algorithm. We also plan to implement the F-TADMM algorithm and TV-TADMM algorithm to other practical applications such as Distributed Model Predictive Control and Distributed Sensor Estimation.

REFERENCES

[1] Y. Dong and J. Huang, "Consensus and flocking with connectivity preservation of uncertain euler–lagrange multi-agent systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 140, no. 9, 2018.

[2] W. Li, "The designated convergence rate problems of consensus or flocking of double-integrator agents with general nonequal velocity and position couplings: Further results and patterns of convergence rate contours," *IEEE transactions on cybernetics*, vol. 47, no. 5, pp. 1325–1335, 2017.

[3] X. Xu, L. Yang, W. Meng, Q. Cai, and M. Fu, "Multi-agent coverage search in unknown environments with obstacles: A survey," in *2019 Chinese Control Conference (CCC)*, IEEE, 2019, pp. 2317–2322.

[4] Z. Gu, Y. Wang, Q.-S. Hua, and F. Lau, *Rendezvous in Distributed Systems*. Springer, 2017.

[5] S. Khan, R. Deshmukh, and I. Hwang, "Optimal kalman consensus filter for weighted directed graphs," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, IEEE, 2019, pp. 7832–7837.

[6] M. Vinyals, J. A. Rodriguez-Aguilar, and J. Cerquides, "A survey on sensor networks from a multiagent perspective," *The Computer Journal*, vol. 54, no. 3, pp. 455–470, 2011.

[7] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid—the new and improved power grid: A survey," *IEEE communications surveys & tutorials*, vol. 14, no. 4, pp. 944–980, 2011.

[8] D. Jakovetić, J. Xavier, and J. M. Moura, "Fast distributed gradient methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, 2014.

[9] 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.

[10] G. França and J. Bento, "Distributed optimization, averaging via admm, and network topology," *Proceedings of the IEEE*, vol. 108, no. 11, pp. 1939–1952, 2020.

[11] C. Xi and U. A. Khan, "On the linear convergence of distributed optimization over directed graphs," *arXiv preprint arXiv:1510.02149*, 2015.

[12] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[13] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, IEEE, 2012, pp. 5445–5450.

[14] A. Nedić and J. Liu, "Distributed optimization for control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 77–103, 2018.

[15] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[16] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan, "A general analysis of the convergence of admm," in *International Conference on Machine Learning*, PMLR, 2015, pp. 343–352.

[17] Q. Ling, Y. Liu, W. Shi, and Z. Tian, "Weighted admm for fast decentralized network optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 22, pp. 5930–5942, 2016.

[18] A. Falsone, I. Notarnicola, G. Notarstefano, and M. Prandini, "Tracking-admm for distributed constraint-coupled optimization," *Automatica*, vol. 117, p. 108 962, 2020.

[19] R. Van Parys and G. Pipeleers, "Distributed mpc for multi-vehicle systems moving in formation," *Robotics and Autonomous Systems*, vol. 97, pp. 144–152, 2017.

[20] R. Vujanic, P. M. Esfahani, P. J. Goulart, S. Mariéthoz, and M. Morari, "A decomposition method for large scale milps, with performance guarantees and a power system application," *Automatica*, vol. 67, pp. 144–156, 2016.

[21] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[22] C. E. Luis and A. P. Schoellig, "Trajectory generation for multiagent point-to-point transitions via distributed model predictive control," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 375–382, 2019.

[23] R. L. Raffard, C. J. Tomlin, and S. P. Boyd, "Distributed optimization for cooperative agents: Application to formation flight," in *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, IEEE, vol. 3, 2004, pp. 2453–2459.