

**Balancing Information Mixing And Optimality:
A Framework For Robust And Efficient Distributed
Decision-Making**

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Vivek Khatana

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Professor. Murti V. Salapaka
Vincentine Hermes-Luh Chair
Department of Electrical and Computer Engineering

January, 2025

© Vivek Khatana 2025
ALL RIGHTS RESERVED

Acknowledgements

I am deeply grateful to the many individuals who have supported and guided me throughout my academic journey. First and foremost, I extend my heartfelt thanks to the remarkable high school teachers at Kendriya Vidyalaya No. 2, Air Force Station Jodhpur, whose belief in my potential and unwavering encouragement inspired me to strive for greater heights. I am equally grateful to my mentors at Resonance Jodhpur Center for their innovative teaching methods and for fostering an environment that consistently nurtured and encouraged scientific thinking.

My professors at my alma mater, the Indian Institute of Technology (IIT) Roorkee, particularly Prof. Anand Bulusu, Prof. Anil Kumar Gourishetty, Prof. Ram Manohar Singh, and Prof. Sumit Ghatak Choudhuri, were immensely inspiring mentors. They provided unwavering support for my decision to pursue research beyond my undergraduate studies. Their guidance and insightful interactions played a pivotal role in shaping my focus on control systems and energy systems as my primary areas of interest during graduate school.

First and foremost, I extend my deepest gratitude to my Ph.D. advisor, Prof. Murti Salapaka, whose unwavering emphasis on building strong foundations profoundly shaped my academic journey. His guidance encouraged me to prioritize mathematics courses early in my Ph.D., a decision that proved pivotal to my development. His mentorship not only inspired me to embark on this transformative path but also nurtured my growth as a researcher and an independent thinker. Under his supervision, I had my first serious engagement with advanced mathematics, marking the beginning of a deeply rewarding intellectual journey. I am also deeply grateful to his wife, Durga, and his children, Siddhartha and Pranav, who were an integral part of all the lab treats and social gatherings. I will fondly miss Durga's delicious food and the late-night discussions at their house, which often went on until 2 AM!

I am also sincerely grateful to the teachers and professors who supported me through

my coursework, helping to solidify my understanding of essential concepts. I would like to express my sincere gratitude to my doctoral committee members—Prof. Nicola Elia, Prof. Sairaj Dhople, Prof. Yulong Lu, Prof. Andrew Lamperski, and Prof. Paul Garrett—for their valuable time, insightful feedback, and unwavering support throughout my doctoral journey. I would also like to acknowledge Prof. Ankur Mani from the University of Minnesota, Twin Cities, for his invaluable guidance and feedback on numerous research topics and projects we collaborated on.

I am deeply grateful to my research colleagues at the *Nanodynamics Systems Laboratory* at the University of Minnesota, Twin Cities, for their invaluable support and collaboration. I would especially like to thank Dr. Govind Saraswat, Dr. Sourav Patel, Dr. Rachit Shrivastava, and Dr. Sivaraman Rajaganapathy for their wise advice and occasional mentorship. My heartfelt thanks also go to Dr. Soham Chakraborty and Dr. Arnab Dey for their unwavering support, research collaborations, and countless long days spent together on experiments and discussions. I am equally thankful to Dr. James Melbourne, Dr. Mishfad Shaikh Veedu, Connor Edlund, Dr. Harish Doddi, Mohammed Tuhin Rana, Cailong Hua, and Bijit Dey for their insightful discussions, engaging conversations, and fruitful research collaborations.

I would like to express my heartfelt gratitude to my internship mentors, Prof. Ravikumar Bhimasingu at IIT Hyderabad, and Dr. Chin-Yao Chang and Dr. Andrey Bernstein at the National Renewable Energy Laboratory, for providing me with invaluable internship opportunities. These experiences allowed me to conduct research in power and energy systems, expanding my understanding of the applications of my work to address some of the pressing challenges in power systems. Their guidance and support have been instrumental in shaping my passion for this field, which continues to be an inspiring and exciting direction for my research. I am also grateful to the University of Minnesota for admitting me to their graduate program and providing financial support through teaching and research assistantships. These opportunities allowed me to share my passion for control theory and power systems with engineering students, an experience that was both rewarding and enriching.

I would like to express my heartfelt gratitude to *my family away from home*, my closest friends—Dr. Shivam Sharma, Dr. Ramanish Singh, Vinitendra Singh, Niyati Panchal, Suman Lata, Divya Kumawat, Tracy Her, Nou-Chee Chang, Dr. Akshit Goyal, Dr. Tarun Gangwar, Dr. Pramesh Kumar, Dr. Anuj Mittal, Alisha Mittal, Dr. Kartik Patel, Dr. Omkar Joshi, Dr. Vindhya Prasad, Dr. Pranav Sharma, Dr. Akhil Singla, Dr. Akshay Jajoo, Shallabh Tiwari, and Abhishek Solanki, and my

furry friends Winnie and Dexter. I am also thankful to my colleagues and friends from Hindu Yuva UMN—Prasanna Salasiya, Resy Verma, Tej Mucherla, Datta Naikwade, Prathamesh Keni, Sanika Betawadkar Deo, Vaishnavi Mankala, and Kumud Verma—for their constant support, camaraderie, and encouragement throughout this journey. Their friendship is one of the invaluable by-products of grad school that I will cherish for the rest of my life.

Badminton played a significant role in my life outside of research during my time at the University of Minnesota, Twin Cities. I would like to extend my gratitude to my fellow players and friends—Sarthak Jain, Mishfad S.V., Arijit Mondal, Subramanya Sastry, Jashan Singh, Abhishek Kulkarni, EngHock Lee, Tuii Srisomboon, Samarth Shirahatti, Adarsh Kodandapani, Nguyễn Hoàng Vĩnh Thụy, Anthony Nguyễn Hồ, Andy Lin, Paul Chen, Anjie Jin, Jane Jien, Chen Wei Kent, Zoe Chen, Vivian To, Daniel Lu, Jake Ma, Dick Chng, Tan Yen Sheng, Wei Nee Long, Terry Li, and Kaleb Dean Wirtzfeld—for the enjoyable moments both on and off the court.

I am thankful to my Karate family that supported me throughout me some rough time during my Ph.D. years and acquainted me to the principles of Karate. I am especially grateful to my Senseis Trung Le and Madison Hawrysz, my cohort and friends, Julianna Kimmel, Jannah Oroceo, Jasmine Beldin, Jason Brisceaux, Lili Wang, Shashank Hegde and Guilherme Preis.

Finally, and most importantly, I am forever indebted to my parents, Saimal and Manju, my sister, Shivani, and, most especially, Zuag Paj and her family, for their unwavering love, support, and belief in me, even during moments when I faltered and struggled. They were my constant source of strength throughout this long and arduous journey. I recognize the numerous sacrifices to allow me to fully immerse myself in my research. I will strive to make up for the lost moments in the future.

Without the love and support of these incredible individuals, along with countless other family and friends, this work would not have been possible. I am deeply thankful to each and every one of you.

Dedication

ॐ भूर्भुवः स्वः तत्सवितुर्वरेण्यं |
भर्गो देवस्य धीमहि धियो यो नः प्रचोदयात् ||

O Divine mother, may your effulgence illuminate all realms (physical, mental, and spiritual) of our being. Please expel any darkness from our hearts and bestow upon us true knowledge.

I dedicate my thesis to

Mummy, Papa, Didi for their love and support, and
all my teachers and mentors for their encouragement and guidance.

Abstract

Driven by the need for resiliency, scalability, and plug-and-play operation, distributed decision-making is becoming increasingly vital. This thesis develops a comprehensive framework to address the challenges in distributed decision-making, focusing on distributed optimization and control of multi-agent networks. It combines theoretical insights, algorithmic tools, and experimental validation to enhance decision-making in networked dynamical systems (NDS) with applications aimed at advancing clean and renewable energy systems. The proposed solutions emphasize distributed consensus, distributed optimization under practical constraints, and resilience against malicious agents and natural disasters in modern power systems.

Achieving consensus in multi-agent systems is a cornerstone of distributed decision-making. The first part of this thesis makes a significant contribution in the development of distributed average consensus algorithms in multi-agent networks. By analyzing the geometry of the ratio consensus algorithm, this thesis introduces a finite-time distributed stopping criterion that guarantees convergence within any specified tolerance, regardless of the dimensionality of the state variables. The approach leverages the monotonicity of network state polytopes indexed by time. Additionally, the thesis presents a finite-time stopping criterion for networks with dynamic interconnection topologies, demonstrating that global maximum and minimum values remain strictly monotonic, even with dynamic links. The practicality of these algorithms is showcased through MATLAB simulations and experiments with Node.js-based agents. To address communication noise in the cyber-physical components of NDS, a resilient average consensus algorithm is proposed. Each agent updates its estimates using local information while incorporating weighted, noise-free initial values during iterations. The resilient average consensus algorithm has a geometric-rate of convergence under *noiseless* conditions and *almost-sure* consensus under *noisy* communication. Numerical experiments confirm its effectiveness under varying noise scenarios and parameters.

Part two of this thesis introduces a framework for distributed multi-agent optimization problems involving a common decision variable. A novel *optimize then agree* approach decouples optimization and consensus steps, ensuring disagreement between agents' estimates remains below a predefined tolerance; existing algorithms do not provide such a guarantee which is required in many engineering scenarios. For the class

of distributed optimization problems with local linear equality, inequality, and set constraints, we develop an algorithm that works over directed communication topologies and accrues all the benefits of the Alternating Direction Method of Multipliers approach. The algorithms synthesize *distributively*, communication overhead remains within a log factor of the lower bound, and guarantees strong convergence properties, achieving global geometric convergence rates for strongly convex and Lipschitz differentiable functions and global sublinear convergence rates for general convex functions. The efficacy of this framework is demonstrated through comparisons with state-of-the-art algorithms on distributed optimization and learning problems.

The last part of the thesis, focuses on developing methods for clean and renewable energy adoption in power systems. We develop a distributed controller for secondary control in microgrids with grid-forming inverter-based resources (GFM IBRs). The controller uses distributed optimization, enabling decentralized measurements and neighborhood information exchange to achieve voltage regulation and reactive power sharing. Additionally, a framework for distributed detection and isolation of maliciously behaving agents is proposed for resilient power apportioning between distributed energy resources (DERs). To address challenges posed by the absence of power grids in catastrophic events, this thesis introduces a net-load management engine (horizon of viability (HoV) engine) that ensures reliable power supply to critical infrastructure over a given time-horizon by generating cost-optimal portfolios of local generation sources and loads using mixed-integer convex programming. Controller-hardware-in-the-loop (CHIL) platforms validate the proposed secondary controller, the resilient power apportioning protocol, and the HoV engine across diverse DERs and loads, demonstrating the robustness of the developed methods.

Contents

Acknowledgements	i
Dedication	iv
Abstract	v
Contents	vii
List of Tables	x
List of Figures	xi
List of Symbols	xvi
List of Abbreviations	xvii
List of Definitions	xviii
1 Introduction	1
1.1 The Dual Imperative: Centralized Efficiency and Distributed Resilience	1
1.1.1 Cracks in the Armor of Centralization	1
1.1.2 A Paradigm Shift: The Promise of Distributed Systems	2
1.2 Optimal Operation and Agreement in Distributed NDS	3
1.3 Management and Control of Distributed Energy Resources	4
1.4 Software-hardware testbed for Emulating NDS	5
1.5 Contributions and Overview of the Thesis	5
1.5.1 Algorithms and Testbed for Distributed Decision-Making	5

1.5.2	Resilient and Clean Power Grid of the Future Enabled by Inference and Control	7
1.6	Publications	8
2	Controlled Agreement in Networked Dynamical Systems: Practical Average Consensus Over Networks	10
2.1	Gist of the Chapter	10
2.2	State-of-the-art on Consensus Algorithms	11
2.3	Research Questions	15
2.4	Contributions of the Chapter	16
2.4.1	Contributions Towards Research Focus 1:	16
2.4.2	Contributions Towards Research Focus 2:	17
2.5	Developed Solutions	18
2.5.1	Noise Resilient Distributed Average Consensus Over Directed Graphs	18
2.5.2	Distributed Finite-time Terminated Consensus Algorithm in Switching Topologies	49
2.5.3	Distributed Finite-time Approximate Consensus in Higher Dimensions	64
3	Resource Management in Networked Dynamical Systems: Distributed Optimization Over Networks	76
3.1	Gist of the Chapter	76
3.2	State-of-the-art on Distributed Optimization Methods	77
3.3	Research Questions	81
3.4	Contributions of the Chapter	84
3.4.1	Contributions Towards Research Focus 1:	84
3.4.2	Contributions Towards Research Focus 2:	85
3.5	Developed Solutions	87
3.5.1	GradConsensus: Linearly Convergent Algorithm for Reducing Disagreement in Multi-agent Optimization	87
3.5.2	DC-DistADMM: ADMM Algorithm for Constrained Optimization over Directed Graphs	112
4	Methods for Optimal and Resilient Operation of Power and Energy Systems	148
4.1	Gist of the Chapter	148

4.2	State-of-the-art on Power and Energy System Operation and Control . .	149
4.3	Research Questions	153
4.4	Contributions of the Chapter	154
4.4.1	Contributions Towards Research Focus 1:	154
4.4.2	Contributions Towards Research Focus 2:	155
4.4.3	Contributions Towards Research Focus 3:	155
4.5	Developed Solutions	156
4.5.1	Distributed Secondary Control for Microgrids with Grid-Forming Inverters	156
4.5.2	Resilient Power Apportioning in Microgrids with Online Malicious Behavior Detection	169
4.5.3	Horizon of Viability Optimization Engine for Sustained Power to Critical Infrastructure	182
5	Conclusion And Future Work	194
5.1	Concluding Remarks	194
5.2	Future Directions of Research	198
	Bibliography	201
	Appendix A.	229
A.1	Software-Hardware Testbed for Emulating Real-life Network Dynamical Systems	229
A.1.1	An Instantiation for Decision-Making in Power Systems	232
A.1.2	Example Commands for the Power Systems Application	235

List of Tables

2.1	Example Sequences for NR-PushSum	44
2.2	Network Topology and Range ($R_1(k, t)$) of Node 1 Starting at Instant k for Iterations $t = 1, 2, 3, 4, 5$	52
3.1	Algorithm Parameters of GradConsensus	106
3.2	Algorithm Parameters of DC-DistADMM	141
4.1	1-Phase GFM IBR Under Study	164
4.2	Ratings of Various Sources and Loads in Central Core of the Microgrid .	169
4.3	Ratings of Various Sources and Loads in External Network of the Microgrid	170
4.4	Ratings of Various Generation Sources, Loads, and Line Parameters of the Microgrid Under Study	186

List of Figures

2.1	Performance of the PushSum algorithm under the perfect communication (a) and with communication noise (b).	21
2.2	NR-PushSum algorithm update: agent i receives noisy information from neighbors j and l	22
2.3	NR-PushSum algorithm update: agent i updates its estimates using the received information, its current estimate, and its initial value.	23
2.4	Performance of the NR-PushSum algorithm under the noiseless case: (a) the sequence of states $x_i(k)$, (b) $y_i(k)$ sequences, (c) ratio states of the 10 agents achieve consensus, (d) geometric decay of the consensus error.	43
2.5	Performance of the NR-PushSum algorithm under noisy communication with different choices of algorithm parameters: (a) Row (i) of Table 2.1, (b) Row (ii) of Table 2.1, (c) Consensus error for parameters in Row (i) of Table 2.1, (d) Consensus error for parameters in Row (ii) of Table 2.1.	44
2.6	Performance of the NR-PushSum algorithm under the variation of the magnitude of additive communication noise: (a) noise realized via $\mathcal{U}(-5, 5)$, (b) noise realized via $\mathcal{U}(-10, 10)$. In both cases, the state estimates under NR-PushSum with appropriate algorithmic parameters converge to the initial average, (c) Consensus error under the noise realization via $\mathcal{U}(-5, 5)$, (d) Consensus error under the noise realization via $\mathcal{U}(-10, 10)$	46
2.7	Performance of the NR-PushSum algorithm under scenarios with the noise realizations violating the upper bound utilized in the algorithm's parameters. Figs. 2.7(a) and 2.7(b) demonstrate the robustness of the proposed NR-PushSum algorithm under extreme violation scenarios, Figs. 2.7(c) and 2.7(d) present the consensus error decay during the iterations of the NR-PushSum algorithm.	47

2.8	Performance comparison of the NR-PushSum algorithm with existing algorithms in the literature: (a) performance under bounded noise with zero mean and (b) performance under bounded communication noise with non-zero mean.	48
2.9	Both $\mathcal{G}_1$ and $\mathcal{G}_2$ with $\mathcal{D}_{max} = 3$	51
2.10	Topology shown as a bidirectional graph at time instants $k = 109$, $k = 239$, and $k = 275$ respectively.	61
2.11	Finite-time termination of ratio consensus on a 1000-node dynamic network in 6000 iterations with $\rho = 0.01$. Each curve (without markers) in the plot is the time evolution of the ratio state of 1000 nodes. MXP/MNP (when using n' as sampling) are monotonic and correctly converge. The MXP/MNP curves are shifted left to coincide with global maximum and minimum, as mentioned in Remark 9.	61
2.12	Node ratio state trajectories for the ratio consensus of the 6-node network shown in Figure 2.9. The MXP/MNP for sampling length of both $\mathcal{D}_{max} = 3$ and $n' = 5$ are plotted. MXP when $\mathcal{D}_{max}$ is used as sampling is non-monotonic. MXP/MNP (when using n' as sampling) are monotonic and correctly converge. The MXP/MNP curves are shifted left to coincide with the global maximum and minimum.	63
2.13	Node numerator and denominator state trajectories for the ratio consensus of the 6-node network shown in Figure 2.9. Both states for each node are bounded above and below.	63
2.14	A communication network represented by a 25 node digraph.	74
2.15	2-norm of 10-dimensional ratio states of all the nodes (25) in the network.	74
2.16	Radius $\overline{R}_i(l)$ at each node.	75
2.17	2-dimensional projection of norm balls for node 1 with changing l	75
3.1	1000-node test network topology	105
3.2	Objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ against the number of gradient computations.	107
3.3	Communication steps required per iteration of the algorithm.	109
3.4	Objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ against total communication steps.	110
3.5	Comparison of the objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ with respect to total run-time.	111

3.6	(a) Comparison of consensus constraint residuals against total iterations (including communication steps) between DC-DistADMM and constrained optimization algorithms (b) Comparison of equality constraint residuals against total iterations between DC-DistADMM and constrained optimization algorithms. $\eta^k = 0.75^k$ gives the best performance for the DC-DistADMM algorithm in terms of the equality constraint residual. The DC-DistADMM algorithm is second best in terms of the consensus constraints.	143
3.7	(a) Comparison of solution residuals against total iterations between DC-DistADMM and constrained optimization algorithms (b) Comparison of solution residuals against CPU time between DC-DistADMM and constrained optimization algorithms. DC-DistADMM algorithm with $\eta^k = 0.75^k$ has the best decrease in the solution residual.	143
3.8	(a) Comparison of solution residuals against total iterations between DC-DistADMM and unconstrained optimization algorithms (b) Comparison of solution residuals against CPU time between DC-DistADMM and unconstrained optimization algorithms. DC-DistADMM algorithm performs the best in terms of the CPU time while having acceptable performance with respect to total iterations. DC-DistADMM has the best decrease in solution residual with respect to algorithm iterations.	145
3.9	(a) Comparison of solution residuals against total iterations between DC-DistADMM and algorithms utilizing multiple communication steps (b) Comparison of solution residuals against CPU time between DC-DistADMM and algorithms utilizing multiple communication steps. $\eta_k = 0.75^k$ has the best decrease in solution residual with respect to total iterations. DC-DistADMM with a constant tolerance performs the best with respect to CPU time.	146
3.10	Comparison of the number of communication iterations performed by DC-DistADMM algorithm for the three choices of η_k and the FDGD and nearDGD algorithms. DC-DistADMM with $\eta_k = 0.01$ and $\eta_k = 1/k^{2.1}$ has lesser communication cost compared to the FDGD and nearDGD algorithms while the FDGD has better communication complexity than DC-DistADMM with $\eta_k = 0.75^k$ as the number of iterations increases. . . .	146

4.1	(a) Distributed <i>secondary control</i> architecture with <i>primary control</i> for multiple GFM IBRs connected in parallel, (b) $P \sim f$, $Q \sim V$ droop-based <i>primary control</i> with <i>secondary control</i> for parallel GFM IBRs i and j with equal ratings, connected to PCC via reactive lines with $X_j > X_i$	157
4.2	The laboratory-based system-in-the-loop (SIL) and controller hardware-in-the-loop (CHIL) experimental hardware setup.	165
4.3	Performance results of the developed distributed secondary controller. (a) Under CASE-1 : equal-rated reactive power sharing, (b) Under CASE-2 : tight voltage regulation, (c) Under CASE-3 : mixed objective DERs 1-7 equal-rated reactive power sharing and DERs 8-10 tight voltage regulation.	168
4.4	Figure of (a) power network of the microgrid under study, (b) communication topology used in the power apportioning.	171
4.5	Resilient distributed power apportioning scheme with malicious DER m . Honest DER i detects the presence of m	176
4.6	Laboratory CHIL setup.	178
4.7	Performance of the distributed power apportioning scheme with malicious hsDu detection under ATTACK-1	179
4.8	Performance of the distributed power apportioning scheme with malicious hsDu detection under and- ATTACK-2 (e)-(h).	181
4.9	The microgrid test system contains 1 critical infrastructure (CI) unit and 3 community building (CB) units. The CI unit contains 2 GFL battery inverters, 2 diesel generators, 1 PV inverter system, 1 critical load (CL), and 30 deferrable loads (DLs). Similarly, CB-1, CB-2, and CB-3 contain 1 GFM battery inverter each, 1 PV inverter system each, 1 CL each, and 25, 20, and 30 DLs, respectively.	187
4.10	Schematic of the CHIL-based demonstration of the HoV optimization engine.	188
4.11	Total remaining energy (in kWh) after 2 hours of the CHIL experiment of the proposed HoV engine and existing NLM engine on the test system in Figure 4.9.	189
4.12	Remaining energy (in kWh) in each generating resource (2 GFL batteries and 2 diesel generators) in the CI unit after 2 hours of the CHIL experiment of the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.	190

4.13	Total cost-weighted energy after 2 hours of the CHIL experiment of the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.	190
4.14	Total energy delivered after 2 hours of the CHIL experiment of the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.	191
4.15	Active power output and SOC of the GFL battery-1 in the CI unit during 2 hours of the CHIL experiment with the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.	191
4.16	Active power output and SOC of the GFL battery-2 in the CI unit during 2 hours of the CHIL experiment with the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.	192
4.17	Ratio of the CWE metric value resulting from HoV engine and NLM engine during 2 hours of CHIL experiment on the test system of Figure 4.9.	192
4.18	Ratio of the TRE metric value resulting from HoV engine and NLM engine during 2 hours of CHIL experiment on the test system of Figure 4.9.	193
A.1	Block diagram showing the integration of the five major components Software-Hardware testbed.	230
A.2	Software-Hardware testbed for power system applications.	233
A.3	Backend server (or Demo Node)-console log at the startup.	236
A.4	Distributed agent running optimization-console log at the startup.	237
A.5	Data storage controller-console log at the startup.	237

List of Symbols

$[\mathbf{N}]$	- The set $\{1, 2, \dots, n\}$
$\mathcal{G}(\mathcal{V}, \mathcal{E})$	- Graph $\mathcal{G}$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$
$\mathcal{D}$	- Diameter of graph $\mathcal{G}$
$\mathcal{N}_i^-$	- In-Neighborhood of agent i
$\mathcal{N}_i^+$	- Out-Neighborhood of agent i
$\mathbf{P}$	- Graph weight matrix
$A_{[ij]}$ or a_{ij}	- Element of matrix A in the i^{th} row and j^{th} column
k	- Iteration counter of the corresponding algorithm
$\mathcal{L}_\gamma$	- Augmented Lagrangian of the optimization problem with parameter γ
$ S $	- Cardinality of set a given set S
$\ x\ $	- Norm of a given vector or matrix x

List of Abbreviations

NDS	- Networked Dynamical Systems
MAS	- Multi-agent Systems
AC	- Alternating Current
MG	- Microgrid
DL	- Deferrable Load
DER	- Distributed Energy Resources
RES	- Renewable Energy Sources
PV	- Photo Voltaic
IBR	- Inverter Based Resources
GFL	- Grid-following
GFM	- Grid-forming
CI	- Critical Infrastructure
DRTS	- Digital Real-time Simulation
RTS	- Real-time Simulator
CHIL	- Controller Hardware-in-the-loop
HoV	- Horizon of Viability

List of Definitions

Here, we present the definitions for the notions of graph theory and linear algebra which are essential for the development of this thesis. Detailed descriptions of graph theory, linear algebra, and convex analysis are available in [1]–[3] respectively.

Definition 1. (*Directed and Undirected Graph*) A directed graph (also digraph) $\mathcal{G}$ is a pair $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is a set of vertices or nodes and $\mathcal{E}$ is a set of edges, which are ordered subsets of two distinct elements of $\mathcal{V}$. If an edge from $j \in \mathcal{V}$ to $i \in \mathcal{V}$ exists then it is denoted as $(i, j) \in \mathcal{E}$.

An undirected graph $\mathcal{G}$ is a pair $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is a set of vertices or nodes and $\mathcal{E}$ is a set of edges such that for every pair of distinct nodes $i \in \mathcal{V}$ and $j \in \mathcal{V}$, if $(i, j) \in \mathcal{E}$ then $(j, i) \in \mathcal{E}$.

Definition 2. (*Path*) In a directed graph, a path from node i to j exists if there is a sequence of distinct directed edges of $\mathcal{G}$ of the form $(k_1, i), (k_2, k_1), \dots, (j, k_m)$.

Definition 3. (*Diameter of a Graph*) The diameter $\mathcal{D}$ of a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is the longest shortest directed path between any two nodes in the graph $\mathcal{G}$.

Definition 4. (*Strongly Connected Graph*) A directed graph is strongly connected if it has a path between every pair of distinct nodes $i \in \mathcal{V}$ and $j \in \mathcal{V}$.

Definition 5. (*In-Neighborhood*) The set of in-neighbors of node $i \in \mathcal{V}$ is called the in-neighborhood of node i and is denoted by $\mathcal{N}_i^- = \{j \mid (i, j) \in \mathcal{E}\}$.

Definition 6. (*Out-Neighborhood*) The set of out-neighbors of node $i \in \mathcal{V}$ is called the out-neighborhood of node i and is denoted by $\mathcal{N}_i^+ = \{j \mid (j, i) \in \mathcal{E}\}$.

Definition 7. (*Union of digraphs*) Given a collection of digraphs, $\{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_m\}$ (for some $m \geq 1$), of the form $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$, $k = 1, 2, \dots, m$, the union of digraphs is defined as $\mathcal{G}(k)_{1,2,\dots,m} = (\mathcal{V}, \cup_{k=1}^m \mathcal{E}(k))$.

Definition 8. (*In-neighborhood of node i at instant k*) The set of in-neighbors of node $i \in \mathcal{V}$ at instant k is denoted by $\mathcal{N}_i^-(k) = \{j : (i, j) \in \mathcal{E}(k)\}$.

Definition 9. (*Out-neighborhood of node i at instant k*) The set of out-neighbors of node $i \in \mathcal{V}$ at instant k is denoted by $\mathcal{N}_i^+(k) = \{j : (j, i) \in \mathcal{E}(k)\}$.

Definition 10. (*Out-degree of node i at instant k*) The out-degree of a node, i , at any instant, k , is equal to the cardinality of the set of out-neighbors of i , $|\mathcal{N}_i^+(k)|$.

Definition 11. (*Column Stochastic Matrix*) A matrix $\mathbf{P} = [p_{ij}] \in \mathbb{R}^{n \times n}$ is called column stochastic if $0 \leq p_{ij} \leq 1$ and $\sum_{i=1}^n p_{ij} = 1$ for all $1 \leq j \leq n$.

Definition 12. (*Row Stochastic Matrix*) A matrix $\mathbf{P} = [p_{ij}] \in \mathbb{R}^{n \times n}$ is called row stochastic if $0 \leq p_{ij} \leq 1$ and $\sum_{j=1}^n p_{ij} = 1$ for all $1 \leq i \leq n$.

Definition 13. (*Doubly Stochastic Matrix*) A matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$ is called doubly stochastic if it is both column and row stochastic.

Definition 14. (*Irreducible Matrix*) A $n \times n$ matrix $\mathbf{P}$ is said to be irreducible if for any $i, j \in \{1, \dots, n\}$, there exists $m > 0$ such that $(\mathbf{P}^m)_{[ij]} > 0$.

Definition 15. (*Primitive Matrix*) A non-negative matrix $\mathbf{P}$ is primitive if it is irreducible and has only one eigenvalue of maximum modulus.

Definition 16. (*Cardinality of a set*) The cardinality of a given set S denoted by $|S|$ is the number of elements of the set S .

Definition 17. (*Diameter of a set*) For a norm $\|\cdot\|$ and a set $K \subset \mathbb{R}^d$ define the diameter of K with respect to the norm $\|\cdot\|$, $\text{diam}_{\|\cdot\|}(K) := \sup_{x, y \in K} \|x - y\|$.

Definition 18. (*Affine hull of a set*) The affine hull $\text{Aff}(S)$ of a set S is the set of all affine combinations of elements of S ,

$$\text{Aff}(S) = \left\{ \sum_{i=1}^k \theta_i x_i \mid k > 0, x_i \in S, \theta_i \in \mathbb{R}, \sum_{i=1}^k \theta_i = 1 \right\}.$$

Definition 19. (*Relative interior of a set*) Let $K \subset \mathbb{R}^d$. A point $x \in K$ is in the relative interior of K , if K contains the intersection of a small enough ball centered at x with the $\text{Aff}(K)$, that is, there exists $r > 0$ such that

$$B_r(x) \cap \text{Aff}(K) := \{y \mid y \in \text{Aff}(K), \|y - x\| \leq r\} \subset K.$$

The set of all relative interior points of K is called its relative interior denoted by $\text{relint}(K)$.

Definition 20. (Convex Set) A set C is convex if the line segment between any two points in C lies in C , i.e.

$$\text{for all } x, y \in C, \text{ for all } \theta \in [0, 1], \theta x + (1 - \theta)y \in C.$$

Definition 21. (Convex Hull) For a set $U \subseteq \mathbb{R}^d$, the convex hull of U is the smallest convex set containing U ,

$$\text{co}(U) = \bigcap_{\{F \text{ convex} : U \subseteq F\}} F.$$

Definition 22. (Convex Function and its Epigraph) A function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is convex if $\text{dom}(f)$ is a convex set and if

$$\text{for all } x, y \in \text{dom}(f), \text{ for all } \theta \in [0, 1], f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y),$$

where, $\text{dom}(f)$ is the set of inputs accepted by the function f . The epigraph of a function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is the set of points $\text{epi}(f) = \{(x, t) | x \in \text{dom}(f), t \geq f(x)\}$.

Definition 23. (Lipschitz Differentiability) A differentiable function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is called Lipschitz differentiable with constant $L > 0$, if the following inequality holds:

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

Definition 24. (Strongly Convex Function) A differentiable function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is called strongly convex with parameter σ , if there exists $\sigma > 0$ such that for all x, y in the domain of f :

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \sigma\|x - y\|^2, \forall x, y \in \text{dom}(f).$$

Definition 25. (Restricted Strongly Convex Function [4]) Given, $\tilde{x} \in \mathbb{R}^d, C \subseteq \mathbb{R}^d$, a differentiable convex function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is called restricted strongly convex with respect to $\tilde{x}$ on C with parameter $\sigma > 0$, if the following inequality holds:

$$\langle \nabla f(x) - \nabla f(\tilde{x}), x - \tilde{x} \rangle \geq \sigma\|x - \tilde{x}\|^2, \forall x \in C.$$

Definition 26. (*Critical and Deferrable Load*) We call a load a critical load (CL) if it does not have any flexibility associated with its control (on/off) and its power requirement must be met whenever requested during the time of operation. We call a load deferrable load (DL) if the power delivery to the load can be deferred. The j^{th} DL being on/off implies that the power delivered to the j^{th} DL is equal to nonzero/zero power.

Definition 27. (*State of Charge and State of Energy*) The state of charge (SOC) of the j^{th} battery, denoted by s_{b_j} , is the level of charge of an electric battery relative to its capacity. Similarly, the state of energy (SOE) of the j^{th} stand-alone diesel generator, denoted by s_{g_j} , is the amount of energy equivalent fuel remaining relative to its reserve capacity.

Definition 28. (*Net-Load Resource*) A collection of resources including distributed energy resources like battery inverter systems, stand-alone diesel generators, renewable energy sources, and deferrable loads is called a net-load resource.

Chapter 1

Introduction

1.1 The Dual Imperative: Centralized Efficiency and Distributed Resilience

Centralized systems have long served as the bedrock of modern infrastructure, celebrated for their remarkable efficiency, uniform standards, and streamlined control [5]–[8]. This centralized approach has revolutionized industries, driving unprecedented gains in productivity and elevating living standards across the globe. Nowhere is this efficiency more evident than in the centralized power grid, which boasts an extraordinary 99.95% reliability under normal conditions in the United States. Yet, this optimization comes with a significant trade-off: vulnerability to disruption.

1.1.1 Cracks in the Armor of Centralization

The fragility of centralized systems becomes glaringly apparent in times of crisis. Malicious cyberattacks, such as the devastating breaches of the Ukrainian power grid in 2015 and 2016 [9], left over 225,000 people without power, starkly illustrating how centralized systems can become prime targets for external threats and are susceptible to single points of failure. Compounding this vulnerability are natural disasters fueled by accelerating climate change—floods, hurricanes, and earthquakes—that have triggered catastrophic events. The 2021 Texas winter storm [10] and ongoing wildfires in California [11] left millions in the dark, exposing the cascading failures inherent in centralized architectures. These incidents underscore a pressing reality: when a single point of failure is compromised, the consequences ripple through the entire system with alarming

speed and scale.

1.1.2 A Paradigm Shift: The Promise of Distributed Systems

As vulnerabilities in centralized models grow more apparent, distributed systems are emerging as a transformative alternative—particularly within the power sector. Innovations in renewable energy and battery storage have accelerated the adoption of distributed energy resources (DERs), which decentralize energy generation and bolster system resilience. Unlike centralized grids, distributed systems can operate autonomously in “islanded” modes, sustaining local energy delivery even when the broader network fails. This localized control fosters quicker recovery and adaptive responses during emergencies, reducing dependency on centralized nodes.

Distributed systems also align seamlessly with global sustainability imperatives. By leveraging renewable energy sources, they contribute to reduced carbon emissions while ensuring stability and reliability. Grid-forming (GFM) DERs autonomously regulate network voltage and frequency, offering robust performance even under heterogeneous conditions. Meanwhile, grid-following (GFL) DERs dynamically complement centralized systems by responding to energy demands. Together, these advancements reimagine distributed architectures as not merely an alternative, but as an indispensable counterpart to traditional centralized models.

As we progress toward a future shaped by increasingly complex and interconnected infrastructures, a pivotal question emerges: *Can networked dynamical systems (NDS) achieve the levels of efficiency and optimization traditionally associated with centralized operations, while simultaneously leveraging distributed architectures to ensure the resiliency needed to withstand crises and foster a sustainable future?*

This conundrum represents the frontier of innovation—where efficiency and resilience must converge to meet the challenges of a rapidly evolving world. My thesis work provides answers to aspects of the above issue via a mathematical formalism based on optimization and distributed agreement. The resulting analytical and synthesis tools are leveraged toward distributed decision-making for the grid of the future, enabled by DERs, which is corroborated by at-scale-experimental demonstration. The highlights of my thesis are:

1.2 Optimal Operation and Agreement in Distributed NDS

Modern, data-rich networked dynamical systems (NDS) are driving innovation and competition in various fields enabling higher levels of autonomy. The demand for innovation necessitates research to address complexities in NDS dynamics and distributed components, paving the way for robust decision-making mechanisms. In centralized systems, decision-making involves a central entity with complete information, responsible for achieving specific objectives. Enabled by low-cost sensors, distributed computation, and storage, multi-agent systems (MAS) are now integral to modern NDS across diverse applications [12]–[22]. MAS necessitate distributed decision-making structures where agents collaboratively determine system-wide objectives [21]–[23].

Distributed decision-making can be formulated in a setup, where multiple agents (computing units) work in parallel, sharing intermediate results to compute a final outcome. This setup is essential for scenarios involving spatially dispersed input data, such as IoT sensor networks [24], or when rapid computation is required, such as distributed machine learning (ML) problems [25]. Effective control of distributed environments further requires global insights without complete system knowledge [26]–[28]. Early works [14], [29] formalized distributed optimization problem where agents iteratively communicate and update their solutions to achieve consensus—a key requirement for distributed protocols. *Consensus*, the agreement among agents on a specific value, has been widely studied in systems and control theory [30], [31]. A notable form, *average consensus*, involves agents agreeing on the average of their initial values. Consensus protocols, which define interaction rules for information exchange, are fundamental for distributed computation. Similarly, *distributed optimization* arises in applications such as sensor network inference [32], multi-agent coordination [33], and distributed ML [25]. These problems involve minimizing a shared cost function f across a network of n agents, formulated as:

$$\begin{aligned} & \underset{x \in \mathbb{R}^p}{\text{minimize}} \quad f(x) = \sum_{i=1}^n f_i(x), \\ & \text{subject to} \quad x \in \mathcal{X} \end{aligned} \tag{1.1}$$

where, $x \in \mathbb{R}^p$ is a common decision vector, and $\mathcal{X}$ is the decision constraint set. This thesis develops a framework for high-confidence distributed decision-making in NDS, focusing on:

- establishing theoretical foundations for distributed decision-making, and,
- designing scalable, robust algorithms for distributed consensus and optimization, capable of operating under constraints such as communication noise and dynamic interaction links, while ensuring finite-time feasibility and performance.

1.3 Management and Control of Distributed Energy Resources

The traditional power grid is generally reliable, delivering electricity through large, centralized, fossil-fuel-based generation systems. However, natural disasters and security breaches can result in significant blackouts, causing devastating and costly disruptions. For instance, between 2000 and 2021, approximately 83% of major outages (over 1,500 incidents) in North America were linked to weather-related events such as winter storms and tropical cyclones. These events increased sharply—by about 78%—between 2011 and 2021 compared to the preceding decade [34]. A stark example is Puerto Rico, which suffered an island-wide blackout following Hurricane Maria in 2017, leaving many areas without electricity for nearly 11 months [35].

To address vulnerabilities of the centralized power grid, there is a growing need for more distributed and resilient methods of power generation, and distribution. Increasingly, utilities and companies are adopting “microgrids”—self-sufficient power systems capable of operating independently from centralized power grids. A typical microgrid setup includes a variety of loads, such as residential buildings, battery energy storage systems, and renewable energy sources (RES). Microgrids have long been recognized as vital for ensuring power resilience during outages. The development of power electronics and advanced digital control systems has made inverter-based resources (IBRs) integral to modern microgrids. These IBRs enable the seamless integration of renewable DERs such as solar panels, wind turbines, and energy storage systems like fuel cells and supercapacitors. Depending on their control schemes and associated DERs, IBRs can operate in either grid-following (GFL) mode or grid-forming (GFM) mode. GFL IBRs, typically paired with renewable DERs, deliver maximum available power to the grid. Conversely, GFM IBRs, often used with battery storage, ensure stable voltage and frequency, particularly during islanded mode operation [36].

Microgrids typically follow a hierarchical control structure based on the ANSI/ISA-95 international standard [37]. This structure consists of four levels (zero $\rightarrow$ primary

→ secondary → tertiary level)), with each level providing supervisory control to the one below and receiving oversight from the one above. To ensure stability, bandwidth requirements decrease at higher levels, minimizing the impact of commands and reference signals between layers. This thesis focuses on designing secondary-level control schemes, which compensate for voltage and frequency deviations introduced by primary-level control [36]. Aggregating multiple IBRs to function as a cohesive unit can enable distributed IBRs to supply diverse loads. An area of ongoing research is improving microgrid capabilities for disaster response and recovery. Natural disasters severely impact the power grid, causing prolonged outages and damaging critical infrastructure (CI). Power disruptions to CI pose significant threats to economic stability, public health, and safety. In such scenarios, microgrids provide a reliable and adaptable power source, supporting emergency response operations and facilitating recovery efforts.

1.4 Software-hardware testbed for Emulating NDS

Practical implementation of distributed decision-making algorithms in NDS requires validation through experimental prototypes. This thesis develops a Software-hardware testbed to test and validate distributed decision-making methods utilizing a *Controller Hardware-in-the-Loop* (CHIL) based framework. CHIL is a cost-effective and low-risk approach for testing algorithms in realistic conditions. CHIL involves a Digital Real-Time Simulation (DRTS) setup that integrates a microcontroller-based hardware with a physical dynamical system or a real-time simulator (RTS) (emulating the dynamical system), creating a testbed that emulates real-world NDS with physical components, communication capabilities, and software layers. This setup enables rigorous evaluation of the developed distributed methods across diverse applications, facilitating their deployment in the field.

1.5 Contributions and Overview of the Thesis

1.5.1 Algorithms and Testbed for Distributed Decision-Making

The theoretical and algorithmic contributions of this dissertation focus on the development of robust, distributively synthesizable optimization and learning algorithms for NDS. In this context, a group of intelligent subsystems—each with its local cost function—collaborate to minimize a global cost function, defined as the aggregation of all local

objectives, while communicating over a constrained connectivity graph. Within this distributed framework, I have developed novel algorithms that address various practical scenarios:

- **Chapter 2** of the thesis introduces a framework for **controlled agreement** in NDS. A new class of algorithms, termed *convex decreasing* algorithms, is proposed, characterized by their monotonic decay properties. For higher-dimensional decision variables, we design a finite-time distributed stopping criterion to determine convergence to a predefined tolerance. This approach is highly efficient, featuring minimal communication overhead and remaining independent of the dimensionality of the decision variables.

The finite-time stopping criterion is further extended to handle time-varying interconnection topologies between agents. For practical scenarios with communication noise, we develop a distributed algorithm that reliably achieves average consensus on the initial state values under bounded noise. This algorithm combines noise-free local decisions of the agents with noisy updates from neighbors, effectively recovering the true global average. Notably, it is the *first* algorithm to achieve such *robust noise resilience*.

- **Chapter 3** of the thesis focuses on a framework for **resource management** in NDS. Specifically, we introduce an every-iterate-feasible distributed method that decouples the computations of individual subsystems from the information-mixing interactions with others. Building on this novel framework, we propose a distributed algorithm, termed **GradConsensus**, where each subsystem computes a local optimization solution and updates it using the finite-time terminated practical agreement protocol developed in Chapter 2.

The **GradConsensus** algorithm is highly versatile, with applications in estimation and decision-making across domains such as power systems and social networks.

Chapter 3 introduces an effective algorithm for optimal distributed decision-making under operational and systemic constraints. This novel approach is based on the Alternating Direction Method of Multipliers (ADMM) and is specifically designed to function effectively even in *unidirectional* communication networks. Departing from traditional optimization techniques, the proposed algorithm incorporates a lower-complexity consensus algorithm (detailed in Chapter 2) and adaptively controls the disagreement between decision variables across all agents.

These innovations enable the algorithm to achieve optimal non-asymptotic computational rates, representing the *first* instance in the literature to attain such performance guarantees.

Deploying distributed algorithms directly in real-life NDS, such as power grids or transportation systems, poses risks to system performance and stability. To mitigate these risks and enable seamless testing and deployment, this dissertation introduces an experimental testbed that integrates physical hardware components, real-time emulated simulators, and a software engine executing the proposed optimization and learning algorithms.

This testbed empowers users—ranging from graduate students in research labs to engineers at utility companies—to rigorously validate their methods before real-world implementation. The developed open-source software package, written in Python and Node.js, is freely available on GitHub. It can be readily deployed on microcontrollers or Raspberry Pi devices to interface with physical hardware in NDS environments. More details about the experimental testbed are provided in Appendix A.1.

1.5.2 Resilient and Clean Power Grid of the Future Enabled by Inference and Control

Another significant contribution of this thesis is to facilitate the transition of the existing power grid to a modern, clean energy system while maintaining the reliability of energy delivery. This thesis emphasizes the design of innovative schemes for the aggregation and optimization of Distributed Energy Resources (DERs) to achieve this goal.

- **Chapter 4** introduces a framework for the *optimal and resilient* operation of inverter-based resources in power and energy systems. This chapter establishes a cooperative setup for aggregating Renewable Energy Sources (RES) owned by individuals and commercial entities. A distributed energy management algorithm is developed to optimize the amount of local energy support provided to the power grid while ensuring stability and regulatory compliance within the physical network of DERs. To address potential threats from malicious behavior among participating DERs, a resilient aggregation method is proposed. This method apportions power among DERs to provide ancillary support to critical infrastructure. Additionally, the framework includes a distributed malicious behavior detection

algorithm, enabling DERs to identify irregular or malign behavior from neighboring DERs effectively.

In another thrust, this dissertation focuses on developing resilient methods for managing and controlling DERs in power systems. The increasing frequency of power blackouts caused by natural disasters has highlighted the unreliable availability of power grids and the severe consequences of disruptions to critical infrastructure, such as hospitals. To address these challenges, an intelligent optimization scheme is developed using mixed-integer convex optimization techniques. This scheme provides energy dispatch decisions for RES and energy storage systems to ensure a sustained and reliable energy supply during grid outages. By carefully balancing the conservation of energy storage for future use with the utilization of available renewable generation, the algorithm ensures the long-term viability of critical infrastructure even under adverse conditions.

1.6 Publications

The following publications came out of this thesis work:

Journal Articles:

- [1] V. Khatana and M. V. Salapaka, “DC-DistADMM: ADMM Algorithm for Constrained Optimization Over Directed Graphs,” in *IEEE Trans. on Automatic Control*, vol. 68, no. 9, pp. 5365-5380, Sept. 2023, doi: 10.1109/TAC.2022.3221856.
- [2] G. Saraswat, V. Khatana, S. Patel and M. V. Salapaka, “Distributed Finite-Time Termination for Consensus Algorithm in Switching Topologies,” in *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 1, pp. 489-499, 1 Jan.-Feb. 2023, doi: 10.1109/TNSE.2022.3216286.
- [3] V. Khatana and M. V. Salapaka, “Noise Resilient Distributed Average Consensus Over Directed Graphs,” in *IEEE Transactions on Signal and Information Processing over Networks*, vol. 9, pp. 770-785, 2023, doi: 10.1109/TSIPN.2023.3324583.
- [4] V. Khatana, G. Saraswat, S. Patel and M. V. Salapaka, “GradConsensus: Linearly Convergent Algorithm for Reducing Disagreement in Multi-Agent Optimization,” in *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 1, pp. 1251-1264, Jan.-Feb. 2024, doi: 10.1109/TNSE.2023.3321757.

- [5] J. Melbourne, G. Saraswat, V. Khatana, S. Patel and M. V. Salapaka, “Convex Decreasing Algorithms: Distributed Synthesis and Finite-Time Termination in Higher Dimension,” in *IEEE Transactions on Automatic Control*, doi: 10.1109/TAC.2023.3344721.

Conference Articles:

- [1] V. Khatana, G. Saraswat, S. Patel and M. V. Salapaka, “Gradient-Consensus Method for Distributed Optimization in Directed Multi-Agent Networks,” 2020 American Control Conference (ACC), Denver, CO, USA, 2020, pp. 4689-4694, doi: 10.23919/ACC45564.2020.9147544.
- [2] V. Khatana and M. V. Salapaka, “D-DistADMM: A $O(1/k)$ Distributed ADMM for Distributed Optimization in Directed Graph Topologies,” 2020 59th IEEE Conference on Decision and Control (CDC), Jeju, Korea (South), 2020, pp. 2992-2997, doi: 10.1109/CDC42340.2020.9304086.
- [3] J. Melbourne, G. Saraswat, V. Khatana, S. Patel, M. V. Salapaka, “On the Geometry of Consensus Algorithms with Application to Distributed Termination in Higher Dimension,” *IFAC-PapersOnLine*, Volume 53, Issue 2, 2020, Pages 2951-2956, ISSN 2405-8963, <https://doi.org/10.1016/j.ifacol.2020.12.971>.
- [4] V. Khatana, S. Chakraborty, and M. V. Salapaka, “A Plug and Play Distributed Secondary Controller for Microgrids With Grid-forming Inverters,” in 2024 Annual Conference of the IEEE Industrial Electronics Society (IECON), 2024.
- [5] V. Khatana, S. Chakraborty, G. Saraswat, P. Sourav, and M. V. Salapaka, “A Distributed Malicious Agent Detection Scheme for Resilient Power Apportioning in Microgrids,” in 2024 Annual Conference of the IEEE Industrial Electronics Society (IECON), 2024.
- [6] V. Khatana, S. Chakraborty, G. Saraswat, B. Lindstrom and M. V. Salapaka, “Evaluation of Horizon of Viability Optimization Engine for Sustained Power to Critical Infrastructure,” 2022 IEEE Power and Energy Conference at Illinois (PECI), Champaign, IL, USA, 2022, pp. 1-8, doi: 10.1109/PECI54197.2022.9744038.

Chapter 2

Controlled Agreement in Networked Dynamical Systems: Practical Average Consensus Over Networks

2.1 Gist of the Chapter

Consensus protocols are critical in distributed MAS, requiring agents to agree on a shared value. One example is *average consensus*, where agents aim to agree on the average of their initial values. These protocols are designed using graph theory, where agents are nodes, and their interactions are modeled as edges in a graph. The challenge lies in achieving consensus under various conditions such as communication noise, dynamic topologies, or higher-dimensional states. Many existing protocols, such as the PushSum algorithm, struggle with noisy environments and dynamic topologies, making it difficult to ensure reliable consensus. Additionally, real-time decision-making requires the ability to detect convergence within a finite time, particularly in dynamic communication settings.

Traditional consensus protocols like PushSum work well under ideal conditions but fail under noise or dynamic topologies. This chapter proposes solutions for achieving consensus under these conditions, including the **NR-PushSum** algorithm and methods for determining approximate consensus in dynamic settings. This chapter also introduces

novel geometric insights and distributed stopping criteria, which are vital for practical applications in resource allocation, sensor networks, and autonomous systems. The algorithms ensure efficient, real-time consensus while minimizing communication costs. The research work presented in this chapter is published in the articles [38]–[41].

2.2 State-of-the-art on Consensus Algorithms

Distributed protocols require every agent to agree on the final decision or state value. When agents within a network unanimously agree on a particular quantity of interest, they are said to have reached a consensus. The challenge of achieving consensus in multi-agent systems has been explored in systems and control theory, with roots tracing back to the seminal works [22], [29], [30]. A notable instance of consensus among agents occurs when the agreed-upon value represents the average of all initial state values, termed average consensus. A formalized rule governing the interaction between agents to achieve consensus is termed a consensus protocol. A consensus protocol models the multi-agent system as a graph, with each agent represented as a node. Each agent in the graph makes a tentative decision or evaluates its state value, communicating this to neighboring agents. Subsequently, each agent updates its decision or state value concurrently, considering information from other agents.

Consensus algorithms can be categorized based on their assumptions about the graph topology and the relative emphasis on information exchange among different agents (see [42]–[48] and references therein). A common assumption in consensus algorithms across the literature is the balanced flow of information into and out of the agents, often represented by a doubly stochastic weight matrix [49], [50]. However, employing a doubly stochastic matrix poses challenges for decentralized consensus algorithm synthesis. With this weighting scheme, agents cannot independently determine the weights to assign to the neighbors using only local information. The requirement for the sum of entries in all rows and columns to equal 1 introduces a need for centralized coordination. Achieving a doubly stochastic matrix in practice assumes that an agent’s *in* and *out*-neighborhoods are identical, implying an undirected communication topology [50]. Furthermore, if identical weights are used for neighboring agents, network balance (i.e., each node having the same in and out-degree) is necessary, which is generally not the case.

The article [42] introduces the PushSum/Ratio-consensus algorithm, which is designed to achieve average consensus within a directed interconnection structure. A

notable feature of the PushSum algorithm is its ability for local synthesis by each agent, eliminating the need for global coordination. Operating through a column stochastic weighting rule, the algorithm involves two states that are updated iteratively. The estimation of the consensus variable is derived from a non-linear update based on the ratio of these two states. In addition to average consensus scenarios, the literature also explores general agreement problems in which the consensus value differs from the average of initial state values. This is achieved using a row stochastic weighting rule. Here, the sum of the agent state values during the algorithm updates does not remain conserved, precluding the achievement of an average consensus [51]. The PushSum protocol, by the use of a column stochastic matrix, circumvents this limitation. However, the non-linear nature of its updates complicates analysis. Addressing this challenge, the authors in [52] proposed an algorithm with linear updates, effectively achieving average consensus. In their approach, agents utilize both column and row stochastic weight matrices, guaranteeing convergence and enhancing analytical tractability.

Consensus protocols have been applied across various fields in the literature, serving diverse purposes. These applications span various domains, including the synchronization of coupled oscillators [53]–[55], the study of flocks of mobile dynamic agents [56]–[58], distributed sensor fusion in IoT networks [59], [60], formation control of vehicle and multi-robot systems [20], [61], [62], and distributed resource allocation in power networks [16], [63], among others. Depending on the nature of these applications, consensus protocols must address various challenges, such as dynamic interaction topologies between the agents, time-delayed and asynchronous communication, and the presence of communication noise among others.

To accommodate these challenges, numerous works in the literature propose modifications and improvements to consensus protocols. For instance, authors in [45], [64], [65] proposed consensus protocols that converge to the average of initial state values despite bounded time delays between communicated state values. Algorithms introduced in [64], [65] achieve average consensus even in the presence of dynamic underlying network topologies. The article [66] extends the PushSum algorithm to dynamic graph topologies under the condition that the union of the communication graphs at different time instants remains strongly connected infinitely often. The authors provide asymptotic guarantees for agreement between the states of the agents.

The works [67]–[102] focus on extending the consensus algorithms to scenarios with communication noise. The articles [67]–[100] focus on symmetric undirected graphs utilizing doubly stochastic matrices. The consensus protocols in [81], [101], [102] utilize

a row stochastic update. In most schemes communication noise is assumed to be a zero mean random variable independent across different agents and time instants [67]–[71], [73]–[82], [86]–[94], [96]–[105]. Article [70] considers the channel noise to follow a Poisson distribution, while [72] assumes a zero median communication noise. Additionally, [101] analyzes the algorithm under the assumption of communication noise with bounded p -th moments i.e., $\mathbb{E}|w_t|^p < \infty, p \in (1, 2]$. In [95], a multiplicative noise model is considered, where the noise intensities depend on the relative states of agents and form a martingale. A bounded noise model is assumed in [83]–[85], [88], where the communication link noise is allowed to randomly vary within upper and lower bounds. Articles [67]–[71], [73], [77], [79]–[82], [87], [90]–[95], [97]–[102], provide guarantees about weak consensus, where the agreement between the agents is reached in the mean square sense; the works in [67], [72], [74], [75], [77], [78], [90], [91], [95], [99], [101], [102], [104], [106] establish a strong consensus guarantee, ensuring an almost sure agreement between the agent states. However, no analysis about the proximity of the asymptotic consensus value to the average of the initial values of the agents is provided; under the assumption of bounded unknown noise, the analyses in [83]–[85], [88], [96], [105] offer an upper bound on the state residuals (differences between the estimates of different agents); however the agent estimates not necessarily achieve consensus.

The consensus protocols discussed earlier were designed for scenarios where each agent’s state is a scalar value and may not be directly suitable for large-scale applications. However, situations involving higher-dimensional states arise in various practical scenarios. For example, in distributed resource allocation problems, where a fixed amount of resource needs to be allocated among multiple agents while minimizing a convex cost associated with each resource [16], a higher-dimensional consensus protocol is required. Other applications requiring higher-dimensional data include diffusion learning, where local agents conduct model training using parallel local datasets [107]. After completing the training process, model parameters are shared with neighboring nodes to facilitate faster dissemination. Similarly, recent frameworks for unsupervised learning like Generative Adversarial Networks aim to train robust learning models against adversarial attacks [108]. Methods proposed by [109] scale adversarial training to large datasets in a distributed setup, where each agent trains a discriminator model on local data, and a shared generator model is trained based on feedback from different agents’ discriminator models. These applications often require sharing high-dimensional parameter vectors, typically of the order of $32 \times 32 \times 3 \sim 1024 \times 1024 \times 3$. Furthermore, applications such as spectrum sensing in ad hoc cognitive radio networks [24] and the

control of autonomous agents like unmanned aerial vehicles (UAVs) for search and survey operations [27] also rely on implementing higher-dimensional consensus protocols. Article [110] introduces a consensus protocol for vector-valued state variables, based on a leader-follower architecture. In this framework, agents are divided into anchors and sensors, where anchors, with fixed states, act as leaders, and sensors adjust their states via a convex combination with neighboring agents' states. This approach ensures convergence to the state of the anchors.

Termination of consensus algorithms in finite time provides an advantage of getting an approximate consensus while saving valuable computation and communication resources. For the *scalar* average consensus protocols, the works [111]–[115] focus on the design of a finite-time stopping criteria for *distributed* consensus algorithms with *discrete-time* updates over *static* communication topologies. In [111], the final consensus value can be computed by each node using the minimal polynomial associated with the weight matrix in the state update iterations. However, this method requires each node to execute n (the total number of agents) distinct linear iterations for at least $n + 1$ time steps to calculate the coefficients of the minimal polynomial. On the other hand, [112] achieved finite-time consensus based on the network observability matrix, necessitating significant storage and computational capabilities for matrix inversions and rank calculations at each node. In contrast, authors in [113]–[115] devised computationally efficient distributed detection mechanisms to determine the convergence of average consensus algorithms under static interconnection topologies. They demonstrated that the sequence of global maximum and minimum state values of the agents, following a consensus algorithm, forms a monotonic sequence converging to the consensus value when the network topology is fixed. To achieve finite-time consensus within a specified tolerance margin, a distributed maximum and minimum protocol was proposed to propagate these global extreme state values. This approach enables each node to detect progress toward consensus by having additional states corresponding to the global maximum and minimum values. Consequently, each node can detect convergence simultaneously within a finite time, leading to the termination of the consensus protocol at the same iteration across all nodes.

2.3 Research Questions

Research Focus 1:

A major challenge in applying consensus protocols in multi-agent systems arises from the presence of noise in wireless communication channels between the agents. Noisy communication links can obscure the true states of agents and potentially push their estimates far from the original consensus value (the average), or even cause the consensus protocol to diverge. Despite the extensions of the applicability of the PushSum algorithm under more flexible graph and communication scenarios in articles such as [43]–[48], [116], the performance guarantees of the PushSum algorithm in the presence of communication noise remains unaddressed. In light of this, this chapter of the thesis aims to tackle the following open research questions:

1. Can the celebrated PushSum algorithm, which is widely utilized in distributed decision-making applications over directed graphs, work under communication noise?
2. If the PushSum algorithm does not perform well under noisy communication, then design a consensus algorithm that has guarantees for agents to achieve consensus and converge towards the average of their initial values, even in the presence of communication noise. This new algorithm should also have all the appealing features of the PushSum algorithm, including distributed synthesis and implementation.

Research Focus 2:

Real-time distributed decision-making hinges on the timely availability of a consensus value for each agent, enabling them to proceed with subsequent tasks or actions. Therefore, establishing a finite-time distributed stopping criterion holds paramount importance in numerous applications. Additionally, if agents can autonomously detect convergence within a predefined tolerance margin of the consensus value, they can cease algorithmic execution, conserving valuable power and computational resources. The state-of-the-art methodologies presented in [113]–[115] offer a means to achieve approximate consensus within finite time under an *invariant graph topology for scalar variables* during consensus algorithm updates. In this context, in the current chapter also focuses on addressing the following key research questions:

1. Can the state-of-the-art algorithms [113]–[115] be adopted to determine an approximate consensus in finite time when the communication topology among agents is dynamic?
2. If not, then how to design a criterion to detect when the agents achieve (approximate) consensus to the average of their initial values when the graph topology is dynamic?
3. What geometric properties do the consensus algorithms need to satisfy in higher dimensions?
4. Can these geometric insights be leveraged towards a computation and communication efficient criterion for achieving practical consensus in higher dimensions?

2.4 Contributions of the Chapter

The current chapter makes the following contributions toward the two research focuses outlined above:

2.4.1 Contributions Towards Research Focus 1:

1. We establish that the PushSum algorithm fails to converge under imperfect communication with additive noise and is not applicable as such in these scenarios.
2. We propose the **Noise Resilient PushSum (NR-PushSum)** algorithm, which enhances and extends the PushSum algorithm by accommodating practical settings with noisy communication channels. Notably, this is the *first and only* such extension present in the literature.
3. The developed **NR-PushSum** algorithm possesses the following features:
 - i. In contrast to the existing algorithms in the literature that focus on achieving average consensus under noisy communication, the proposed **NR-PushSum** algorithm achieves average consensus over general directed graph topologies. Importantly, it doesn't necessitate restrictive assumptions regarding the communication graph's symmetry and/or weight-balance.
 - ii. The **NR-PushSum** algorithm can be synthesized in a distributed manner. Notably, using decentralized information, **NR-PushSum** can be *designed* (rather

than solely implemented distributively). This noteworthy advantage facilitates the seamless integration of new agents into the network.

- iii. The **NR-PushSum** algorithm attains average consensus at a geometric rate when communication is noiseless. In the presence of communication noise, the agents' estimates under the **NR-PushSum** algorithm achieve consensus (perfect agreement) *almost surely*. Additionally, the algorithm parameters can be adjusted to ensure the consensus value is close to the average of the agents' initial states.

2.4.2 Contributions Towards Research Focus 2:

1. We propose the first algorithm for determining approximate consensus under a directed dynamic interconnection topology between the agents. We extend the algorithms in [113], [114] to the case of time-varying topologies. The scheme proposed here is amenable for distributed implementation because it only requires each agent to know an upper bound on the number of nodes.
2. We introduce a novel concept of a time path that incorporates the influence of a node on the other nodes of the network with changes in time as well as the topology. We prove the existence of time paths with a finite length between every pair of nodes and establish an upper bound on the number of iterations required for each node in the network to influence all other nodes when the network topology changes.
3. We develop a geometric notion, termed “convex decreasing”, to get insights into the behavior of consensus algorithms. We prove that many popular consensus algorithms fit into this general class of algorithms. The convex decreasing property allows for guarantees to obtain approximate average consensus. In particular, we demonstrate that convex decreasing consensus algorithms satisfy monotone convergence properties. It can be determined that all nodes of the convex decreasing consensus algorithm belong to a convex set, then all algorithm nodes will remain in said convex set. We develop distributed stopping criteria based on this geometric insight.
4. We introduce two new algorithms for distributed stopping of $\mathbb{R}^d$ -valued consensus algorithms within an ε -threshold of consensus with respect to arbitrary norm.

5. The algorithms correspond to distributed diameter estimation methods of independent interest. In particular, the first diameter estimation algorithm is based on distributed convex hull determination, which is of broad utility. The second introduced algorithm is based on a radius-sharing technique. It has a minimal communication cost of the stopping algorithm, requiring only a single scalar value to be shared at each iteration. This is especially significant as previous stopping algorithms carried a communication cost of order d scalars.

2.5 Developed Solutions

2.5.1 Noise Resilient Distributed Average Consensus Over Directed Graphs

Consider a set of n agents connected through a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The agents communicate via uni-directional communication links in $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The communication link between any agent i and its neighbor j is corrupted by noise. Each agent $i \in \mathcal{V}$ has an initial state $u_i \in \mathbb{R}^p$. The objective is to design an algorithm that allows each agent to distributively compute the average,

$$\bar{u} := \frac{1}{n} \sum_{i=1}^n u_i, \quad (2.1)$$

by interacting only with its neighboring agents in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ under the communication constraints: uni-directional communication links with additive noise.

PushSum with noisy communication:

Under the PushSum algorithm, at any discrete time-instant k each agent i maintains three estimates $x_i(k) \in \mathbb{R}^p$, $y_i(k) \in \mathbb{R}$ and $z_i(k) \in \mathbb{R}^p$. Due to noise in communication channels in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ the agent i receives a noisy version of the information sent by the agent j . Node i updates its estimates at $(k+1)^{th}$ discrete time-instant, by performing a weighted combination of the (noisy) information received from its in-neighbors and

its own information (available without any additive noise at the agent i) as follows:

$$x_i(k+1) = p_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i^-} (p_{ij}x_j(k) + \eta_{x_{ij}}(k)), \quad (2.2)$$

$$y_i(k+1) = p_{ii}y_i(k) + \sum_{j \in \mathcal{N}_i^-} (p_{ij}y_j(k) + \eta_{y_{ij}}(k)), \quad (2.3)$$

where, $x_i(0) = u_i$, and $y_i(0) = 1$ for all $i \in [\mathbf{N}]$ and the terms $\eta_{x_{ij}}(k)$ and, $\eta_{y_{ij}}(k)$ are the additive noise realizations in the estimates of agent j sent to agent i at any instant k . The variable $z_i(k)$ is the estimate of the average $\bar{u}$ maintained by agent $i \in \mathcal{V}$ at time-instant k which is updated as,

$$z_i(k+1) = \frac{1}{y_i(k+1)}x_i(k+1). \quad (2.4)$$

Let $\mathbf{P}$ be weight matrix corresponding to (2.2) and (2.3) with $\mathbf{P}_{ij} \in [0, 1]$ and,

$$\mathbf{P}_{ij} = \begin{cases} p_{ij} \neq 0, & \text{if } i = j \text{ or } (i, j) \in \mathcal{E}, \\ 0, & \text{otherwise.} \end{cases} \quad (2.5)$$

The following assumption is typically made to study the PushSum algorithm:

Assumption 1. *The directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is strongly connected, and the associated weight matrix $\mathbf{P}$ is column stochastic.*

One choice of weights that satisfy Assumption 1 is the *out-degree based equal neighbor weights rule*: Here,

$$p_{ij} = \begin{cases} \frac{1}{1 + |\mathcal{N}_j^+|}, & \text{if } i \in \mathcal{N}_j^+, \forall j, \\ 0, & \text{if } i \notin \mathcal{N}_j^+, \forall j. \end{cases} \quad (2.6)$$

Define for all $i \in [\mathbf{N}]$,

$$\eta_{x_i}(k) := \sum_{j \in \mathcal{N}_i^-} \eta_{x_{ij}}(k), \quad \eta_{y_i}(k) := \sum_{j \in \mathcal{N}_i^-} \eta_{y_{ij}}(k). \quad (2.7)$$

We stack the agents' estimates in a column vector form and define the following quantities:

$$\mathbf{X}(k) := [x_1(k); \dots; x_n(k)] \in \mathbb{R}^{n \times p}, \quad (2.8)$$

$$\mathbf{y}(k) := [y_1(k); \dots; y_n(k)] \in \mathbb{R}^n, \quad (2.9)$$

$$\eta_x(k) := [\eta_{x_1}(k); \dots; \eta_{x_n}(k)] \in \mathbb{R}^{n \times p}, \text{ and} \quad (2.10)$$

$$\eta_y(k) := [\eta_{y_1}(k); \dots; \eta_{y_n}(k)] \in \mathbb{R}^n. \quad (2.11)$$

Thus, updates (2.2) and (2.3) can be written as:

$$\mathbf{X}(k+1) = \mathbf{P}\mathbf{X}(k) + \eta_x(k), \quad \mathbf{y}(k+1) = \mathbf{P}\mathbf{y}(k) + \eta_y(k),$$

where, $\eta_x(k)$ and $\eta_y(k)$ are the additive noise. Left multiplying by $\mathbf{1}^\top$ on both sides of the above equations, we have

$$\mathbf{1}^\top \mathbf{X}(k+1) = \mathbf{1}^\top \mathbf{X}(k) + \mathbf{1}^\top \eta_x(k), \text{ and } \mathbf{1}^\top \mathbf{y}(k+1) = \mathbf{1}^\top \mathbf{y}(k) + \mathbf{1}^\top \eta_y(k).$$

Thus, the sum of estimates $\mathbf{X}(k)$ and $\mathbf{y}(k)$ integrate the communication noise and thus is equivalent to a random walk with unbounded variance. Note that in the case when the communication links are noiseless the sum of the estimates $\mathbf{X}(k)$ and $\mathbf{y}(k)$ are conserved during the iterations [117] that plays a crucial role in the convergence of the PushSum algorithm to the average of the initial values. The random walk behavior of the sums implies that the estimates $\mathbf{X}(k)$ and $\mathbf{y}(k)$ and hence, $\mathbf{Z}(k) = [z_1(k), \dots, z_n(k)] \in \mathbb{R}^{n \times p}$ do not converge to a meaningful value. This is clear from the trajectories of the agent states $\mathbf{Z}_i(k)$ under the PushSum algorithm for a 10 node directed graph with noisy communication shown in Figure 2.1. Here, the initial values of the agents are chosen to be $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$ (with the average being 5.5). The communication links have an additive noise modeled as a normal random variable, $\mathcal{N}(0, 0.1)$. These observations suggest that the PushSum algorithm *does not* perform adequately under communication noise.

Achieving Average Consensus in Noisy Directed Networks:

In this section, we propose a distributed algorithm that allows all the agents to determine the initial average $\bar{u}$ (defined in (2.1)) under noisy communication. We call it the **NR-PushSum** algorithm as it is based on the PushSum algorithm, similar to

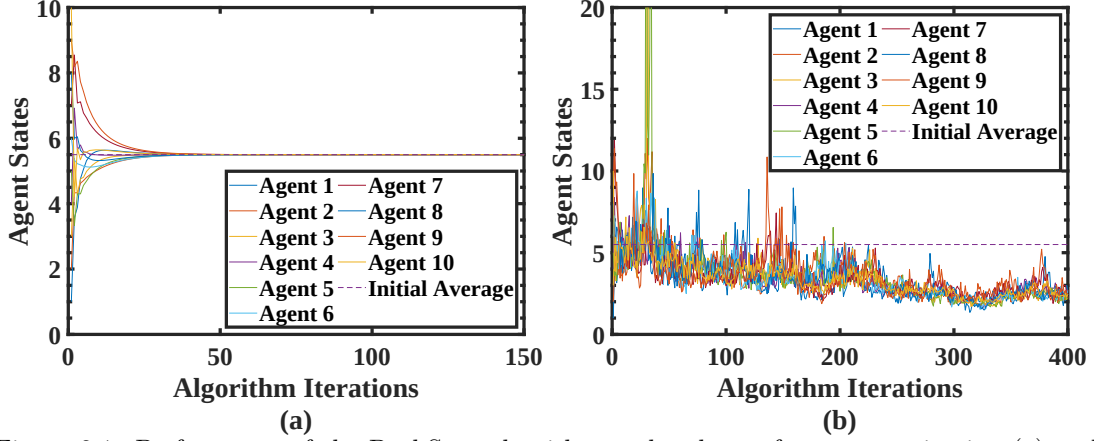


Figure 2.1: Performance of the PushSum algorithm under the perfect communication (a) and with communication noise (b).

which here each agent i , at any discrete time-instant k , maintains three estimates $x_i(k) \in \mathbb{R}^p$, $y_i(k) \in \mathbb{R}$ and $z_i(k) \in \mathbb{R}^p$. We utilize two strategies to design the **NR-PushSum** algorithm that counter the effect of the communication noise:

1. The estimates $x_i(k), y_i(k)$ are updated via a local mixing across the neighborhood of the agent $i \in \mathcal{V}$. Every agent i , weighs the information sent by its neighbors (agent $j \in \mathcal{N}_i^-$) and its own information to appropriately control the amount of the noise to be added to its updated estimates.
2. Each agent i , utilizes its initial values $x_i(0)$ and $y_i(0)$ to update its estimates. The initial values $x_i(0)$ and $y_i(0)$ are deterministic and known to the agent i . Utilizing the noise-free initial values provides a tether to the agents' estimates in tracking the initial average in the presence of communication noise.

Incorporating these two strategies together, the estimates in the **NR-PushSum** algorithm are updated as follows, for $i \in \mathcal{V}$:

$$x_i(k+1) = \alpha_i(k)p_{ii}x_i(k) + \beta(k) \sum_{j \in \mathcal{N}_i^-} (p_{ij}x_j(k) + \eta_{x_{ij}}(k)) + \theta(k)x_i(0), \quad (2.12)$$

$$y_i(k+1) = \alpha_i(k)p_{ii}y_i(k) + \beta(k) \sum_{j \in \mathcal{N}_i^-} (p_{ij}y_j(k) + \eta_{y_{ij}}(k)) + \theta(k)y_i(0), \quad (2.13)$$

respectively, where, $x_i(0) = u_i$, and $y_i(0) = 1$ for all $i \in [\mathbf{N}]$. $\alpha_i(k) \in [0, 1], \forall i \in [\mathbf{N}], k$, and $\beta(k) \in [0, 1], \forall k$ are the relative weights that agent i utilize to control the information mixing, and $\theta(k) \in \mathbb{R}, \forall k$, is utilized to control the influence of the noise-free

initial values on the estimate updates. The variable $z_i(k)$ maintained by agent $i \in \mathcal{V}$ at time-instant k is updated as in the PushSum algorithm described by

$$z_i(k+1) = \frac{1}{y_i(k+1)} x_i(k+1). \quad (2.14)$$

Figs. 2.2 and 2.3 show the estimate updates for an agent $i \in \mathcal{V}$ at any discrete-time instant k . Figure 2.2 shows the communication path where agent i receives information from its in-neighbors j and l . Figure 2.3 shows the computation step where agent i updates its estimates based on the information from the neighbors j and l , its own current state values $(x_i(k), y_i(k))$ and the noise-free initial values $(x_i(0), y_i(0))$.

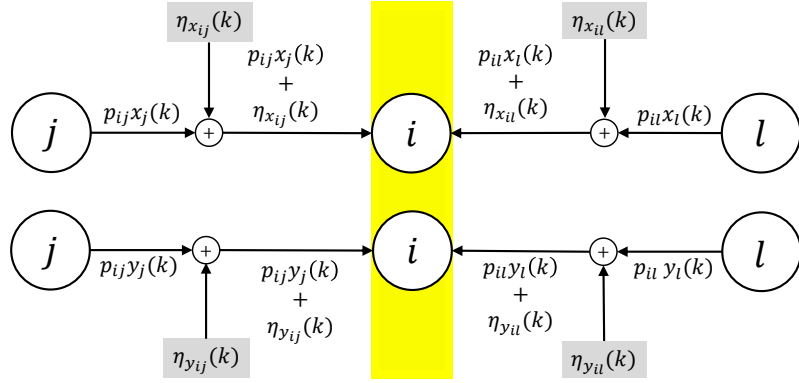


Figure 2.2: **NR-PushSum** algorithm update: agent i receives noisy information from neighbors j and l

Define $\hat{\mathbf{P}}$ such that

$$\hat{\mathbf{P}}_{ij} = \begin{cases} \mathbf{P}_{ij}, & \text{if } i \neq j \\ 0, & \text{if } i = j. \end{cases} \quad (2.15)$$

Note that due to Assumption 1 the matrix $\hat{\mathbf{P}}$ defined in (2.15) is a column sub-stochastic matrix, i.e., $\sum_{i=1}^n \hat{\mathbf{P}}_{ij} < 1$ for all $j \in [\mathbf{N}]$. The parameter $\alpha_i(k), i \in [\mathbf{N}]$ is set as:

$$\alpha_i(k) = \frac{1 - \beta(k)(1 - p_{ii})}{p_{ii}}. \quad (2.16)$$

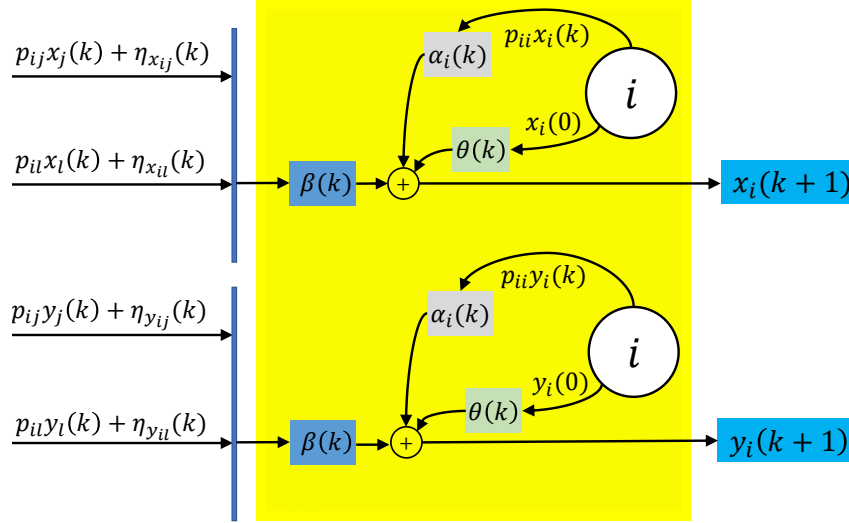


Figure 2.3: **NR-PushSum** algorithm update: agent i updates its estimates using the received information, its current estimate, and its initial value.

Let $\mathbf{A}(k) := (\text{diag}(\alpha_i(k)) \odot \mathbf{P})$ and $\mathbf{B}(k) := \beta(k) \hat{\mathbf{P}}$. Using the defined notations in (2.7)-(2.11), updates (2.12)-(2.14) can be written as:

$$\mathbf{X}(k+1) = \mathbf{A}(k)\mathbf{X}(k) + \mathbf{B}(k)\mathbf{X}(k) + \theta(k)\mathbf{X}(0) + \beta(k)\eta_x(k), \quad (2.17)$$

$$\mathbf{y}(k+1) = \mathbf{A}(k)\mathbf{y}(k) + \mathbf{B}(k)\mathbf{y}(k) + \theta(k)\mathbf{y}(0) + \beta(k)\eta_y(k), \quad (2.18)$$

$$\mathbf{Z}(k+1) = \mathbf{X}(k+1) \oslash \mathbf{y}(k+1). \quad (2.19)$$

The initial values of the agents in updates (2.12)-(2.14) help preserve the true information of the agents and reduce the effect of the communication noise. Moreover, the agents vary the weights $\beta(k)$ and $\theta(k)$ dynamically which allows the agents' estimates $x_i(k)$ and $y_i(k)$ to reach a steady state. Specifically, the weights $\beta(k)$ and $\theta(k)$ satisfy the following assumptions:

Assumption 2. *The sequence $\{\beta(k)\}_{k \geq 0}$ is such that:*

i) $\beta(k) \in [0, 1)$, for all k ,

ii) there exists a $K_\beta \geq 1$ such that

$$ak^{-q} \leq \beta(k) \leq bk^{-q}, \quad (2.20)$$

for all $k \geq K_\beta$, where $q > 1$ and $0 \leq a \leq b < \infty$.

Note that the requirement (2.20) with a suitable K_β , allows more flexibility in choosing $\beta(k)$ as larger values of a and b can be chosen while maintaining $\beta(k) \in [0, 1]$, $k \geq K_\beta$. In the sequel, the parameters a, b and q will be treated as fixed constants.

Assumption 3. *The sequence $\{\theta(k)\}_{k \geq 0}$ satisfies:*

$$\theta(k) \in [0, \infty), \text{ for all } k, \text{ and } \sum_{k=0}^{\infty} \theta(k) < \infty. \quad (2.21)$$

Note that one can appropriately choose a $K_\theta > 0$ with large values of $\theta(k)$ chosen early on, i.e. when $k < K_\theta$, to give more importance to the initial values of the agents and then $\theta(k)$ be gradually reduced, i.e. $\theta(k) \in [0, 1]$ for $k > K_\theta$ to satisfy Assumption 3. In section 2.5.1, we demonstrate the performance of the proposed **NR-PushSum** algorithm under different choices of the $\beta(k)$ and $\theta(k)$ sequences.

Remark 1. *The proposed **NR-PushSum** algorithm inherits nice properties of the PushSum algorithm, i.e. the **NR-PushSum** algorithm is applicable on general directed graph topologies and can be both implemented and synthesized/designed distributively. Recall Figure 2.2, every agent $j \in \mathcal{V}$ decides the weights p_{ij} for sending its information over the out-going links to the neighbor agent i , independent of the other agents. The agent $i \in \mathcal{V}$ upon receiving the information from its in-neighbors adds up its estimate and the initial value with the received information after weighing them by weights $\alpha_i(k), \theta(k)$ and $\beta(k)$ respectively. Note that the agents do not need to have any centralized information for weights $\beta(k)$ and $\theta(k)$. Each agent can dynamically update the weights $\beta(k)$ and $\theta(k)$ based on Assumptions 2 and 3.*

Convergence Analysis:

The proposed **NR-PushSum** algorithm being new, we examine its convergence properties in the noiseless case. Following that, our analysis expands to investigate the *almost sure* convergence of the proposed **NR-PushSum** algorithm to a consensus value, even in the presence of additive noise affecting communication channels. As the proposed **NR-PushSum** algorithm itself is new, its properties in the noiseless case must be established. This examination helps establish that the proposed **NR-PushSum** algorithm doesn't compromise convergence properties without noise, all while attaining favorable guarantees in the presence of noise. Furthermore, the exploration of the noise-free scenario provides a solid mathematical foundation that can be extended to establish

guarantees for our method in situations where communication is affected by additive noise.

A. Analysis under noiseless communication channels:

Here, we will analyze a special case of the scheme (2.12)-(2.14) where the communication channels are assumed to be noiseless. In particular, $\eta_{x_{ij}}(k) = 0 = \eta_{y_{ij}}(k)$ for all $i, j \in [\mathbf{N}]$ at all time-instants $k \geq 0$. Let $\mathbf{M}(k) := \mathbf{A}(k) + \mathbf{B}(k)$ for all $k \geq 0$ and define the following operation,

$$\widehat{\prod}_{r=s}^l \mathbf{M}(r) := \begin{cases} \prod_{r=s}^l \mathbf{M}(r), & \text{if } l \geq s \\ \mathbf{I}_n, & \text{otherwise.} \end{cases} \quad (2.22)$$

Further define,

$$W(l, k) := \widehat{\prod}_{r=l}^k \mathbf{M}(r). \quad (2.23)$$

We begin by establishing that the following relation holds.

Lemma 1. $\sum_{j=1}^n W_{ij}(l, k) > w_o^{k-l+1}$, for all $k \geq l \geq 1$, where, $w_o > 0$ is a constant.

Proof. For $\mathbf{M}(l), l \geq 1$ note that,

$$\begin{aligned} \sum_{j=1}^n \mathbf{M}_{ij}(l) &= 1 - \beta(l)(1 - p_{ii}) + \beta(l) \sum_{j \neq i}^n p_{ij} \\ &= 1 - \beta(l) + \beta(l) \sum_{j=1}^n p_{ij} \\ &> 1 - \beta(l) > 1 - \max_l \beta(l) = w_o > 0. \end{aligned} \quad (2.24)$$

We use an argument based on induction on the matrices of the form $W(l, l+t)$. Let $T = k - l$. For $t = 1$, note that $W(l, l+1) = \mathbf{M}(l)\mathbf{M}(l+1)$ and the i - j^{th} entry of the matrix $W(l, l+1)$ is given by $W_{ij}(l, l+1) = [\mathbf{M}(l)\mathbf{M}(l+1)]_{ij} = \sum_{s=1}^n \mathbf{M}_{is}(l)\mathbf{M}_{sj}(l+1)$.

Taking the sum over j we have,

$$\begin{aligned}
\sum_{j=1}^n W_{ij}(l, l+1) &= \sum_{j=1}^n \sum_{s=1}^n \mathbf{M}_{is}(l) \mathbf{M}_{sj}(l+1) \\
&= \sum_{s=1}^n \left[\mathbf{M}_{is}(l) \sum_{j=1}^n \mathbf{M}_{sj}(l+1) \right] \\
&> \sum_{s=1}^n \mathbf{M}_{is}(l) w_o > w_o^2,
\end{aligned}$$

where, we used (2.24) in the last two inequalities. Assume, for $t = T = k - l$, that $\sum_{j=1}^n W_{ij}(l, l+T) = \sum_{j=1}^n W_{ij}(l, k) > w_o^{k-l+1}$. For $t = T + 1$,

$$\begin{aligned}
\sum_{j=1}^n W_{ij}(l, l+T+1) &= \sum_{j=1}^n \sum_{s=1}^n W_{is}(l, l+T) \mathbf{M}_{sj}(l+T+1) \\
&= \sum_{s=1}^n \left[W_{is}(l, l+T) \sum_{j=1}^n \mathbf{M}_{sj}(l+T+1) \right] \\
&> \sum_{s=1}^n W_{is}(l, l+T) w_o \\
&> w_o^{k-l+1} w_o = w_o^{k-l+2},
\end{aligned}$$

where, we used (2.24) and the induction hypothesis in the last inequality. Therefore, induction holds. Thus,

$$\sum_{j=1}^n W_{ij}(l, k) > w_o^{k-l+1}, \tag{2.25}$$

for all $k \geq l \geq 1$. □

Lemma 2. *Let Assumption 1 hold. Assume that the communication channels in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ are **noiseless**. Let the sequences $\{\beta(k)\}_{k \geq 0}$ and $\{\theta(k)\}_{k \geq 0}$ satisfy assumptions 2 and 3 respectively. Under the initialization $y_i(0) = 1$ for all $i \in [\mathbf{N}]$, there exist $y_o > 0$ such that we have $y_i(k) > y_o$ for all $k \geq 1, i \in [\mathbf{N}]$. As a consequence the states $z_i(k)$ are well-defined for all $k \geq 1, i \in [\mathbf{N}]$.*

Proof. For $\eta_y(k) = 0$, with $\mathbf{M}(k) := \mathbf{A}(k) + \mathbf{B}(k)$ the update $\mathbf{y}(k)$ in the iteration (2.18)

becomes:

$$\mathbf{y}(k+1) = \mathbf{M}(k)\mathbf{y}(k) + \theta(k)\mathbf{y}(0). \quad (2.26)$$

Recursive application of (2.26) using the notation in (2.22) gives,

$$\mathbf{y}(k+1) = \widehat{\prod}_{r=0}^k \mathbf{M}(r)\mathbf{y}(0) + \sum_{s=0}^k \widehat{\prod}_{r=s+1}^k \mathbf{M}(r)\theta(s)\mathbf{y}(0). \quad (2.27)$$

From the definition of matrices $\mathbf{A}(k)$ and $\mathbf{B}(k)$ under Assumption 1 it is easy to check that the matrix $\mathbf{M}(k)$ is column-stochastic for all $k \geq 0$:

Recall $\mathbf{M}(k) := \mathbf{A}(k) + \mathbf{B}(k)$. Consider the column sum, for any column j ,

$$\begin{aligned} \sum_{i=1}^n \mathbf{M}_{ij}(k) &= \alpha_{ii}(k)p_{ii} + \beta(k) \sum_{i=1, i \neq j}^n p_{ij} = \frac{1 - \beta(k)(1 - p_{ii})}{p_{ii}} p_{ii} + \beta(k) \sum_{i=1, i \neq j}^n p_{ij} \\ &= 1 - \beta(k)(1 - p_{ii}) + \beta(k)(1 - p_{ii}) = 1, \end{aligned}$$

where, we used (2.16) in the second equality and the fact that matrix $\mathbf{P}$ satisfy Assumption 1. Since j is arbitrary, we conclude that $\mathbf{M}(k)$ is column-stochastic.

Further, note that $W(l, k)$ for all $k, l \geq 0$ is a column stochastic, Irreducible, and Aperiodic (cSIA) matrix:

Claim: Let R and S be two $n \times n$ column stochastic matrices. Then $RS \in \mathbb{R}^{n \times n}$ is column stochastic.

Proof: Consider, the sum

$$\sum_{i=1}^n [RS]_{ij} = \sum_{i=1}^n \sum_{k=1}^n R_{ik} S_{kj} = \sum_{k=1}^n \sum_{i=1}^n R_{ik} S_{kj} = \sum_{k=1}^n S_{kj} = 1,$$

where, we used the fact that R and S are column-stochastic. Thus, RS is column-stochastic. $\square$

Column stochasticity follows by induction and the column stochasticity of $\mathbf{M}(k)$. From the expression for $W(l, k)$ in (2.23), the zero/non-zero structure of $W(l, k)$ corresponds to a graph of n nodes that includes all the edges in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and since the original graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is strongly connected, $W(l, k)$ corresponds to a graph that is strongly connected

and hence is irreducible [118]. Since the graph corresponding to $W(l, k)$ is strongly connected and the diagonal entries of $W(l, k)$ are positive, therefore, $W(l, k)$ is aperiodic. We utilize the following result from [119] on the product of column stochastic matrices:

Theorem 1. *Let $\mathcal{M} := \{\mathbf{M}(1), \mathbf{M}(2), \dots\}$ be a collection of $n \times n$ column stochastic matrices. Define a word in $\mathcal{M}$ of length ℓ as the product of ℓ matrices from the collection $\mathcal{M}$ (repetitions permitted). Let $\mathcal{M}$ be such that any word in $\mathcal{M}$ is cSIA. Then given $\varepsilon > 0$ there exists an integer ℓ_ε such that any word $\mathbf{N} \in \mathbb{R}^{n \times n}$ in $\mathcal{M}$ of length $\ell \geq \ell_\varepsilon$ satisfies $\max_j \max_{i_1, i_2} |\mathbf{N}_{j, i_1} - \mathbf{N}_{j, i_2}| < \varepsilon$.*

Theorem 1 implies that any sufficiently long word in the $\mathcal{M}$ has all its columns such that the difference in the columns element-wise can be made arbitrarily small. Thus, given $\varepsilon > 0$, there exists a $\ell_\varepsilon > 0$ such that $W(l, k)$ (a word in $\mathcal{M}$ that is cSIA) is equal to $\mathbf{v}(k)\mathbf{1}^\top + E(k)$, for all $k > l + \ell_\varepsilon$, where $\mathbf{v}(k)$ is an appropriate vector with positive entries, and $E(k)$ is an error matrix such that the absolute value of its elements is smaller than $\varepsilon/2$ i.e., $|E_{ij}(k)| < \varepsilon/2$ for all $i, j \in [\mathbf{N}]$.

Further, recall $\mathbf{M}(k) := \mathbf{A}(k) + \mathbf{B}(k)$. Based on the definition of $\mathbf{A}(k)$ and $\mathbf{B}(k)$, we can re-write $\mathbf{M}(k)$ as

$$\mathbf{M}(k) := \mathbf{I}_n + \beta(k)\tilde{\mathbf{P}}, \quad (2.28)$$

where, $\tilde{\mathbf{P}} = \mathbf{diag}(p_{ii} - 1) + \hat{\mathbf{P}}$.

Given any $\varepsilon > 0$ for $k > \ell_\varepsilon$, note that $W(0, k) = \mathbf{v}(k)\mathbf{1}^\top + E(k)$, with $|E_{ij}(k)| < \varepsilon/2$ for all i, j .

$$\begin{aligned} W(0, k) &= W(0, \ell_\varepsilon) \prod_{r=\ell_\varepsilon+1}^k \mathbf{M}(r) \\ &= [\mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top + E(\ell_\varepsilon)] \prod_{r=\ell_\varepsilon+1}^k \mathbf{M}(r) \\ &= [\mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top + E(\ell_\varepsilon)] \prod_{r=\ell_\varepsilon+1}^k (\mathbf{I}_n + \beta(r)\tilde{\mathbf{P}}) \\ &= \mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top \prod_{r=\ell_\varepsilon+1}^k (\mathbf{I}_n + \beta(r)\tilde{\mathbf{P}}) + E(\ell_\varepsilon) \prod_{r=\ell_\varepsilon+1}^k (\mathbf{I}_n + \beta(r)\tilde{\mathbf{P}}) \\ &= \mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top + E(\ell_\varepsilon)W(\ell_\varepsilon + 1, k), \end{aligned} \quad (2.29)$$

where, we used the fact that $\mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top \tilde{\mathbf{P}} = \mathbf{0}_{n \times n}$.

From (2.27), (2.23) (2.28), (2.29), and initialization $y_j(0) = 1$ for all j we have for $k > \ell_\varepsilon$,

$$\begin{aligned}
y_i(k+1) &= \sum_{j=1}^n W_{ij}(0, k) + \sum_{j=1}^n \sum_{s=0}^k \theta(s) W_{ij}(s+1, k) \\
&> \sum_{j=1}^n W_{ij}(0, k) \\
&= \sum_{j=1}^n [\mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top + E(\ell_\varepsilon)W(\ell_\varepsilon + 1, k)]_{ij} \\
&> \sum_{j=1}^n [\mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top - \frac{\varepsilon}{2}\mathbf{1}\mathbf{1}^\top W(\ell_\varepsilon + 1, k)]_{ij} \\
&> \sum_{j=1}^n [\mathbf{v}(\ell_\varepsilon)\mathbf{1}^\top - \frac{\varepsilon}{2}\mathbf{1}\mathbf{1}^\top]_{ij} \\
&= n\mathbf{v}_i(\ell_\varepsilon) - n\frac{\varepsilon}{2} \\
&> n \min_{1 \leq i \leq n} \mathbf{v}_i(\ell_\varepsilon) - \varepsilon \frac{n}{2} > 0,
\end{aligned}$$

where, we used the fact that $W(l, k)$ is column stochastic for all $k, l \geq 0$ in the third last inequality. For $k \leq \ell_\varepsilon$, from Lemma 1 we have,

$$\begin{aligned}
y_i(k) &> w_o^{k-1} + \sum_{s=0}^{k-1} \theta(s) w_o^{k-s-1} \\
&> w_o^{k-1} + \theta(0) w_o^{k-1} \geq (1 + \theta(0)) w_o^{\ell_\varepsilon} > 0.
\end{aligned}$$

Therefore, $y_i(k) > y_o := \min\{n \min_{1 \leq i \leq n} \mathbf{v}_i(\ell_\varepsilon), (1 + \theta(0)) w_o^{\ell_\varepsilon}\}$, for all $k \geq 1$. Thus, $z_i(k) = \frac{1}{y_i(k)} x_i(k)$ is well defined for all $i \in [\mathbf{N}], k \geq 0$. $\square$

Theorem 2. *Let Assumption 1 hold. Assume the edges in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ are **not** corrupted by additive noise in communication channels. Let $\{z_i(k)\}_{k \geq 0}^{i \in [\mathbf{N}]}$ be the sequences generated by the iteration (2.14). Let the sequence $\{\theta(k)\}_{k \geq 0}$ satisfy Assumption 3. Then, the estimates $z_i(k)$ converges to the average $\bar{u}$, i.e., $\lim_{k \rightarrow \infty} z_i(k) = \bar{u}$, for all $i \in [\mathbf{N}]$.*

Proof. Similar to (2.26) the update $\mathbf{X}(k)$ in the iteration (2.17) when $\eta_x(k) = 0$ becomes:

$$\mathbf{X}(k+1) = \mathbf{M}(k)\mathbf{X}(k) + \theta(k)\mathbf{X}(0). \quad (2.30)$$

Recursive application of (2.30) with the notation (2.23) gives,

$$\mathbf{X}(k+1) = \underbrace{W(0, k)\mathbf{X}(0)}_{\text{first}} + \underbrace{\sum_{s=0}^k \theta(s)W(s+1, k)\mathbf{X}(0)}_{\text{second}}. \quad (2.31)$$

Note that as $W(l, k)$ is column stochastic for all $k, l \geq 1$, $\|W(l, k)\| \leq \bar{w} < \infty$ for all k, l .

Under Assumption 3 and from Theorem 1, given $\varepsilon > 0$. Define $\hat{\varepsilon} := \frac{\varepsilon / \sum_{s=0}^{\infty} \theta(s)}{\max\{\|\mathbf{X}(0)\|, n\}}$. There exists $\tilde{k}$ and $\mathbf{v}(k) \in \mathbb{R}^n$ such that $\sum_{s=\tilde{k}}^{\infty} \theta(s) \leq \frac{\hat{\varepsilon}}{2\bar{w}}$ and $W(l, k) = \mathbf{v}(k)\mathbf{1}^\top + E(k)$, for all $k > \tilde{k} + l$, where, $|E_{ij}(k)| \leq \frac{\hat{\varepsilon}}{2}$ for all $i, j \in [\mathbf{N}]$.

Thus, for all $k > \tilde{k}$, the first term in (2.31) becomes,

$$W(0, k)\mathbf{X}(0) = [\mathbf{v}(k)\mathbf{1}^\top + E(k)]\mathbf{X}(0) \quad (2.32)$$

where, $|E_{ij}(k)| < \hat{\varepsilon}/2$ for all $i, j \in [\mathbf{N}]$. Consider the second term in (2.31),

$$\sum_{s=0}^k \theta(s)W(s+1, k)\mathbf{X}(0) = \underbrace{\sum_{s=0}^{\tilde{k}} \theta(s)W(s+1, k)\mathbf{X}(0)}_{:=\psi_1(k)} + \underbrace{\sum_{s=\tilde{k}+1}^k \theta(s)W(s+1, k)\mathbf{X}(0)}_{:=\psi_2(k)} \quad (2.33)$$

Note, $k > \tilde{k} + s + 1$ for all the terms in $\psi_1(k)$. Substituting $W(s+1, k) := \mathbf{v}(k)\mathbf{1}^\top + \hat{E}(k)$, with $|\hat{E}_{ij}(k)| < \hat{\varepsilon}/2$ for all $i, j \in [\mathbf{N}]$ we get,

$$\psi_1(k)\mathbf{X}(0) = \sum_{s=0}^{\tilde{k}} \theta(s)[\mathbf{v}(k)\mathbf{1}^\top + \hat{E}(k)]\mathbf{X}(0). \quad (2.34)$$

Consider $\psi_2(k)$,

$$\begin{aligned} \psi_2(k) &= \sum_{s=\tilde{k}+1}^k W(s+1, k)\theta(s) \\ &= \theta(k) + \mathbf{M}(k)\theta(k-1) + W(k-1, k)\theta(k-2) + \cdots + W(\tilde{k}+2, k)\theta(\tilde{k}+1). \end{aligned}$$

Therefore,

$$\|\psi_2(k)\| \leq \theta(k) + \bar{w}\theta(k-1) + \cdots + \bar{w}\theta(\tilde{k}+1) \leq \bar{w} \sum_{s=\tilde{k}}^{\infty} \theta(s) \leq \frac{\hat{\varepsilon}}{2}. \quad (2.35)$$

Using (2.31)-(2.35) we have for all $k > \tilde{k}$,

$$\mathbf{X}(k+1) \leq \underbrace{\left(1 + \sum_{s=0}^{\tilde{k}} \theta(s)\right) \mathbf{v}(k) \mathbf{1}^\top \mathbf{X}(0) + \varepsilon \mathbf{1}_n \mathbf{1}_p^\top}_{:=\chi_1(k)}, \quad (2.36)$$

$$\mathbf{X}(k+1) \geq \left(1 + \sum_{s=0}^{\tilde{k}} \theta(s)\right) \mathbf{v}(k) \mathbf{1}^\top \mathbf{X}(0) - \varepsilon \mathbf{1}_n \mathbf{1}_p^\top, \quad (2.37)$$

where, $\mathbf{1}_n$ and $\mathbf{1}_p$ are the vectors of dimensions n and p with all entries equal to 1. We drop the subscript when the dimension of the vector is clear from the context. Following the similar procedure for (2.26) we get, for $k \geq \tilde{k}$,

$$\mathbf{y}(k+1) \leq \chi_1(k) \mathbf{y}(0) + \varepsilon \mathbf{1}, \text{ and}, \quad (2.38)$$

$$\mathbf{y}(k+1) \geq \chi_1(k) \mathbf{y}(0) - \varepsilon \mathbf{1}. \quad (2.39)$$

From (2.19), and (2.36)-(2.39), for any $i \in [\mathbf{N}]$,

$$\frac{[\chi_1(k) \mathbf{X}(0) - \varepsilon \mathbf{1}_n \mathbf{1}_p^\top]_i}{[\chi_1(k) \mathbf{y}(0) + \varepsilon \mathbf{1}]_i} \leq z_i(k+1) \leq \frac{[\chi_1(k) \mathbf{X}(0) + \varepsilon \mathbf{1}_n \mathbf{1}_p^\top]_i}{[\chi_1(k) \mathbf{y}(0) - \varepsilon \mathbf{1}]_i}, \quad (2.40)$$

where, $[\cdot]_i$ denote the i^{th} row (i^{th} entry) of the corresponding matrices (vectors). Denote $\bar{\theta} := 1 + \sum_{s=0}^{\tilde{k}} \theta(s)$, and $\mathbf{v}_i(k)$ as the i^{th} entry of $\mathbf{v}(k)$.

For $k > \tilde{k}$, for any $i \in [\mathbf{N}]$, from (2.40)

$$\begin{aligned} z_i(k+1) &\leq \frac{[\chi_1(k) \mathbf{X}(0) + \varepsilon \mathbf{1}_n \mathbf{1}_p^\top]_i}{[\chi_1(k) \mathbf{y}(0) - \varepsilon \mathbf{1}]_i} \\ &= \frac{\bar{\theta} \mathbf{v}_i(k) \mathbf{1}^\top \mathbf{X}(0) + \varepsilon \mathbf{1}^\top}{\bar{\theta} \mathbf{v}_i(k) \mathbf{1}^\top \mathbf{y}(0) - \varepsilon} \\ &= \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}} \mathbf{1}^\top}{1 - \frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}}} = \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \left(\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \mathbf{1}^\top \right) \left(\frac{\frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}}}{1 - \frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}}} \right) \end{aligned}$$

Similarly, for $k > \tilde{k}$, for any $i \in [\mathbf{N}]$,

$$\begin{aligned} z_i(k+1) &\geq \frac{[\chi_1(k)\mathbf{X}(0) - \varepsilon \mathbf{1}_n \mathbf{1}_p^\top]_i}{[\chi_1(k)\mathbf{y}(0) + \varepsilon \mathbf{1}]_i} \\ &= \frac{\bar{\theta} \mathbf{v}_i(k) \mathbf{1}^\top \mathbf{X}(0) - \varepsilon \mathbf{1}^\top}{\bar{\theta} \mathbf{v}_i(k) \mathbf{1}^\top \mathbf{y}(0) + \varepsilon} \\ &= \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} - \frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}} \mathbf{1}^\top}{1 + \frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}}} = \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} - \left(\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \mathbf{1}^\top \right) \left(\frac{\frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}}}{1 + \frac{\varepsilon}{n \mathbf{v}_i(k) \bar{\theta}}} \right) \end{aligned}$$

Let $\bar{\varepsilon}$ be such that, $\varepsilon \leq \frac{\bar{\varepsilon}}{(1+\bar{\varepsilon})}(n\bar{\theta}\mathbf{v}_i(k))$. Therefore, for all $k > \tilde{k}$, and for all $i \in [\mathbf{N}]$,

$$-\left(\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \mathbf{1}^\top \right) \bar{\varepsilon} \leq z_i(k) - \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} \leq \left(\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \mathbf{1}^\top \right) \bar{\varepsilon},$$

where, $\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} = \bar{u}$. Note that $\bar{\varepsilon}$ can be made arbitrarily small. Thus, we have the desired result. $\square$

Theorem 3. *Let Assumption 1 hold. Assume the edges in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ are **not** corrupted by additive noise in communication channels. Let $\{x_i(k), y_i(k), z_i(k)\}_{k \geq 0}^{i \in [\mathbf{N}]}$ be the sequences generated by iterations (2.12)-(2.14). Let the sequence $\{\theta(k)\}_{k \geq 0}$ satisfy Assumption 3. Then, for all $j \in [\mathbf{N}]$ and $k \geq 0$,*

$$\left\| z_j(k) - \bar{u} \right\| \leq C_1 \left[\lambda^k + \sum_{s=0}^{k-1} \theta(s) \lambda^{k-s-1} + \frac{1}{C} \theta(k) \right],$$

where, C, C_1 and $\lambda \in (0, 1)$ are some positive constants.

Proof. Recall (2.27) and (2.31),

$$\begin{aligned} \mathbf{y}(k+1) &= W(0, k) \mathbf{y}(0) + \sum_{s=0}^k \theta(s) W(s+1, k) \mathbf{y}(0), \\ \mathbf{X}(k+1) &= W(0, k) \mathbf{X}(0) + \sum_{s=0}^k \theta(s) W(s+1, k) \mathbf{X}(0). \end{aligned}$$

Since, $W(l, k), k \geq l \geq 0$ is column stochastic we have,

$$\begin{aligned}\mathbf{1}^\top \mathbf{X}(k+1) &= \mathbf{1}^\top W(0, k) \mathbf{X}(0) + \sum_{s=0}^k \theta(s) \mathbf{1}^\top W(s+1, k) \mathbf{X}(0) \\ &= \mathbf{1}^\top \mathbf{X}(0) + \sum_{s=0}^k \theta(s) \mathbf{1}^\top \mathbf{X}(0).\end{aligned}\quad (2.41)$$

From [120] Corollary 2, there exists stochastic vectors $\mathbf{v}(k) \in \mathbb{R}^n$ and constants $C, \lambda \in (0, 1)$ such that for any given i , for all $j = 1, \dots, n$,

$$|W_{ij}(l, k) - \mathbf{v}_i(k)| \leq C\lambda^{k-l}, \text{ for all } k \geq l \geq 0. \quad (2.42)$$

Multiplying $\mathbf{M}(k+1)$ on both sides to (2.31) and $\mathbf{v}(k+1)$ to (2.41) and subtracting we get,

$$\begin{aligned}(\mathbf{M}(k+1) - \mathbf{v}(k+1)\mathbf{1}^\top) \mathbf{X}(k+1) &= (W(0, k+1) - \mathbf{v}(k+1)\mathbf{1}^\top) \mathbf{X}(0) + \sum_{s=0}^k \theta(s) (W(s+1, k+1) - \mathbf{v}(k+1)\mathbf{1}^\top) \mathbf{X}(0) \\ \implies \mathbf{M}(k+1) \mathbf{X}(k+1) &= \mathbf{v}(k+1)\mathbf{1}^\top \mathbf{X}(k+1) + (W(0, k+1) - \mathbf{v}(k+1)\mathbf{1}^\top) \mathbf{X}(0) \\ &\quad + \sum_{s=0}^k (W(s+1, k+1) - \mathbf{v}(k+1)\mathbf{1}^\top) \mathbf{X}(0) \theta(s).\end{aligned}\quad (2.43)$$

Define $\mathbf{D}(l, k) := W(l, k) - \mathbf{v}(k)\mathbf{1}^\top$. Replacing $k+1$ by k in (2.43) and adding $\theta(k)\mathbf{X}(0)$ on both sides. Comparing with (2.30) we get,

$$\begin{aligned}\mathbf{X}(k+1) &= \mathbf{M}(k) \mathbf{X}(k) + \theta(k) \mathbf{X}(0) \\ &= \mathbf{v}(k)\mathbf{1}^\top \mathbf{X}(k) + \mathbf{D}(0, k) \mathbf{X}(0) + \sum_{s=0}^{k-1} \mathbf{D}(s+1, k) \mathbf{X}(0) \theta(s) + \theta(k) \mathbf{X}(0).\end{aligned}\quad (2.44)$$

Similarly, for (2.27)

$$\begin{aligned}\mathbf{y}(k+1) &= \mathbf{M}(k) \mathbf{y}(k) + \theta(k) \mathbf{y}(0) \\ &= \mathbf{v}(k)\mathbf{1}^\top \mathbf{y}(k) + \mathbf{D}(0, k) \mathbf{y}(0) + \sum_{s=0}^{k-1} \mathbf{D}(s+1, k) \mathbf{y}(0) \theta(s) + \theta(k) \mathbf{y}(0).\end{aligned}\quad (2.45)$$

Using (2.19), (2.44), and (2.45) for any agent i and $k \geq 0$,

$$\begin{aligned}
& z_i(k+1) - \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} \\
&= \frac{[\mathbf{v}(k)\mathbf{1}^\top \mathbf{X}(k) + \mathbf{D}(0, k)\mathbf{X}(0)]_i + \sum_{s=0}^{k-1} [\mathbf{D}(s+1, k)\mathbf{X}(0)\theta(s)]_i + \theta(k)\mathbf{X}_i(0)}{y_i(k)} - \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} \\
&= \frac{\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{X}(k) + \mathbf{D}_i(0, k)\mathbf{X}(0) + \sum_{s=0}^{k-1} \theta(s)\mathbf{D}_i(s+1, k)\mathbf{X}(0) + \theta(k)\mathbf{X}_i(0)}{y_i(k)} - \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} \\
&= \frac{n(\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{X}(k) + \mathbf{D}_i(0, k)\mathbf{X}(0) + \sum_{s=0}^{k-1} \theta(s)\mathbf{D}_i(s+1, k)\mathbf{X}(0) + \theta(k)\mathbf{X}_i(0))}{ny_i(k)} \\
&\quad - \frac{(\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{y}(k) + \mathbf{D}_i(0, k)\mathbf{y}(0) + \sum_{s=0}^{k-1} \theta(s)\mathbf{D}_i(s+1, k)\mathbf{y}(0))\mathbf{1}^\top \mathbf{X}(0)}{ny_i(k)} - \frac{\theta(k)\mathbf{y}_i(0)\mathbf{1}^\top \mathbf{X}(0)}{ny_i(k)} \\
&= \frac{\mathbf{D}_i(0, k)(n\mathbf{X}(0) - \mathbf{y}(0)\mathbf{1}^\top \mathbf{X}(0))}{ny_i(k)} + \frac{\sum_{s=0}^{k-1} \theta(s)\mathbf{D}_i(s+1, k)(n\mathbf{X}(0) - \mathbf{y}(0)\mathbf{1}^\top \mathbf{X}(0))}{ny_i(k)} \\
&\quad + \frac{\theta(k)(n\mathbf{X}_i(0) - \mathbf{y}_i(0)\mathbf{1}^\top \mathbf{X}(0))}{ny_i(k)},
\end{aligned}$$

where, $[\cdot]_i$ denote the i^{th} row of the corresponding matrix and in the last equality we used the fact $\frac{n\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{X}(k)}{ny_i(k)} - \frac{\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{y}(k)\mathbf{1}^\top \mathbf{X}(0)}{ny_i(k)} = 0$. Note $\|n\mathbf{X}(0) - \mathbf{y}(0)\mathbf{1}^\top \mathbf{X}(0)\| \leq \|n\mathbf{I}_n - \mathbf{1}\mathbf{1}^\top\| \|\mathbf{X}(0)\| \leq 2n^2 \|\mathbf{X}(0)\|$. From Lemma 2, there exists $y_o > 0$ such that $\mathbf{y}_i(k) > y_o$ for all i and $k \geq 0$. Therefore,

$$\begin{aligned}
\left\| z_i(k) - \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} \right\| &\leq \frac{\|\mathbf{D}_i(0, k)\| \|n\mathbf{X}(0) - \mathbf{y}(0)\mathbf{1}^\top \mathbf{X}(0)\|}{n\|y_i(k)\|} + \frac{\theta(k)\|n\mathbf{X}_i(0) - \mathbf{y}_i(0)\mathbf{1}^\top \mathbf{X}(0)\|}{n\|y_i(k)\|} \\
&\quad + \frac{\sum_{s=0}^{k-1} \theta(s)\|\mathbf{D}_i(s+1, k)\| \|n\mathbf{X}(0) - \mathbf{y}(0)\mathbf{1}^\top \mathbf{X}(0)\|}{n\|y_i(k)\|}, \\
&\leq \frac{2n^3 \|\mathbf{X}(0)\| C \lambda^k}{ny_o} + \frac{2n^3 \|\mathbf{X}(0)\| \theta(k)}{ny_o} + \frac{2n^3 \|\mathbf{X}(0)\| C \sum_{s=0}^{k-1} \theta(s) \lambda^{k-s-1}}{ny_o} \\
&= \frac{2n^2 \|\mathbf{X}(0)\| C}{y_o} \left[\lambda^k + \sum_{s=0}^{k-1} \theta(s) \lambda^{k-s-1} + \frac{1}{C} \theta(k) \right].
\end{aligned}$$

□

Corollary 1. *Let Assumption 1 hold. Assume the edges in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ are **not** corrupted by additive noise in communication channels. Let $\{x_i(k), y_i(k), z_i(k)\}_{k \geq 0}^{i \in [\mathbf{N}]}$ be the sequences generated by the iterations (2.12)-(2.14). Let $\theta(k) = \rho^k$, for all $k \geq 1$ such that*

$\rho \in (0, 1/\alpha)$, where $\alpha > 1$. Then, for all $j \in [\mathbf{N}]$ and $k \geq 1$,

$$\|z_j(k) - \bar{u}\| \leq \Upsilon \varrho^k,$$

where, Υ and $\varrho \in (0, 1)$ are positive constants.

Proof. Let $\varrho := \max\{\lambda, \alpha\rho\}$. From Theorem 3,

$$\begin{aligned} \|z_j(k) - \bar{u}\| &\leq C_1 \left[\lambda^k + \sum_{s=0}^{k-1} \theta(s) \lambda^{k-s-1} + \frac{1}{C} \theta(k) \right] \\ &\leq C_1 \left[\lambda^k + \sum_{s=0}^{k-1} \lambda^{k-s-1} \rho^s + \frac{1}{C} \rho^k \right] \\ &\leq C_1 \left[\lambda^k + \sum_{s=0}^{k-1} \varrho^{k-s-1} \rho^s + \frac{1}{C} \rho^k \right] \\ &\leq C_1 \left[\varrho^k + \frac{\varrho^k}{\varrho} \sum_{s=0}^{k-1} \left(\frac{\rho}{\varrho} \right)^s + \frac{1}{C} \varrho^k \right] \\ &\leq C_1 \left[1 + \frac{1}{\varrho} \sum_{s=0}^{\infty} \left(\frac{\rho}{\varrho} \right)^s + \frac{1}{C} \right] \varrho^k \\ &= C_1 \left[1 + \frac{1}{\varrho - \rho} + \frac{1}{C} \right] \varrho^k = \Upsilon \varrho^k, \end{aligned}$$

where, $\Upsilon := C_1 \left[1 + \frac{1}{\varrho - \rho} + \frac{1}{C} \right]$. □

B. Convergence under noisy communication channels:

Here, we analyze the performance of the proposed **NR-PushSum** algorithm under the communication noise. We first introduce conditions on the additive noise in the communication links. In particular, we make the following assumption:

Assumption 4. *The noise in communication channels in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is bounded, i.e. there exists $\delta_{i,j}$ such that $|\eta_{x_{ij}}(k)| \leq \delta_{i,j}$, $|\eta_{y_{ij}}(k)| \leq \delta_{i,j}$ for $i, j \in [\mathbf{N}]$ and $k \geq 1$.*

Consider a family Θ of sequences with any element $\theta(k) \in \Theta$ such that,

$$\theta(k) = \begin{cases} d_1 n \delta \beta(k) \in \mathbb{R}, & k \leq K_\theta, \\ \frac{d_2 n b \delta}{k^q} \in [0, 1], & k > K_\theta, \end{cases} \quad (2.46)$$

where, $1 \leq d_1 < \infty$ and $1 \leq d_2 \leq K_\theta^q/nb\delta$ and $K_\theta \geq \max\{K_\beta, (n\delta b)^{1/q}\}$, where K_β is the time index in Assumption 2, such that $ak^{-q} \leq \beta(k) \leq bk^{-q}$ for all $k \geq K_\beta$, with $q > 1$ and $0 \leq a \leq b < \infty$. Different choices of d_1 and d_2 lead to different sequences in Θ . Note that every sequence $\theta(k)$ in the family Θ satisfies Assumption 3. We have the following result for any sequence $\theta(k) \in \Theta$.

Lemma 3. *Let $\{\theta(k)\}_{k \geq 0} \in \Theta$, then for all $k \geq 0$, $(\theta(k)\mathbf{y}(0) + \beta(k)\eta_y(k)) > 0$.*

Proof. Let $k \leq K_\theta$,

$$\begin{aligned} \theta(k)\mathbf{y}(0) + \beta(k)\eta_y(k) &= d_1 n \delta \beta(k) \mathbf{y}(0) + \beta(k)\eta_y(k) \geq d_1 n \delta \beta(k) \mathbf{1}^\top - \delta \beta(k) \mathbf{1}^\top \\ &= (d_1 n - 1) \delta \beta(k) \mathbf{1}^\top > 0. \end{aligned}$$

For $k > K_\theta$,

$$\begin{aligned} \theta(k)\mathbf{y}(0) + \beta(k)\eta_y(k) &= \frac{d_2 n b \delta}{k^q} \mathbf{y}(0) + \beta(k)\eta_y(k) \geq \frac{d_2 n b \delta}{k^q} \mathbf{1}^\top - \frac{b \delta}{k^q} \mathbf{1}^\top = (d_2 n - 1) \frac{b \delta}{k^q} \mathbf{1}^\top \\ &> 0. \end{aligned}$$

This completes the proof. $\square$

Lemma 4. *Let Assumption 1 hold. Let sequence $\{\beta(k)\}_{k \geq 0}$ satisfy Assumption 2. Assume that the channel noise satisfies Assumption 4. Let $\{\theta(k)\}_{k \geq 0} \in \Theta$ (defined in (2.46)), then there exist $\tilde{y}_o > 0$ such that $y_i(k) > \tilde{y}_o$ for all $k \geq 1$ and $i \in [\mathbf{N}]$, i.e., the estimate $y_i(k)$ remain positive at all time-instants and the states $z_i(k)$ are well-defined for all $k, i \in [\mathbf{N}]$.*

Proof. Let $\delta := \max_{i,j} \delta_{ij}$. Using (2.18) and proceeding as in (2.27) we have,

$$\begin{aligned} \mathbf{y}(k+1) &= W(0, k) \mathbf{y}(0) + \sum_{s=0}^k \theta(s) W(s+1, k) \mathbf{y}(0) + \sum_{s=0}^k \beta(s) W(s+1, k) \eta_y(s) \quad (2.47) \\ &= W(0, k) \mathbf{y}(0) + \sum_{s=0}^k W(s+1, k) (\theta(s) \mathbf{y}(0) + \beta(s) \eta_y(s)). \end{aligned}$$

Let the sequence $\theta(k) \in \Theta$ as defined in (2.46). We follow the same steps in the proof

of Lemma 2. Given, $\varepsilon > 0$, for any $i \in [\mathbf{N}]$ and $k > \ell_\varepsilon$,

$$\begin{aligned}
y_i(k+1) &= \sum_{j=1}^n W_{ij}(0, k) y_j(0) + \sum_{j=1}^n \sum_{s=0}^k W_{ij}(s+1, k) (\theta(s) y_j(0) + \beta(s) \eta_{y_j}(s)) \\
&> \sum_{j=1}^n W_{ij}(0, k) \\
&= \sum_{j=1}^n [\mathbf{v}(\ell_\varepsilon) \mathbf{1}^\top + E(\ell_\varepsilon) W(\ell_\varepsilon + 1, k)]_{ij} \\
&> \sum_{j=1}^n [\mathbf{v}(\ell_\varepsilon) \mathbf{1}^\top - \frac{\varepsilon}{2} \mathbf{1} \mathbf{1}^\top W(\ell_\varepsilon + 1, k)]_{ij} \\
&> \sum_{j=1}^n [\mathbf{v}(\ell_\varepsilon) \mathbf{1}^\top - \frac{\varepsilon}{2} \mathbf{1} \mathbf{1}^\top]_{ij} \\
&= n \mathbf{v}_i(\ell_\varepsilon) - n \frac{\varepsilon}{2} > n \min_{1 \leq i \leq n} \mathbf{v}_i(\ell_\varepsilon) - \varepsilon \frac{n}{2} > 0,
\end{aligned}$$

where, we used (2.25), the result of Lemma 3 and the fact that $W(l, k)$ is column stochastic for all $k, l \geq 0$.

For $k \leq \ell_\varepsilon$, from (2.25) and Lemma 3 we have,

$$y_i(k) > w_o^{k-1} > \theta(0) w_o^{\ell_\varepsilon} > 0.$$

Therefore, $y_i(k) > \tilde{y}_o := \min\{n \min_{1 \leq i \leq n} \mathbf{v}_i(\ell_\varepsilon), \theta(0) w_o^{\ell_\varepsilon}\}$ for all $k \geq 1$ with any $\theta(k) \in \Theta$ and $\beta(k)$ satisfying Assumption 2. $\square$

Theorem 4. *Let Assumption 1 hold. Let sequence $\{\beta(k)\}_{k \geq 0}$ satisfy Assumption 2. Assume that the channel noise satisfies Assumption 4. Let $\{z_i(k)\}_{k \geq 0}^{i \in [\mathbf{N}]}$ be the sequences generated by iteration (2.14). Given $\mu > 0$, let $\{\theta(k)\}_{k \geq 0} \in \Theta$ (defined in (2.46)) with $\theta(0) := n\delta \sum_{s=0}^{\infty} \beta(s)/\mu$, where, $\delta = \max_{i,j} \delta_{ij}$. Then, for all $i \in [\mathbf{N}]$, almost surely, $z_i(k)$ converge to the same vector z_∞ . In particular,*

$$\frac{\bar{u} - \mu}{1 + \mu} \leq \lim_{k \rightarrow \infty} z_i(k) \leq \frac{\bar{u} + \mu}{1 - \mu}.$$

Proof. Recall (2.17) and proceed as in (2.31),

$$\mathbf{X}(k+1) = \underbrace{W(0, k)\mathbf{X}(0)}_{\text{first}} + \underbrace{\sum_{s=0}^k \theta(s)W(s+1, k)\mathbf{X}(0)}_{\text{second}} + \underbrace{\sum_{s=0}^k \beta(s)W(s+1, k)\eta_x(s)}_{\text{third}} \quad (2.48)$$

Note as $W(l, k)$ is column stochastic for all $k, l \geq 1$, $\|W(l, k)\| \leq \bar{w} < \infty$ for all k, l . Under Assumption 3 and from Theorem 1, given $\varepsilon > 0$. Define $\hat{\varepsilon} := \frac{\varepsilon / \sum_{s=0}^{\infty} \theta(s)}{\max\{\|\mathbf{X}(0)\|, n\}}$. There exists $\tilde{k}$ and $\mathbf{v}(k) \in \mathbb{R}^n$ such that $\sum_{s=\tilde{k}}^{\infty} \theta(s) < \frac{\hat{\varepsilon}}{3\bar{w}}$, $\sum_{s=\tilde{k}}^{\infty} \beta(s) \leq \frac{\hat{\varepsilon}}{3n\delta\bar{w}}$ and $W(l, k) = \mathbf{v}(k)\mathbf{1}^\top + E(k)$, for all $k > \tilde{k} + l$, where, $|E_{ij}(k)| \leq \frac{\hat{\varepsilon}}{3}$ for all $i, j \in [\mathbf{N}]$.

Thus, first and second terms in (2.48) can be re-formulated in the same manner as (2.32)-(2.35). Consider, the third term in (2.48),

$$\sum_{s=0}^k \beta(s)W(s+1, k)\eta_x(s) = \underbrace{\sum_{s=0}^{\tilde{k}} \beta(s)W(s+1, k)\eta_x(s)}_{:=\zeta_1(k)} + \underbrace{\sum_{s=\tilde{k}+1}^k \beta(s)W(s+1, k)\eta_x(s)}_{:=\zeta_x(k)} \quad (2.49)$$

Note, $k > \tilde{k} + s + 1$ for all the terms in $\zeta_1(k)$. Substituting $W(s+1, k) := \mathbf{v}(k)\mathbf{1}^\top + \tilde{E}(k)$, with $|\tilde{E}_{ij}(k)| < \hat{\varepsilon}/3$ for all $i, j \in [\mathbf{N}]$ we get,

$$\zeta_1(k) = \sum_{s=0}^{\tilde{k}} \beta(s)[\mathbf{v}(k)\mathbf{1}^\top + \tilde{E}(k)]\eta_x(s). \quad (2.50)$$

Now consider $\zeta_x(k)$,

$$\begin{aligned} \zeta_x(k) &= \sum_{s=\tilde{k}+1}^k \beta(s)W(s+1, k)\eta_x(s) \\ &= \beta(k)\eta_x(k) + \beta(k-1)\mathbf{M}(k)\eta_x(k-1) + \cdots + \beta(\tilde{k}+1)W(\tilde{k}+2, k)\eta_x(\tilde{k}+1). \end{aligned}$$

Therefore,

$$\|\zeta_x(k)\| \leq n\delta\beta(k) + n\delta\bar{w}\beta(k-1) \cdots + n\delta\bar{w}\beta(\tilde{k}+1) \leq n\delta\bar{w} \sum_{s=\tilde{k}}^{\infty} \beta(s) \leq \frac{\hat{\varepsilon}}{3}. \quad (2.51)$$

Using (2.48), (2.32)-(2.35), and (2.49)-(2.51), for all $k > \tilde{k}$,

$$\mathbf{X}(k+1) \leq \underbrace{\mathbf{v}(k)\mathbf{1}^\top(\mathbf{X}(0) + \sum_{s=0}^{\tilde{k}} \theta(s)\mathbf{X}(0) + N_x(\tilde{k}))}_{:=\chi_x(k)} + \varepsilon\mathbf{1}_n\mathbf{1}_p^\top \quad (2.52)$$

$$\mathbf{X}(k+1) \geq \mathbf{v}(k)\mathbf{1}^\top(\mathbf{X}(0) + \sum_{s=0}^{\tilde{k}} \theta(s)\mathbf{X}(0) + N_x(\tilde{k})) - \varepsilon\mathbf{1}_n\mathbf{1}_p^\top \quad (2.53)$$

where, $N_x(\tilde{k}) := \sum_{s=0}^{\tilde{k}} \beta(s)\eta_x(s)$, $N_y(\tilde{k}) := \sum_{s=0}^{\tilde{k}} \beta(s)\eta_y(s)$ and $\mathbf{1}_n$ and $\mathbf{1}_p$ are the vectors of dimensions n and p with all entries equal to 1. We drop the subscript when the dimension of the vector is clear from the context. Following the similar procedure for (2.47) we get, for $k \geq 1$,

$$\mathbf{y}(k+1) \leq \mathbf{v}(k)\mathbf{1}^\top \chi_y(k) + \varepsilon\mathbf{1}, \quad (2.54)$$

$$\mathbf{y}(k+1) \geq \mathbf{v}(k)\mathbf{1}^\top \chi_y(k) - \varepsilon\mathbf{1}. \quad (2.55)$$

From (2.19), and (2.52)-(2.55),

$$\frac{[\mathbf{v}(k)\mathbf{1}^\top \chi_x(k) - \varepsilon\mathbf{1}_n\mathbf{1}_p^\top]_i}{[\mathbf{v}(k)\mathbf{1}^\top \chi_y(k) + \varepsilon\mathbf{1}]_i} \leq z_i(k+1) \leq \frac{[\mathbf{v}(k)\mathbf{1}^\top \chi_x(k) + \varepsilon\mathbf{1}_n\mathbf{1}_p^\top]_i}{[\mathbf{v}(k)\mathbf{1}^\top \chi_y(k) - \varepsilon\mathbf{1}]_i}, \quad (2.56)$$

where, $[\cdot]_i$ denote the i^{th} row (i^{th} entry) of the corresponding matrices (vectors). Denote $\bar{\theta} := 1 + \sum_{s=0}^{\tilde{k}} \theta(s)$, and $\mathbf{v}_i(k)$ as the i^{th} entry of $\mathbf{v}(k)$.

For $k > \tilde{k}$, for any $i \in [\mathbf{N}]$, from (2.56),

$$\begin{aligned} z_i(k+1) &\leq \frac{[\mathbf{v}(k)\mathbf{1}^\top \chi_x(k) + \varepsilon\mathbf{1}_n\mathbf{1}_p^\top]_i}{[\mathbf{v}(k)\mathbf{1}^\top \chi_y(k) - \varepsilon\mathbf{1}]_i} \\ &\leq \frac{\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{X}(0) + \bar{\theta}\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{X}(0) + \mathbf{v}_i(k)\mathbf{1}^\top N_x(\tilde{k}) + \varepsilon\mathbf{1}^\top}{\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{y}(0) + \bar{\theta}\mathbf{v}_i(k)\mathbf{1}^\top \mathbf{y}(0) + \mathbf{v}_i(k)\mathbf{1}^\top N_y(\tilde{k}) - \varepsilon} \\ &\leq \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\bar{\theta}\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\mathbf{1}^\top N_x(\tilde{k})}{n} + \varepsilon\mathbf{1}^\top}{1 + \bar{\theta} + \frac{\mathbf{1}^\top N_y(\tilde{k})}{n} - \varepsilon} \\ &\leq \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} (1 + \bar{\theta}) + \frac{\mathbf{1}^\top N_x(\tilde{k})}{n} + \varepsilon\mathbf{1}^\top}{1 + \bar{\theta} + \frac{\mathbf{1}^\top N_y(\tilde{k})}{n} - \varepsilon} \end{aligned}$$

$$\begin{aligned}
&\leq \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\mathbf{1}^\top N_x(\tilde{k})}{n(1+\bar{\theta})} + \frac{\varepsilon}{(1+\bar{\theta})} \mathbf{1}^\top}{1 + \frac{\mathbf{1}^\top N_y(\tilde{k})}{n(1+\bar{\theta})} - \frac{\varepsilon}{(1+\bar{\theta})}} \\
&\leq \frac{a}{b} + \left(\frac{a}{b} + \mathbf{1}^\top\right) \left(\frac{\frac{\varepsilon}{b(1+\bar{\theta})}}{1 + \frac{\varepsilon}{b(1+\bar{\theta})}}\right),
\end{aligned}$$

where, $a := \frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\mathbf{1}^\top N_x(\tilde{k})}{n(1+\bar{\theta})}$, $b := 1 + \frac{\mathbf{1}^\top N_y(\tilde{k})}{n(1+\bar{\theta})}$. Similarly, for $k > \tilde{k}$, for any $i \in [\mathbf{N}]$,

$$z_i(k+1) \geq \frac{a}{b} - \left(\frac{a}{b} + \mathbf{1}^\top\right) \left(\frac{\frac{\varepsilon}{b(1+\bar{\theta})}}{1 - \frac{\varepsilon}{b(1+\bar{\theta})}}\right).$$

Let $\bar{\varepsilon}$ be such that, $\varepsilon \leq \frac{\bar{\varepsilon}}{b(1+\bar{\varepsilon})}(1+\bar{\theta})$. Therefore, for all $k > \tilde{k}$, and for all $i \in [\mathbf{N}]$,

$$-\left(\frac{a}{b} + \mathbf{1}^\top\right) \bar{\varepsilon} \leq z_i(k) - \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\mathbf{1}^\top N_x(\tilde{k})}{n(1+\bar{\theta})}}{1 + \frac{\mathbf{1}^\top N_y(\tilde{k})}{n(1+\bar{\theta})}} \leq \left(\frac{a}{b} + \mathbf{1}^\top\right) \bar{\varepsilon}.$$

Note that $\bar{\varepsilon}$ can be made arbitrarily small. Thus, for all $i \in [\mathbf{N}]$,

$$\lim_{k \rightarrow \infty} z_i(k) = \frac{\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} + \frac{\mathbf{1}^\top N_x(\tilde{k})}{n(1+\bar{\theta})}}{1 + \frac{\mathbf{1}^\top N_y(\tilde{k})}{n(1+\bar{\theta})}} := z_\infty.$$

Moreover, under assumptions 2–4,

$$\frac{\bar{u} - \frac{n\delta \sum_{s=0}^{\infty} \beta(s)}{(1+\sum_{s=0}^{\infty} \theta(s))} \mathbf{1}^\top}{1 + \frac{\delta \sum_{s=0}^{\infty} \beta(s)}{(1+\sum_{s=0}^{\infty} \theta(s))}} \leq z_\infty \leq \frac{\bar{u} + \frac{n\delta \sum_{s=0}^{\infty} \beta(s)}{(1+\sum_{s=0}^{\infty} \theta(s))} \mathbf{1}^\top}{1 - \frac{\delta \sum_{s=0}^{\infty} \beta(s)}{(1+\sum_{s=0}^{\infty} \theta(s))}},$$

where, we used $\frac{\mathbf{1}^\top \mathbf{X}(0)}{n} = \bar{u}$. Note that given $\mu > 0$, if $\theta(0) := n\delta \sum_{s=0}^{\infty} \beta(s)/\mu$, we have

$$\frac{\bar{u} - \mu}{1 + \mu} \leq z_\infty \leq \frac{\bar{u} + \mu}{1 - \mu}.$$

□

Remark 2. The upper and lower bounds on z_∞ in Theorem 4 can be made very close to the average $\bar{u}$ by appropriately choosing the sequences $\theta(k)$ and $\beta(k)$. In particular,

if $\sum_{s=0}^{\infty} \theta(s) \gg \sum_{s=0}^{\infty} \beta(s)$ then z_{∞} is very close to the actual average $\bar{u}$. This can be achieved via suitable choices of K_{θ} and the starting values of the sequence $\theta(k), k < K_{\theta}$ in (2.46). We demonstrate this using numerical simulation studies with different noise realizations in Section 2.5.1.

Remark 3. Unlike the state-of-the-art in [75], [81], [84], [90], [93], the **NR-PushSum** algorithm remains applicable even for noise distributions that are not necessarily zero-mean and independent. Furthermore, it achieves almost sure average consensus as shown in Theorem 4 and substantiated through simulation results in Section 2.5.1.

Moreover, the articles [75], [84], [90], [93] are confined to undirected graphs with agents connected via bidirectional edges, in a symmetric communication topology. Additionally, these articles follow a centralized synthesis approach, utilizing global information of the interconnection structure. In contrast, the **NR-PushSum** algorithm operates differently. Notably, the consensus protocol presented in [81] diverges from the average of the initial values of the agents, setting it apart from the **NR-PushSum** algorithm.

The next result establishes the existence of the moments of the estimates $z_i(k)$ generated by the **NR-PushSum** algorithm.

Theorem 5. Let Assumption 1 hold. Let sequence $\{\beta(k)\}_{k \geq 0}$ satisfy Assumption 2. Assume that the channel noise satisfies Assumption 4. Let $\{\theta(k)\}_{k \geq 0} \in \Theta$ (defined in (2.46)). Let $\{x_i(k), y_i(k), z_i(k)\}_{k \geq 0}^{i \in [\mathbf{N}]}$ be the sequences generated by the iterations (2.12)-(2.14). Then, for any m non-negative, $\mathbb{E}(|z_{ij}(k)|^m) < \infty$, for all $i \in [\mathbf{N}], j = 1, \dots, p$ and $k \geq 0$.

Proof. From (2.48) and (2.52), under Assumptions 2-4 there exists $x_{up} < \infty$ such that $|\mathbf{X}_{ij}(k)| \leq x_{up}$ for all $i \in [\mathbf{N}], j = 1, \dots, p$ and $k \geq 0$. Thus, for any m non-negative, $\mathbb{E}(|\mathbf{X}_{ij}(k)|^m) = x_{up}^m < \infty$. From Lemma 4, there exists $\tilde{y}_o$ such that we have for all $i \in [\mathbf{N}]$ and $k \geq 0$, **Probability** $(|y_i(k)| < \tilde{y}_o) = 0$. Using Hölder's inequality for $(\mathbf{X}_{ij}(k))^m$ and $(y_i(k))^{-m}$. Since, **Probability** $((\tilde{y}_o)^{-m} < (y_i(k))^{-m} < (\tilde{y}_o)^{-m}) = 1$, we have

$$\mathbb{E}(|z_{ij}(k)|^m) \leq \tilde{y}_o^{-m} \mathbb{E}(|\mathbf{X}_{ij}(k)|^m) = \tilde{y}_o^{-m} x_{up}^m < \infty, \quad (2.57)$$

for all k , for any $i \in [\mathbf{N}]$ and $j = 1, \dots, p$. $\square$

Remark 4. As a consequence of Theorem 5, it follows that the variance $\mathbf{Var}(z_i(k))$ of the estimates $z_i(k)$, $\mathbf{Var}(z_i(k)) = \mathbb{E}(|z_i(k)|^2) - (\mathbb{E}(z_i(k)))^2 \leq 2\sqrt{n}\tilde{y}_o^{-2}x_{up}^2 < \infty$ is bounded.

Simulation Study:

In this section, we demonstrate the performance of the proposed **NR-PushSum** algorithm. A network of 10 agents generated using the Erdos-Renyi model [121] with a connectivity probability of 0.35 is considered for demonstration. The initial values of the agents chosen for the numerical study are $u = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$ with the average value being $\bar{u} = 5.5$. The examples in the numerical study are implemented in MATLAB and run on a desktop computer with 16 GB RAM and an Intel Core i7 processor running at 1.90 GHz. The noise realizations are obtained by simulating random variables drawn independently from a probability distribution at each iteration of the algorithm under study. We present the plots of the evolution of agents' estimates against the algorithm iterations.

First, we present the performance of the proposed **NR-PushSum** algorithm under perfect communication. Then we present the performance under imperfect communication via the following case studies: (i) performance of **NR-PushSum** algorithm under different algorithm parameters, (ii) performance under different magnitudes of the additive noise, and (iii) robustness of the **NR-PushSum** algorithm under the violation of the upper bound in Assumption 4 during the run-time of the algorithm. Lastly, we present a comparison between the proposed **NR-PushSum** algorithm and other existing algorithms in the literature. Consensus error for the **NR-PushSum** algorithm at any iteration k is defined as,

$$\sum_{i=1}^n (z_i(k) - \bar{u})^2 = \sum_{i=1}^n (z_i(k) - 5.5)^2.$$

A. Performance under perfect communication:

Figure 2.4 presents the agent state trajectories for the **NR-PushSum** algorithm when the communication links are noiseless. The algorithm parameters are: $\theta(k) = 0.7^k$ for all $k \geq 1$ and $K_\beta = 200$ such that $\beta(k) = 0.35, \forall k < 200$ and $\beta(k) = 100/k^{1.1}, \forall k \geq 200$. Figs. 2.4(a) and 2.4(b) show the estimates $x_i(k)$ and $y_i(k)$ (called the numerator and denominator states respectively) for all the agents. The numerator and denominator states reach a steady state. The agent states $z_i(k)$ are plotted in Figure 2.4(c); it can be seen that the agents' states $z_i(k)$ achieve consensus to the average value of 5.5. Figure 2.4(d) shows the decay of consensus error with respect to the number of iterations of the **NR-PushSum** algorithm.

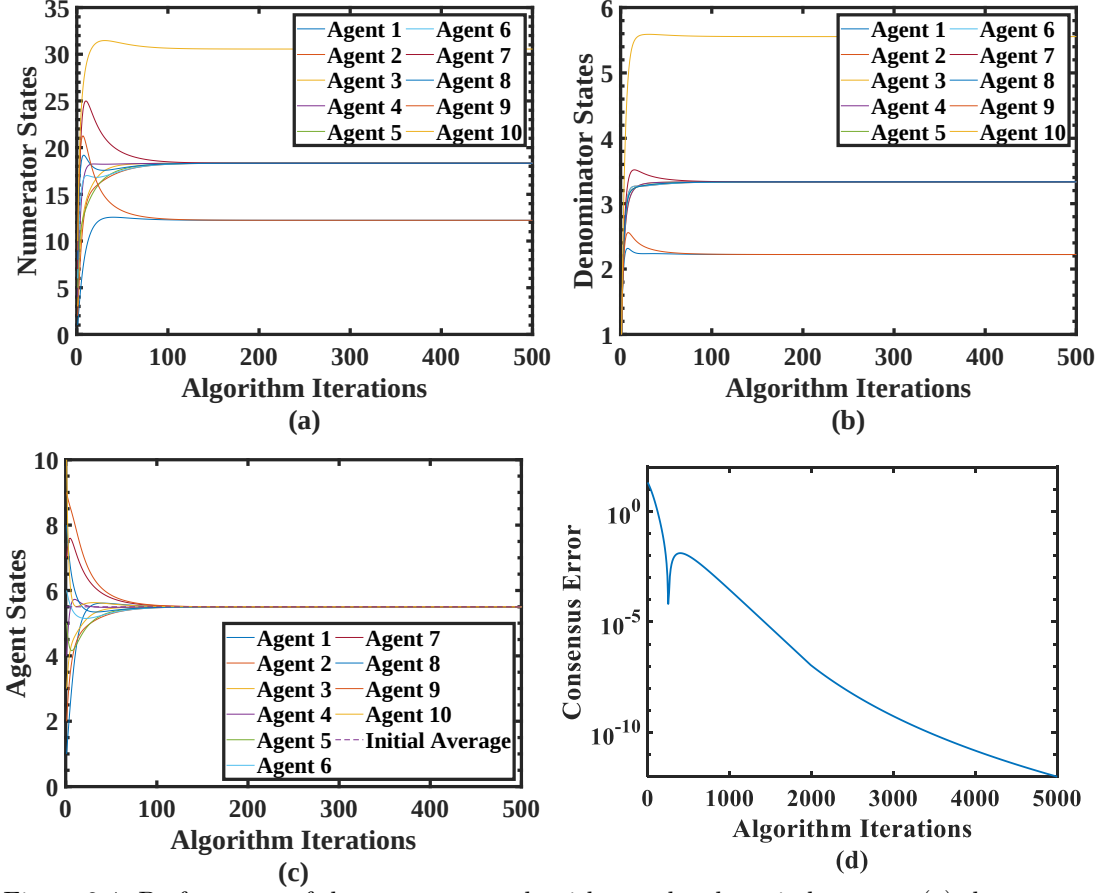


Figure 2.4: Performance of the **NR-PushSum** algorithm under the noiseless case: (a) the sequence of states $x_i(k)$, (b) $y_i(k)$ sequences, (c) ratio states of the 10 agents achieve consensus, (d) geometric decay of the consensus error.

B. Performance under communication noise:

In this section, we present the performance of the proposed **NR-PushSum** algorithm under noisy communication links.

1. Different choices of algorithm parameters: Here, we demonstrate that the **NR-PushSum** algorithm achieves average consensus under the additive noise. The noise in communication links is realized from a uniform distribution $\mathcal{U}(-1, 1)$. We employ two different choices of $\beta(k)$ sequences and accordingly choose the $\theta(k)$ sequences from the family of sequences in (2.46). The choices are summarized in Table 2.1. Figs. 2.5(a) and 2.5(b) present the agent state trajectories for the **NR-PushSum** algorithm realizations with sequences in Table 2.1 rows (i) and (ii) respectively under the additive noise. The inset in Figs. 2.5(a) and 2.5(b) show that the proposed **NR-PushSum** algorithm has strong

Table 2.1: Example Sequences for **NR-PushSum**

(i)	$\beta(k) = 0.2, \forall k < 500,$ $\beta(k) = 1/k^{1.5}, \forall k \geq 500$	$\theta(k) = 100, \forall k < 500,$ $\theta(k) = 10/k^{1.5}, \forall k \geq 500$
(ii)	$\beta(k) = 0.35, \forall k < 200,$ $\beta(k) = 100/k^{1.1}, \forall k \geq 200$	$\theta(k) = 100, \forall k < 200,$ $\theta(k) = 150/k^{1.1}, \forall k \geq 200$

performance leading to consensus among the agents under additive noise. Moreover, the final consensus value of the agents is close to the initial average of 5.5. Figs. 2.5(c) and (d) show the consensus error with the parameters in Table 2.1 for rows (i) and (ii), respectively. The sequence $\theta(k)$ belongs to the family of sequences given by (2.46). Here, the **NR-PushSum** algorithm achieves a sub-linear decay of consensus error, analogous to Theorem 3.

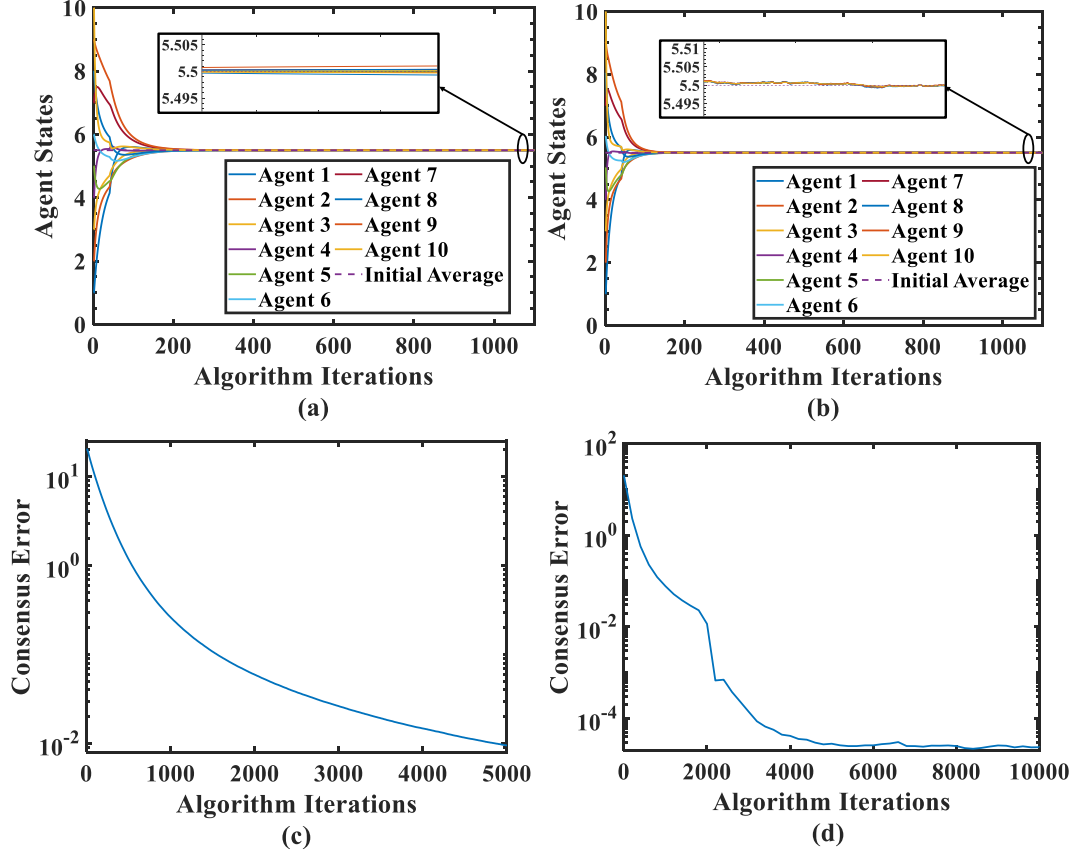


Figure 2.5: Performance of the **NR-PushSum** algorithm under noisy communication with different choices of algorithm parameters: (a) Row (i) of Table 2.1, (b) Row (ii) of Table 2.1, (c) Consensus error for parameters in Row (i) of Table 2.1, (d) Consensus error for parameters in Row (ii) of Table 2.1.

2. Different magnitudes of the additive noise: The effect of the magnitude of the communication noise relative to the initial values of the agents is presented here. The noise is realized as a uniform random variable. For the numerical experiment, the magnitude of the noise realizations is scaled to (i) 5 times the original value, i.e. $\mathcal{U}(-5, 5)$ and (ii) 10 times the original value i.e. $\mathcal{U}(-10, 10)$. We choose the sequence $\beta(k) = 0.35, \forall k < 200, \beta(k) = 1/k^{1.2}, \forall k \geq 200$. The sequence $\theta(k)$ is modified according to the noise upper bound. For the two cases, the following $\theta(k)$ sequences are chosen in accordance to the family (2.46) for the **NR-PushSum** algorithm: (i) $\theta(k) = 100, \forall k < 200, \theta(k) = 50/k^{1.2}, \forall k \geq 200$ and (ii) $\theta(k) = 100, \forall k < 200, \theta(k) = 100/k^{1.2}, \forall k \geq 200$. Figs. 2.6(a) and 2.6(b) provide the state trajectories of different agents under the **NR-PushSum** algorithm for cases (i) and (ii) respectively. It can be seen that the agent states achieve consensus close to the initial average value of 5.5. Figs. 2.6(c) and 2.6(d) plot the consensus error decay under the noise realizations via $\mathcal{U}(-5, 5)$ and $\mathcal{U}(-10, 10)$, respectively.

3. Robustness to noise with extreme magnitudes: The parameters of the proposed **NR-PushSum** algorithm depend on an upper bound on the communication noise (see (2.46)). To test the applicability of the **NR-PushSum** algorithm in extreme scenarios, here we test the robustness of the **NR-PushSum** algorithm under situations when the communication noise violates the upper bound in Assumption 4. In particular, we design the algorithm parameters assuming that the communication noise is a uniform random variable $\mathcal{U}(-1, 1)$. We test the following two scenarios: (i) communication noise in the worst case becomes 400 times at every 50 iterations, i.e. noise is realized as $\mathcal{U}(-400, 400)$ at every iteration of the form $50i$, where, $i = 1, 2, \dots$, (ii) communication noise in the worst case becomes 400 times at every 10 iterations, i.e. noise is realized as $\mathcal{U}(-400, 400)$ at every iteration of the form $10i$, where, $i = 1, 2, \dots$. The two situations are designed to test the effect of the number of violations of the bound in Assumption 4 during run-time on the convergence of the **NR-PushSum** algorithm. Figs. 2.7(a) and 2.7(b) show the agent states for the two test situations. The frequency of the event when the noise value is greater than the original upper bound affects the final converged value of the algorithm. In case (i) (Figure 2.7(a)) with less number of violations the **NR-PushSum** algorithm converges closer to the initial average value of 5.5, while the estimates in the situation (ii) have a larger deviation from the original average value of 5.5 (Figure 2.7(b)). However, we emphasize that the proposed **NR-PushSum** algorithm

performs well and remains stable under the extreme violations of the design assumptions as shown by the state trajectories in Figs. 2.7(a) and 2.7(b). The consensus error decay for the two test situations is presented in Figs. 2.7(c) and 2.7(d), respectively. Note that the extreme noise magnitudes do affect the consensus error. However, the **NR-PushSum** algorithm can achieve a small consensus error in both test scenarios.

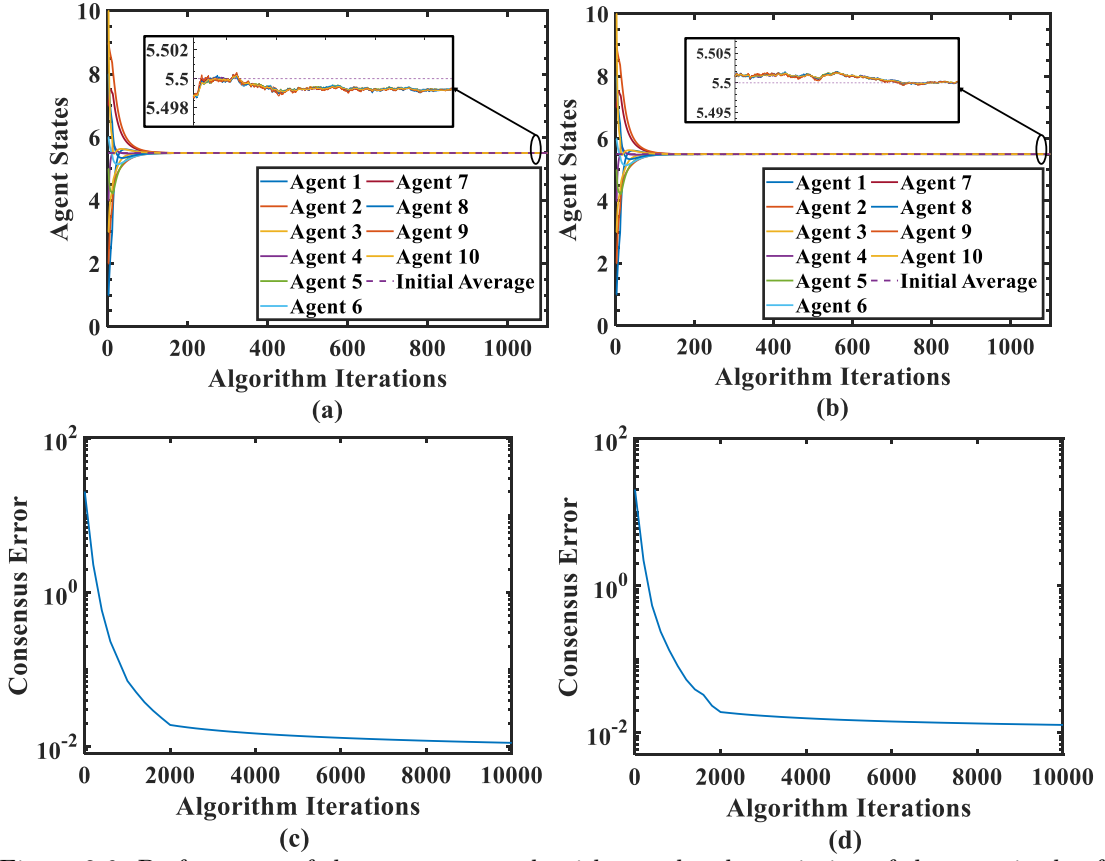


Figure 2.6: Performance of the **NR-PushSum** algorithm under the variation of the magnitude of additive communication noise: (a) noise realized via $\mathcal{U}(-5, 5)$, (b) noise realized via $\mathcal{U}(-10, 10)$. In both cases, the state estimates under **NR-PushSum** with appropriate algorithmic parameters converge to the initial average, (c) Consensus error under the noise realization via $\mathcal{U}(-5, 5)$, (d) Consensus error under the noise realization via $\mathcal{U}(-10, 10)$.

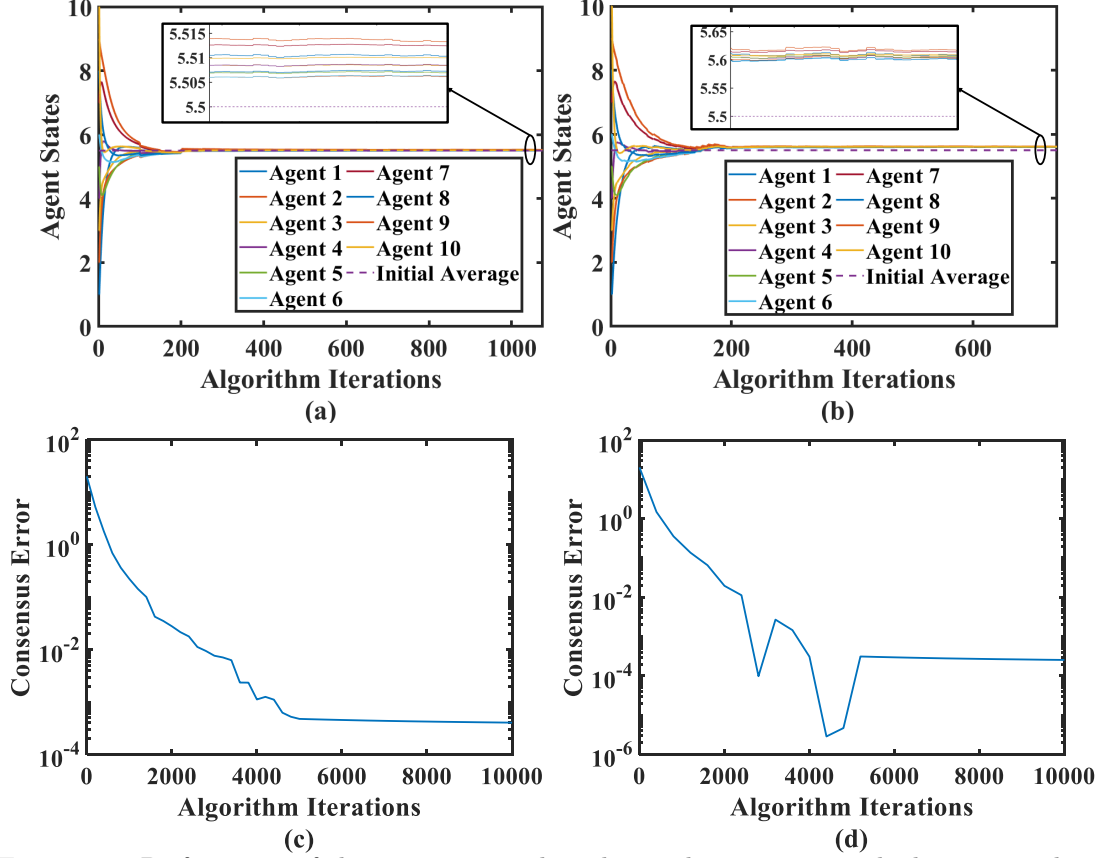


Figure 2.7: Performance of the **NR-PushSum** algorithm under scenarios with the noise realizations violating the upper bound utilized in the algorithm's parameters. Figs. 2.7(a) and 2.7(b) demonstrate the robustness of the proposed **NR-PushSum** algorithm under extreme violation scenarios, Figs. 2.7(c) and 2.7(d) present the consensus error decay during the iterations of the **NR-PushSum** algorithm.

C. Comparison with other algorithms:

In this section, we present a performance comparison between the proposed **NR-PushSum** algorithm and existing algorithms in the literature. Here, we refer to the existing algorithms via the initials of the authors' names. We compare with the following algorithms: WE algorithm of [93], HM algorithm in [81], TN algorithm in [90], HZCSC algorithm in [84], and the KM algorithm in [75]. To obtain the consensus error for any other algorithm we replace the $z_i(k)$ estimates with the corresponding estimates in the chosen algorithm. We compare the results under two different noise realizations: (i) we first consider the communication noise realized as a uniform random variable $\mathcal{U}(-1, 1)$. (ii) Next, we compare the performance of all the algorithms when the communication noise

has a **non-zero** mean. In particular, we consider the noise realizations to be generated as a non-zero mean uniform random variable $\mathcal{U}(0, 1)$. In both cases we choose the following sequences for the **NR-PushSum** algorithm: $\beta(k) = 0.2, \forall k < 500, \beta(k) = 1/k^{1.5}, \forall k \geq 500$ and $\theta(k) = 100, \forall k < 500, \theta(k) = 10/k^{1.5}, \forall k \geq 500$.

To present the performance comparison, we plot the consensus error for all the algorithms with respect to the iteration count under the zero mean and non-zero mean cases in Figure 2.8(a) and 2.8(b) respectively. It can be seen that the proposed **NR-PushSum** algorithm performs the best in both cases. The **NR-PushSum** algorithm has a consensus error that decays faster compared to the other algorithms when the communication noise has zero mean (see Figure 2.8(a)). When the communication noise has a non-zero mean, it can be seen that the consensus error for the existing algorithms in the literature is greater than 1 and increases with the iteration count of the algorithm. In this case, too the proposed **NR-PushSum** algorithm performs well. The consensus error for the **NR-PushSum** algorithm is less than 10^{-2} for all the iterations.

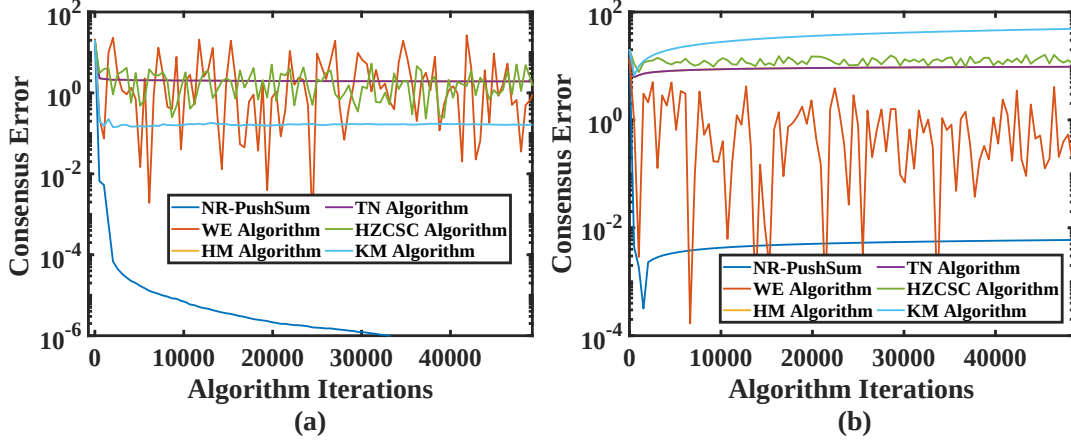


Figure 2.8: Performance comparison of the **NR-PushSum** algorithm with existing algorithms in the literature: (a) performance under bounded noise with zero mean and (b) performance under bounded communication noise with non-zero mean.

Summarizing, the simulation results indicate that the proposed **NR-PushSum** algorithm performs well under communication noise and shows robustness to uncertain changes in communication noise. Compared to the existing state-of-the-art algorithms the **NR-PushSum** algorithm has superior performance concerning the ability to achieve consensus to the initial average under communication noise.

2.5.2 Distributed Finite-time Terminated Consensus Algorithm in Switching Topologies

We first summarize a key result from [43], which enables reaching the average consensus in the presence of dynamic topology. Consider a scenario where the network topology is dynamic but with a fixed set of nodes, $\mathcal{V}$ ($|\mathcal{V}| = n$), i.e., at any given instant, k , the network is described by a digraph, $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$. Let $\mathbf{P}(k) = (p_{ij}(k))$ be the weighted adjacency matrix associated with the digraph, $\mathcal{G}(k)$. Here, $\mathcal{G}(k) \in \bar{\mathcal{G}} := \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_m\}$, $m \leq 2^{n^2-n}$ is the set of all possible digraphs for a given set of nodes, $\mathcal{V}$. We assume that a node always has access to its information. Each node, $i \in \mathcal{V}$, maintains two states at time k , denoted by $x_i(k)$ and $y_i(k)$. Node i updates its state at the $(k+1)^{th}$ iteration according to the following policy:

$$x_i(k+1) = p_{ii}(k)x_i(k) + \sum_{j \in \mathcal{N}_i^-(k)} p_{ij}(k)x_j(k) \quad (2.58)$$

$$y_i(k+1) = p_{ii}(k)y_i(k) + \sum_{j \in \mathcal{N}_i^-(k)} p_{ij}(k)y_j(k), \quad (2.59)$$

where $y_i(0) = 1$ for all $i \in \mathcal{V}$. Using (2.58) and (2.59), we get

$$\frac{x_i(k+1)}{y_i(k+1)} = \frac{p_{ii}(k)x_i(k) + \sum_{j \in \mathcal{N}_i^-(k)} p_{ij}(k)x_j(k)}{p_{ii}(k)y_i(k) + \sum_{j \in \mathcal{N}_i^-(k)} p_{ij}(k)y_j(k)}. \quad (2.60)$$

The network with dynamic topology satisfies the following assumptions. We also assume that the underlying communication network is free from communication delays.

Assumption 5. *The digraphs $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$ are strongly connected for all discrete-time instants $k = 1, 2, \dots$.*

Assumption 6. *$\mathbf{P}(k)$ for all k is a column-stochastic matrix.*

One choice of weights that satisfies Assumption 6 is the *out-degree-based equal neighbor*

weights rule [51]: Here, for all $j \in \mathcal{V}$ and $k \geq 0$:

$$p_{ij}(k) = \begin{cases} \frac{1}{1 + |\mathcal{N}_j^+(k)|}, & \text{if } i \in \mathcal{N}_j^+(k) \\ 0, & \text{if } i \notin \mathcal{N}_j^+(k) \end{cases} \quad (2.61)$$

and,

$$p_{jj}(k) = \frac{1}{1 + |\mathcal{N}_j^+(k)|} \text{ for all } k \geq 0, \quad (2.62)$$

where $|\mathcal{N}_j^+(k)|$ denotes the out-degree of a node, $j \in \mathcal{V}$ at time instant k . Note that because the out-degree is known locally, each agent can decide the weights, $p_{ij}(k)$, in a decentralized manner, leading to a distributed synthesis. Using the equal neighbor weights rule, (2.61)–(2.62), the updates (2.58) and (2.59) are done in the following manner:

Every agent j sends the weighted information, $p_{ij}(k)x_j(k)$ and $p_{ij}(k)y_j(k)$, to every agent i in its out-neighborhood. Then, at the next time instant, agent i scales its estimates, $x_i(k), y_i(k)$, with a weight $p_{ii}(k)$, and adds it to the information sent by all the agents in its in-neighborhood, $j \in \mathcal{N}_i^-(k)$, to get the estimates $x_i(k+1)$ and $y_i(k+1)$. Thus, a distributed implementation of the updates (2.58) and (2.59) is realized. Note that $\mathbf{P}(k)$ synthesized using (2.61) and (2.62) satisfies the primitiveness and irreducibility properties (see [43] Lemma 2). We next present the convergence result from [43], which forms the basis of our work.

Theorem 6 (Theorem 2 in [43]). *Consider a sequence of digraphs of the form $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$, $k = 0, 1, 2, \dots$, satisfying Assumption 5 and Assumption 8. With the update rule (2.58) and (2.59), the ratio $\frac{x_i(k)}{y_i(k)}$ asymptotically converges to $\frac{\sum_{i=1}^n x_i(0)}{n}$ for all $i = 1, \dots, n$, that is, the ratio of the numerator and denominator states converge to the average of the initial conditions of x_i variables (referred to as the ratio consensus).*

Remark 5. Under Assumption 6, $x_i(k)$ and $y_i(k)$ converges to $\gamma_i \sum_{i=1}^n x_i(0)$ and $\gamma_i \sum_{i=1}^n y_i(0)$, respectively, where γ_i is the i^{th} entry of the vector, γ given by $\lim_{k \rightarrow \infty} \prod_{s=1}^k \mathbf{P}(s) = \gamma^\top \mathbf{1}_n$, where $\mathbf{1}_n$ is the n -dimensional vector with all entries equal to one. By having the ratio $\frac{x_i(k)}{y_i(k)}$ and initializing $y_i(0) = 1$ for all $i \in \mathcal{V}$, agent i 's state, $y_i(k)$, converges to $n\gamma_i$, and the ratio $\frac{x_i(k)}{y_i(k)}$ converges to the average of the initial states of x_i 's, $\frac{1}{n} \sum_{i=1}^n x_i(0)$.

Remark 6. Even though Theorem 6 does not require $\mathbf{P}(k)$ to be row-stochastic, Lemma 4.2 in [122] proves that the $x_i(k)$ and $y_i(k)$ following the updates of (2.58) and (2.59) are

almost surely bounded below by some positive constant. Thus, the state updates will not become arbitrarily small, and the ratio state will always be well-defined. Further, due to the weak ergodicity [123] of the products of $\mathbf{P}(k)$'s, along with the bounded initializations $x_i(0)$ and $y_i(0)$, the individual states, $x_i(k)$ and $y_i(k)$, will also remain bounded.

Remark 7. *The discrete-time consensus algorithm is agnostic to the switching frequency of the underlying graph topology. In particular, only the state of the network at the instant of information communication is used. All the constraints on the network are imposed at the time instant of communication, that is, any switching happening between the two discrete-time instants is not considered.*

It was shown in [113], [114] that within $\mathcal{D}_{max}$ iterations, every node has access to global information on the updates, where $\mathcal{D}_{max}$ is an upper bound on the graph diameter. We show that the result from [113], [114] does not hold for a network with dynamic topology. The following counterexample highlights the case where an upper bound on the maximum diameter ($\mathcal{D}_{max} = \{\max_i(\mathcal{D}(\mathcal{G}_i)); \mathcal{G}_i \in \bar{\mathcal{G}}\}$, where $\mathcal{D}(\mathcal{G}_i)$ is the diameter of graph $\mathcal{G}_i$) does not provide access to the global maximum and minimum to all the nodes of the network. Consider a network of six nodes, where the network topology can be either of undirected graphs $\mathcal{G}_1$ or $\mathcal{G}_2$ (see Figure 2.9), that is, $\bar{\mathcal{G}} = \{\mathcal{G}_1, \mathcal{G}_2\}$, with switching as described in Table 2.2. Here, the maximum diameter among the two topologies $\mathcal{D}_{max}$ is 3.

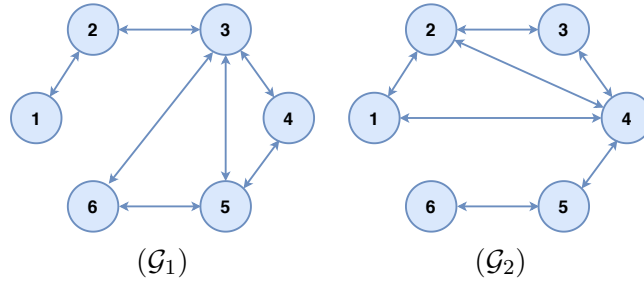


Figure 2.9: Both $\mathcal{G}_1$ and $\mathcal{G}_2$ with $\mathcal{D}_{max} = 3$.

We introduce the following notion,

Definition 29. *(Range of a node) Let us define the range of any node i with initial time, k , and time horizon, $t > 0$ as:*

$$R_i(k, t) := \bigcup_{m \in R_i(k, t-1)} \{l : l \in \mathcal{N}_m^+(k + t - 1)\},$$

with $R_i(k, 0) = \{i\}$, $|R_i(k, 0)| = 1$, and $\mathcal{N}_m^+(k)$ is the out-neighborhood of node m at instant k and $|R_i(k, 0)|$ denotes the cardinality of $|R_i(k, 0)|$.

The range of node 1 starting from any instant k is also shown in Table 2.2.

Iteration	Topology	Range of node 1
$t = 1$	$\mathcal{G}_1$	$R_1(k, 1) = \{1, 2\}$
$t = 2$	$\mathcal{G}_1$	$R_1(k, 1) = \{1, 2, 3\}$
$t = 3$	$\mathcal{G}_2$	$R_1(k, 1) = \{1, 2, 3, 4\}$
$t = 4$	$\mathcal{G}_2$	$R_1(k, 1) = \{1, 2, 3, 4, 5\}$
$t = 5$	$\mathcal{G}_2$	$R_1(k, 1) = \{1, 2, 3, 4, 5, 6\}$

Table 2.2: Network Topology and Range ($R_1(k, t)$) of Node 1 Starting at Instant k for Iterations $t = 1, 2, 3, 4, 5$.

It takes five iterations for the range to contain all the nodes, which is more than the maximum diameter ($\mathcal{D}_{max} = 3$) seen among the set of topologies $\bar{\mathcal{G}} = \{\mathcal{G}_1, \mathcal{G}_2\}$. Therefore, the criteria proposed in [113], [114] has the following two issues:

1. Lemma 2.2 in [114] and Theorem 4.1 in [113] (strict monotonicity of global maximum/minimum states implementing the average consensus protocols) assume that the global maximum/minimum are available to each agent within $\mathcal{D}_{max}$ iterations, which is deemed untrue for this example (Table 2.2).
2. Thus, nodes implementing Algorithm 1 of [113], [114] do not have access to the global maximum/minimum and might terminate the algorithm (when the *local* maximum and minimum are within the tolerance) before actual global convergence (when the *global* maximum and minimum are the within tolerance).

Distributed Finite-Time Termination Under Dynamic Topologies:

Consider the global maximum and minimum values of the ratio of the consensus protocol states given by (2.58) and (2.59) at any time instant k ,

$$M(k) := \max_{i \in \mathcal{V}} \frac{x_i(k)}{y_i(k)}, y_j(k) \neq 0, j \in \mathcal{V}, \quad (2.63)$$

$$m(k) := \min_{i \in \mathcal{V}} \frac{x_i(k)}{y_i(k)}, y_j(k) \neq 0, j \in \mathcal{V} \quad (2.64)$$

The following Lemma shows that the ratio of states at each node stays within the maximum and minimum for subsequent iterations.

Lemma 5. Consider the ratio consensus protocol comprising two updates, given by (2.58) and (2.59). Let Assumption 5 and Assumption 6 hold. Then for all time instants $k' \geq k$ and for all $i \in \mathcal{V}$:

$$m(k) \leq \frac{x_i(k')}{y_i(k')} \leq M(k).$$

Proof. We first prove the inequality for $M(k)$ using induction. By definition of $M(k)$, for $k' = k$, the proof is trivial. Suppose it is asserted that for $k' = k + l, l \geq 1$, $\frac{x_i(k+l)}{y_i(k+l)} \leq M(k)$ for all $i \in \mathcal{V}$. Then we have:

$$\begin{aligned} \frac{x_i(k+l+1)}{y_i(k+l+1)} &= \frac{\sum_{j \in \mathcal{N}_i^-(k+l)} p_{ij}(k+l)x_j(k+l)}{\sum_{j \in \mathcal{N}_i^-(k+l)} p_{ij}(k+l)y_j(k+l)} \\ &= \frac{p_{ii}(k+l)\frac{x_i(k+l)}{y_i(k+l)} + \sum_{j \in \mathcal{N}_i^-(k+l)} p_{ij}(k+l)\frac{x_j(k+l)}{y_i(k+l)}}{p_{ii}(k+l) + \sum_{j \in \mathcal{N}_i^-(k+l)} p_{ij}(k+l)\frac{y_j(k+l)}{y_i(k+l)}}. \end{aligned}$$

It follows from the inductive assumption that:

$$\frac{x_i(k+l+1)}{y_i(k+l+1)} \leq \frac{p_{ii}(k+l)M(k) + \sum_{j \in \mathcal{N}_i^-(k+l)} p_{ij}(k+l)M(k)\frac{y_j(k+l)}{y_i(k+l)}}{p_{ii}(k+l) + \sum_{j \in \mathcal{N}_i^-(k+l)} p_{ij}(k+l)\frac{y_j(k+l)}{y_i(k+l)}} = M(k).$$

Therefore, $\frac{x_i(k+l+1)}{y_i(k+l+1)} \leq M(k)$ for all $i \in \mathcal{V}$. The other inequality is similar and is left to the reader. $\square$

Next, Lemma 6 strengthens the result of Lemma 5 to a strict inequality.

Lemma 6. Consider the ratio consensus protocol comprising two updates, given by (2.58) and (2.59), where the initial time instant is k . Let Assumption 5 and Assumption 6 hold., Let $M(k)$ and $m(k)$ be as in (2.63) and (2.64). Let i be a node such that $\frac{x_i(k')}{y_i(k')} < M(k)$, and let j be a node such that $\frac{x_j(k')}{y_j(k')} > m(k)$ for some time instant $k' \geq k$. Then for all time instants $k'' \geq k'$, $\frac{x_i(k'')}{y_i(k'')} < M(k)$ and $\frac{x_j(k'')}{y_j(k'')} > m(k)$.

Proof. The proof is based on induction and follows similarly to the proof of Lemma 5

and is left to the reader. $\square$

The following definition and lemmas introduce the concept of a time path and derive an upper bound on the number of iterations required for one node to access the information of any other node in the network.

Definition 30. (*Time path*) In the case of the switching topology, a time path of length l at time t from node $i \in \mathcal{V}$ to $j \in \mathcal{V}$ is a sequence of nodes, $k_1, k_2, \dots, k_{l-1}$, such that $(k_1, i) \in \mathcal{E}(t), (k_2, k_1) \in \mathcal{E}(t+1), \dots, (j, k_{l-1}) \in \mathcal{E}(t+l-1)$. In other words, node j has access to node i 's information after l time steps through the nodes $k_1, k_2, \dots, k_{l-1}$ starting at time t .

Lemma 7. Consider a network where Assumption 5 and Assumption 6 hold. Then for any node $i \in \mathcal{V}$, the following holds:

1. $R_i(k, t-1) \subset R_i(k, t)$ for all $t = 1, 2, \dots$
2. $R_i(k, n-1) = \mathcal{V}$.

Proof. Here, $R_i(k, t+1)$ is the union of out-neighborhoods of all the elements of $R_i(k, t)$ at instant t . Clearly, $R_i(k, t-1) \subseteq R_i(k, t)$. Indeed, $l \in R_i(k, t-1)$ implies $l \in \mathcal{N}_l^+(k+t-1)$, which, in turn, implies, $l \in R_i(k, t)$.

Now, for $t = 1$:

$$R_i(k, 1) = \{m_1 : m_1 \in \mathcal{N}_i^+(k)\} = \mathcal{N}_i^+(k).$$

If $R_i(k, 1) = \mathcal{V}$, the claim is proven. Suppose $R_i(k, 1) \subset \mathcal{V}$. Note that at time instant $k+1$, the graph is strongly connected. Thus, there exists an outgoing edge between the set of nodes $R_i(k, 1)$ and $\mathcal{V} \setminus R_i(k, 1)$. That is, there exists $m_2 \in \mathcal{V} \setminus R_i(k, 1)$, such that $m_2 \in \mathcal{N}_{m_1}^+(k+1)$ for some $m_1 \in R_i(k, 1)$. Thus, $m_2 \in R_i(k, 2)$. It follows that $m_2 \in R_i(k, 2) \setminus R_i(k, 1)$, which implies that:

$$|R_i(k, 2) \setminus R_i(k, 1)| \geq 1.$$

Now, as $R_i(k, 1) \subset R_i(k, 2)$, we have:

$$|R_i(k, 2)| = |R_i(k, 1)| + |R_i(k, 2) \setminus R_i(k, 1)|$$

which implies $|R_i(k, 2)| \geq |R_i(k, 1)| + 1$. Note here that $R_i(k, 1)$ is a proper subset of $R_i(k, 2)$. Following in the same manner, we get:

$$|R_i(k, t)| \geq |R_i(k, t-1)| + 1 \geq |R_i(k, t-2)| + 2 \geq \dots \geq |R_i(k, 0)| + t,$$

and thus $|R_i(k, t)| \geq t + 1$. For $t = n - 1$, $|R_i(k, n - 1)| \geq n$; however $R_i(k, t) \subseteq \mathcal{V}$ and thus $|R_i(k, t)| \leq n$ for all t . Thus, $|R_i(k, n - 1)| = n$, implying $R_i(k, n - 1) = \mathcal{V}$. Therefore, the claim is true. $\square$

Lemma 8. *Let Assumption 5 and Assumption 6 hold. Then for any two nodes, $i, j \in \mathcal{V}$, and at any time instant k , there exists a time path of length s from i to j with $s \leq n - 1$. In other words, j has access to information of i in s number of time steps.*

Proof. Note that $R_i(k, t)$ is the set of nodes influenced by node i 's current state within the next t time steps. Now, we show the existence of a time path. We have $R_i(k, 0) \subset R_i(k, 1) \subset R_i(k, 2) \subset \dots \subset R_i(k, s) = \mathcal{V}$. Let s be the smallest time step, such that $j \in R_i(k, s)$. From Lemma 7, $s \leq n - 1$. As $j \in R_i(k, s)$, there exists $m_{s-1} \in R_i(k, s-1)$, such that $j \in N^+(m_{s-1}, k + s - 1)$. As $m_{s-1} \in R_i(k, s-1)$, it follows that there exists $m_{s-2} \in R_i(k, s-2)$, such that $m_{s-1} \in N^+(m_{s-2}, k + s - 2)$. Choosing m_i 's in this manner, we get $m_{s-2} \in R_i(k, s-2)$, $m_{s-3} \in R_i(k, s-3)$, ..., $m_1 \in R_i(k, 1)$, such that $m_1 \in N^+(i, k)$. Therefore, there exists a time path $(m_1, i), (m_2, m_1), \dots, (j, m_{s-1})$ with $s \leq n - 1$. $\square$

Using Lemma 7 and Lemma 8, we next provide a sampling of the $M(k)$ and $m(k)$, such that the resulting subsequences are strictly monotonic and converge to the average of the initial conditions of x_i variables.

Lemma 9. *Consider the ratio consensus protocol comprising updates (2.58) and (2.59). Let Assumption 5 and Assumption 6 hold. Let $M(k)$ and $m(k)$ be as in (2.63) and (2.64), such that $m(k) < M(k)$, where the initial time instant is k . Then for all $k' \geq k + n'$ and for all $i \in \mathcal{V}$:*

$$m(k) < \frac{x_i(k')}{y_i(k')} < M(k),$$

where n' is an upper bound on $n - 1$.

Proof. As $m(k) < M(k)$, it follows that there exists a node $i \in \mathcal{V}$, such that $\frac{x_i(k)}{y_i(k)} < M(k)$. Let $j \in \mathcal{V}$ be an arbitrary node, then from Lemma 8, there exists a time path

of length l from node i to node j at instant k , where $l \leq n - 1 \leq n'$. Let this path be denoted as $(m_1, i), (m_2, m_1), \dots, (j, m_{l-1})$. Then:

$$\begin{aligned}
\frac{x_{m_1}(k+1)}{y_{m_1}(k+1)} &= \frac{p_{m_1 i}(k)x_i(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)x_u(k)}{p_{m_1 i}(k)y_i(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)y_u(k)} \\
&= \frac{p_{m_1 i}(k)\frac{x_i(k)}{y_i(k)} + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)\frac{x_u(k)}{y_u(k)}}{p_{m_1 i}(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)\frac{y_u(k)}{y_i(k)}} \\
&< \frac{p_{m_1 i}(k)M(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)\frac{x_u(k)}{y_i(k)}}{p_{m_1 i}(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)\frac{y_u(k)}{y_i(k)}}.
\end{aligned}$$

By definition of $M(k)$, $\frac{x_u(k)}{y_u(k)} \leq M(k)$ for all $u \in \mathcal{V}$; thus, $x_u(k) \leq y_u(k)M(k)$. It follows that:

$$\frac{x_{m_1}(k+1)}{y_{m_1}(k+1)} < \frac{p_{m_1 i}(k)M(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)M(k)\frac{y_u(k)}{y_i(k)}}{p_{m_1 i}(k) + \sum_{u \in \mathcal{N}_{m_1}^-(k)} p_{m_1 u}(k)\frac{y_u(k)}{y_i(k)}} = M(k).$$

Thus, $\frac{x_{m_1}(k+1)}{y_{m_1}(k+1)} < M(k)$. Therefore, from Lemma 6, it follows that for all $k' \geq k+1$, $\frac{x_{m_1}(k')}{y_{m_1}(k')} < M(k)$. Similarly, if $k' \geq k+2$, then $\frac{x_{m_2}(k')}{y_{m_2}(k')} < M(k)$ and that for all $k' \geq k+l$, $\frac{x_j(k')}{y_j(k')} < M(k)$. Note that because $n' \geq n-1 \geq l$, it follows that $k+n' \geq k+l$. The condition $k' \geq k+n'$ is independent of the index j and where the node j was chosen arbitrarily. Thus, it can be concluded that $\frac{x_i(k')}{y_i(k')} < M(k)$ for all $k' > k+n'$ and for all $i \in \mathcal{V}$. This completes the proof for $M(k)$. The other inequality for $m(k)$ can be proven similarly and is left to the reader. $\square$

Remark 8. From Lemma 9, after a finite number of iterations given by n' , all ratios of the nodal states under (2.58) and (2.59) become strictly less than the maximum value of the ratio in the network in the past and strictly greater than the minimum value of the ratio in the network in the past.

The following theorem shows that after a finite time, the maximum value of the ratio

in the network decreases, and the minimum value of the ratio in the network increases.

Theorem 7. *Consider the ratio consensus protocol comprising two updates given by (2.58) and (2.59). Let Assumption 5 and Assumption 6 hold, and let the initial ratio vector be:*

$$r(un') := \left[\frac{x_1(un')}{y_1(un')} \quad \frac{x_2(un')}{y_2(un')} \cdots \frac{x_N(un')}{y_N(un')} \right]$$

such that $\min r(un') < \max r(un')$, where $u = 0, 1, 2, \dots$, and n' is an upper bound on $n - 1$. Then, $M((u + 1)n') < M(un')$ and $m((u + 1)n') > m(un')$.

Proof. Let $k = un'$, then using Lemma 9, it follows that for $k' \geq (u + 1)n'$, $\frac{x_i(k')}{y_i(k')} < M(un')$ for all $i \in \mathcal{V}$. Hence, $M((u + 1)n') := \max_{i \in \mathcal{V}} \frac{x_i((u+1)n')}{y_i((u+1)n')} < M(un')$. This completes the proof of the inequality involving $M(un')$. The other inequality involving $m(un')$ can be proved similarly and is left to the reader. $\square$

Theorem 8. *Consider the ratio consensus protocol comprising two updates given by (2.58) and (2.59). Let Assumption 5 and Assumption 6 hold. Then:*

$$\lim_{u \rightarrow \infty} M(un') = \lim_{u \rightarrow \infty} m(un') = \frac{1}{n} \sum_{j=1}^n x_j(0),$$

where n' is an upper bound on $n - 1$.

Proof. This result follows directly from Theorem 6 and Theorem 7 and is left to the reader. $\square$

Corollary 2. *Consider the ratio consensus protocol comprising two updates given by (2.58) and (2.59). Let Assumption 5 and Assumption 8 hold, and let n' be an upper bound on $n - 1$. Then:*

$$\lim_{u \rightarrow \infty} M(un') - m(un') = 0.$$

We next present the maximum-minimum consensus protocol in the case of dynamic topology and use the preceding theorems to design a finite-time termination criterion.

Maximum-Minimum Consensus Protocol:

The maximum and minimum consensus protocols, denoted by MXP and MNP protocols respectively, compute the maximum and minimum of the given initial node conditions:

$$\begin{aligned} z(0) &= [z_1(0) \ z_2(0) \ \dots \ z_n(0)]^\top, \\ w(0) &= [w_1(0) \ w_2(0) \ \dots \ w_n(0)]^\top \end{aligned}$$

in a distributed manner, respectively. The MXP and MNP protocols take $z(0)$ and $w(0)$ as inputs and generate a sequence of node values based on the following update rules for node i respectively:

$$z_i(k+1) = \max_{j \in \mathcal{N}_i^-(k)} z_j(k), \quad (2.65)$$

$$w_i(k+1) = \min_{j \in \mathcal{N}_i^-(k)} w_j(k). \quad (2.66)$$

Proposition 1. *The MXP protocol given by (2.65) converges to $\max_{j \in \mathcal{V}} z_j(0)$ in finite time $k \leq n'$ for any $n' \geq n - 1$.*

Proof. Let m be a node with state value at $z_m(0) = \max_{j \in \mathcal{V}} z_j(0)$. Then, at $k = 1$, all nodes connected to m at instant $k = 0$ will have the maximum value $z_m(0)$. Then, from the definition of $R_m(0, k)$ in Lemma 8, if $m_1 \in R_m(0, 1)$ then $z_{m_1}(1) = z_m(0)$. Similarly, at $k = 2$, all the nodes connected to the elements of the set $R_m(0, 1)$ at instant $k = 1$ will have the maximum value, that is, if $m_2 \in R_m(0, 2)$, then $z_{m_2}(2) = z_m(0)$, and so on. At $k = n - 1$, if $m_{n-1} \in R_m(0, n - 1)$, then $z_{m_{n-1}}(n - 1) = z_m(0)$. As $R_m(0, n - 1) = \mathcal{V}$, we have for all $j \in \mathcal{V}$, $z_j(n - 1) = z_m(0)$. Thus, at instant $k = n - 1$, all nodes will have the maximum value. As $n' \geq n - 1$, $z_j(k)$ converges to $\max_{j \in \mathcal{V}} z_j(0)$ in finite time $k \leq n'$. $\square$

Similarly, we have the following result for the MNP protocol,

Proposition 2. *The MNP protocol given by (2.66) converges to $\min_{j \in \mathcal{V}} w_j(0)$ in finite time $k \leq n'$ for any $n' \geq n - 1$.*

Distributed Finite-Time Termination Algorithm for Ratio Consensus:

Here, we propose an algorithm using the MXP-MNP protocol, which allows each node to simultaneously detect the convergence of the ratio consensus within a prespecified

threshold ρ . In the proposed algorithm, the initial conditions for the MXP-MNP protocol are set as the ratio of the initial values held by the nodes.

Definition 31. (*Epoch*) Let n' be an upper bound on $n - 1$, then the u^{th} epoch is defined as the state update iteration at the instant un' for any positive integer u .

The MXP-MNP protocol is re-initialized at every u^{th} epoch, that is, $k = un'$, where $u = 1, 2, \dots$, with $z(un') = \frac{x(un')}{y(un')}$ and $w(un') = \frac{x(un')}{y(un')}$, respectively. We define $\bar{\alpha}_i(un') := \max z(un') = M(un')$, $\underline{\alpha}_i(un') = \min w(un') = m(un')$ and $\beta_i(un') := \bar{\alpha}_i - \underline{\alpha}_i = M(un') - m(un')$. Each node compares β_i with ρ at every epoch, and if $\beta_i < \rho$, then it terminates the consensus protocol. Details of this scheme are given in Algorithm 1.

Algorithm 1: Finite-time termination of ratio consensus for switching topology (at each node $i \in \mathcal{V}$)

```

Input:
    |  $x_j(0), y_j(0) = 1, j \in \mathcal{N}_i^-(0), \rho$  ;                               // Initial condition
Initialize:
    |  $k := 0; z_i := x_i(0)/y_i(0); w_i := x_i(0)/y_i(0); u := 1;$ 
Repeat:
    | /* ratio consensus updates of node  $i$  given by (2.58) and (2.59)
    |   */
    |  $x_i(k+1) := \sum_{j \in \mathcal{N}_i^-(k)} p_{ij}(k)x_j(k); y_i(k+1) := \sum_{j \in \mathcal{N}_i^-(k)} p_{ij}(k)y_j(k);$ 
    | /* global max-min updates of node  $i$  given by (2.65) and (2.66),
    |   respectively */
    |  $z_i := \max_{j \in \mathcal{N}_i^-(k)} z_j; w_i := \min_{j \in \mathcal{N}_i^-(k)} w_j;$ 
    | if  $k = un'$  then
    |   |  $\bar{\alpha}_i := z_i; \underline{\alpha}_i := w_i; \beta_i := \bar{\alpha}_i - \underline{\alpha}_i;$ 
    |   | if  $\beta_i < \rho$  then
    |     | break ;                               // stop  $x_i, y_i, z_i$  and  $w_i$  updates
    |   | else
    |     | set  $z_i = x_i(un')/y_i(un'); w_i = x_i(un')/y_i(un');$ 
    |     |  $u = u + 1;$ 
    |   end
    |  $k = k + 1;$ 

```

Theorem 9. Algorithm 1 simultaneously converges in finite time at each node.

Proof. From Corollary 3, it follows that $M(un') - m(un') \rightarrow 0$ as $u \rightarrow \infty$. Thus, for any given $\rho > 0$, there exists an integer $t(\rho)$, such that for all $u \geq t(\rho)$, $|M(un') - m(un')| < \rho$

for all nodes in the network. As each node has access to $M(un')$ and $m(un')$, each node simultaneously detects convergence at the same iteration. $\square$

Remark 9. *Note that using the maximum-minimum protocol, the global maximum and minimum values at any instant k are available to each node at instant $k + n'$. Further, the only global parameter needed for Algorithm 1 is knowledge of the number of nodes in the network. Note, however, that each node does not need to know the actual number of nodes but only some upper bound on the same. In most applications, an upper bound on the number of nodes is readily available.*

This result is a nontrivial extension of [113], [114] because these works considered only a static network, where the finite-time consensus is based on knowledge of the upper bound on the graph diameter. For the specific example with a six-node network considered earlier, it was observed that it takes five iterations for the range to contain all the nodes. In other words, it requires at least 5 ($= n - 1$) iterations for any node to receive information from node 1, which follows directly from Lemma 8. Thus, choosing n' as $(n - 1)$ will lead to strict monotonicity of the global maximum/minimum, as described in Theorem 7. Moreover, each node implementing Algorithm 1 will have a finite-time distributed criterion to detect global convergence of the consensus updates (2.59) and (2.58).

Remark 10. *Here, we summarize the dependence of the proposed finite-time termination algorithm on various design parameters. The tolerance ρ has an inverse relationship with the time it takes for the algorithm to converge; the lower the tolerance, the longer it takes for the algorithm to converge. Weights p_{ij} need to be column-stochastic for the “ratio consensus” algorithm to converge. Further, weights p_{ij} also govern the rate of convergence. The termination protocol is agnostic to the weights and will detect convergence independent of the weights. All y_i s need to be initialized to 1, otherwise the “ratio consensus” algorithm will not converge to the average of initial values.*

Experimental Results:

Consider a network of 1000 agents with switching topology represented by a digraph chosen at random from a set of 50 strongly connected digraphs for every time instant. Three instances of these digraphs are shown in Figure 2.10. Here, each agent running the consensus Algorithm 1 is implemented using a NodeJS *socket.io* server. This is the first time a salable real-world experimental test bed is implemented for consensus algorithms

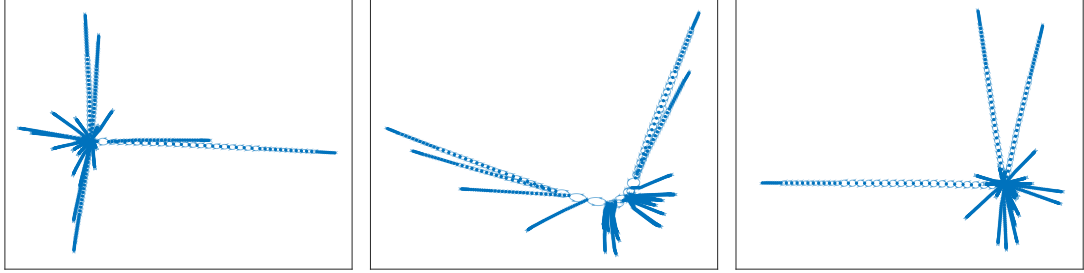


Figure 2.10: Topology shown as a bidirectional graph at time instants $k = 109$, $k = 239$, and $k = 275$ respectively.

in a time-varying network. Here, at each time instant, the network is based on a random topology (chosen from 50 strongly connected digraphs), with the weights using the equal neighbor weights rule (2.61) and (2.62). This guarantees that Assumption 6 is always valid. This provides an experimental environment where the information exchange between the agents happens through a communication channel (with arbitrary delays), thus proving the applicability of our protocol in a real-world setting, unlike the simulation-based validation present in most existing literature.

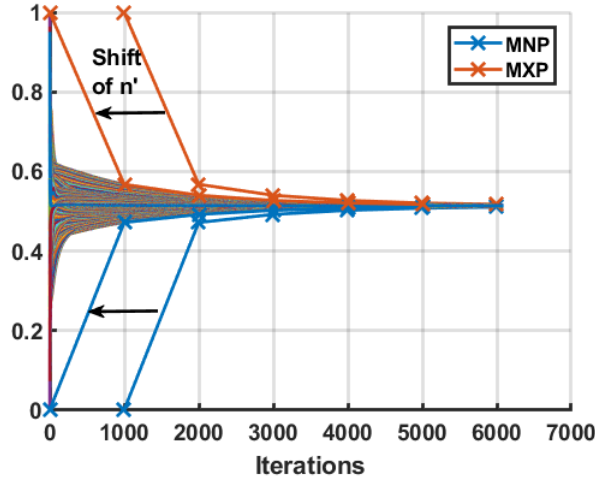


Figure 2.11: Finite-time termination of ratio consensus on a 1000-node dynamic network in 6000 iterations with $\rho = 0.01$. Each curve (without markers) in the plot is the time evolution of the ratio state of 1000 nodes. MXP/MNP (when using n' as sampling) are monotonic and correctly converge. The MXP/MNP curves are shifted left to coincide with global maximum and minimum, as mentioned in Remark 9.

The initial conditions of the numerator states are chosen at random from $(0, 1)$. With $\rho = 0.01$ and $n' = 1000$, Algorithm 1 results in the distributed finite-time termination

of computations performed by the agents in 6000 iterations, as shown in Figure 2.11. Observe that the ratios of the nodal states are close to the average of the numerator's initial conditions (consensus value). The consensus value for this instance is 0.514. This experiment was repeated multiple times by randomly choosing digraphs for each iteration as well as the initial values of the agents. The proposed MXP-MNP protocol was able to distributively terminate the algorithm each time. As noted in Remark 9, the global maximum and minimum state values at k are available at $n' + k$. This is shown in Figure 2.11, where shifting the MXP and MNP plots left by n' time instants, we get the MXP' and MNP' plots, which coincide with the global maximum and minimum.

Remark 11. *This protocol uses the knowledge of only the upper bound on the number of nodes (n') and checks for convergence every n' number of iterations. As a result, after the ρ -convergence, the MXP-MNP protocol might take additional $2n'$ iterations for convergence in the worst-case scenario. Further, when computational and storage capacity is not a constraint for an agent, it can run the n' MXP-MNP protocol, shifted in time, with the first one starting at $k = 1$, the second one starting at $k = 2$, and so on. In this case, detection will only be delayed by n' if the agents monitor all the MXP-MNP protocols. Note that without additional information on the network topology, which imposes sharing private information (for example, degree of neighbors), detection and stopping upon convergence cannot be achieved in fewer iterations. For example, a chain network takes $(n - 1)$ iterations for information to travel from one end to another, so even if an individual node has achieved convergence, it can take up to $(n - 1)$ iterations for all the other nodes in the network to know about it. Thus, when the agents do not have information on the underlying communication topology, the global stopping criterion will always be delayed by at least n' .*

We next present the specific six-node example discussed earlier, where it was shown analytically that some nodes do not have access to the global maximum and minimum within $\mathcal{D}_{max}$ iterations. We consider a 6-node network, where the topology is given by the sequence $\{\mathcal{G}_2, \mathcal{G}_2, \mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_2, \mathcal{G}_1, \dots\}$, with $\mathcal{G}_1$ and $\mathcal{G}_2$, as defined in Figure 2.9. The initial state value of the nodes is chosen as $x(0) = [10, 2, 2, 2, 2, 2]$, with an average of 3.33. As mentioned earlier, each node is implemented using a NodeJS *socket.io* server. We use the sampling length of both $\mathcal{D}_{max} = 3$ and $n' = 5$ to reset z_i and w_i in Algorithm 1, and we plot the MXP/MNP of node 5 in Figure 2.12 for both cases. The MXP/MNP plots are shifted by the sampling length to coincide with the initialized values of z_i and w_i . The results clearly show that the MXP, when using

$\mathcal{D}_{max}$ ($\text{MXP}_{\mathcal{D}_{max}}$), the plot for node 5 is non-monotonic. Moreover, $\text{MXP}_{\mathcal{D}_{max}}$ is unable to capture the global maximum value of the network (because some nodes have state values greater than $\text{MXP}_{\mathcal{D}_{max}}$ between iterations 1 and 10). Further, the algorithm will stop incorrectly at the third iteration for node 5 because $\text{MXP}_{\mathcal{D}_{max}}$ and $\text{MNP}_{\mathcal{D}_{max}}$ have the same value. On the other hand, MXP/MNP (using n') is monotonic and converges when the states have converged.

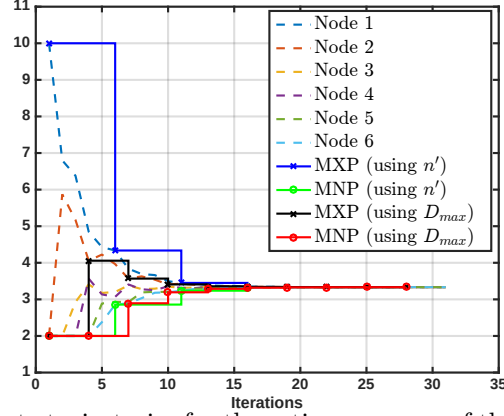


Figure 2.12: Node ratio state trajectories for the ratio consensus of the 6-node network shown in Figure 2.9. The MXP/MNP for sampling length of both $\mathcal{D}_{max} = 3$ and $n' = 5$ are plotted. MXP when $\mathcal{D}_{max}$ is used as sampling is non-monotonic. MXP/MNP (when using n' as sampling) are monotonic and correctly converge. The MXP/MNP curves are shifted left to coincide with the global maximum and minimum.

We also present the evolution of the numerator ($x_i(k)$) and denominator states ($y_i(k)$) in Figure 2.13. As mentioned in Remark 6, the numerator and denominator are bounded above and below.

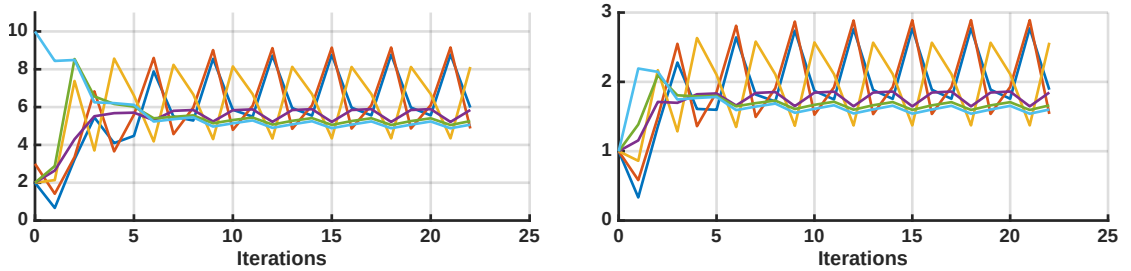


Figure 2.13: Node numerator and denominator state trajectories for the ratio consensus of the 6-node network shown in Figure 2.9. Both states for each node are bounded above and below.

2.5.3 Distributed Finite-time Approximate Consensus in Higher Dimensions

Vector Consensus Framework:

Here, we extend a key result from [42], [63] where a ratio of two states was maintained to reach an average consensus. We consider the network topology to be represented by a digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ containing n nodes and satisfies the following assumptions:

Assumption 7. *The digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ representing the agent interconnections is strongly connected.*

Assumption 8. *Let $\mathbf{P} = [P_{ij}]$ be a primitive column stochastic matrix with digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $p_{ij} > 0$ if and only if $(i, j) \in \mathcal{E}$*

Each node $i \in \mathcal{V}$ maintains three state estimates at time k , denoted by $x_i(k) \in \mathbb{R}^d$, $y_i(k) \in \mathbb{R}$ and $r_i(k) \in \mathbb{R}^d$. Here $d (\geq 1)$ is the dimension of each node's state. Node i updates its numerator and denominator states at the $(k+1)^{th}$ discrete iteration according to the following update law:

$$x_i(k+1) = p_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i^-} p_{ij}x_j(k), \quad (2.67)$$

$$y_i(k+1) = p_{ii}y_i(k) + \sum_{j \in \mathcal{N}_i^-} p_{ij}y_j(k), \quad (2.68)$$

where, $\mathcal{N}_i^-$ is the set of in-neighbors of node i . The initial conditions for the numerator vector state and denominator state for any node $i \in \mathcal{V}$ are:

$$x_i(0) = [x_{i1}(0) \ x_{i2}(0) \ \dots \ x_{id}(0)]^\top, \ y_i(0) = 1, \text{ for all } i \in \mathcal{V}. \quad (2.69)$$

Node i further updates its ratio state as:

$$r_i(k+1) = \frac{1}{y_i(k+1)}x_i(k+1). \quad (2.70)$$

Under Assumptions 7, 8 and the initialization in (2.69), ratio state in (2.70) is well defined. The next theorem establishes the convergence of the ratio state.

Theorem 10. *Let $\{x_i(k)\}$, $\{y_i(k)\}$ and $\{r_i(k)\}$ be the sequences generated by (2.67), (2.68) and (2.70) respectively. Let the initial conditions for the network states be as*

defined in (2.69). Then, under Assumptions 7 and 8 the ratio state $r_i(k)$ asymptotically converges to $\bar{r} := \lim_{k \rightarrow \infty} \frac{1}{y_i(k)} x_i(k) = \frac{1}{n} \sum_{j=1}^n x_j(0)$ for all $i \in \{1, \dots, n\}$.

Proof. Let $r_{is}(k)$ and $x_{is}(k)$ be the s -th elements of $r_i(k)$ and $x_i(k)$ respectively. Then the update equations (2.67) and (2.70) can be written as

$$\begin{aligned} x_{is}(k+1) &= p_{ii}x_{is}(k) + \sum_{j \in \mathcal{N}_i^-} p_{ij}x_{js}(k) \\ r_{is}(k+1) &= \frac{1}{y_i(k+1)} x_{is}(k+1) \end{aligned}$$

Then from [42], [63], for all $s \in \{1, 2, \dots, d\}$ and $i \in \mathcal{V}$ we have $\lim_{k \rightarrow \infty} r_{is}(k) = \frac{1}{n} \sum_{j=1}^n x_{js}(0)$. Therefore,

$$\begin{aligned} \lim_{k \rightarrow \infty} r_i(k) &= \left[\lim_{k \rightarrow \infty} r_{i1}(k) \ \dots \ \lim_{k \rightarrow \infty} r_{id}(k) \right]^\top = \left[\frac{1}{n} \sum_{j=1}^n x_{j1}(0) \ \dots \ \frac{1}{n} \sum_{j=1}^n x_{jd}(0) \right]^\top \\ &= \frac{1}{n} \sum_{j=1}^n [x_{j1}(0) \ \dots \ x_{jd}(0)]^\top = \frac{1}{n} \sum_{j=1}^n x_j(0). \end{aligned}$$

Thus, we have convergence in the case when node states are vectors. $\square$

A Monotonicity Result for Certain Consensus Protocols:

In this section, we demonstrate a monotonicity property of the ratio consensus problem. We will show that under the update framework given by (2.70), the network states $\{r_i(k)\}_{i=1}^n$ at time k can be used to define a sequence of polytopes $\{r_k\}_{k=0}^\infty$ as

$$r_k := \text{co}(\{r_i(k)\}_{i=1}^n). \quad (2.71)$$

We then prove that these polytopes are descending in the sense that $r_k \supseteq r_{k+1}$. In the process we recover monotonicity results of the min-max protocols from the one-dimensional case, see [115], as well as their generalization to the vector case, see [124]. This insight is used to develop distributed stopping criteria, guaranteeing convergence of all nodes' vector ratio states within an ε -ball of the consensus for a general norm.

Lemma 10. *Let $p_{ij} \geq 0$, $v \in \mathbb{R}^d$, $x_1, \dots, x_n \in \mathbb{R}^d$, and $y_1, \dots, y_n \in \mathbb{R}$ with $M_v \in \mathbb{R}$*

such that $x_i \cdot v \leq M_v y_i$ for all i then

$$\sum_j p_{ij} x_j \cdot v \leq M_v \sum_j p_{ij} y_j.$$

holds for all i as well.

Proof. Computing directly,

$$\sum_j p_{ij} x_j \cdot v \leq \sum_j p_{ij} M_v y_j = M_v \sum_j p_{ij} y_j.$$

□

Theorem 11. *The vector ratio consensus algorithm is monotonically convex in the sense for $r_i(k)$ defined in (2.70) $r_k \subseteq r_{k+1}$, for r_k polytopes defined in (2.71).*

Proof. If $r_k = \text{co}(\{r_i(k)\}_{i=1}^n)$ is contained in a half space $\{w : w \cdot v \leq M_v\}$ then in particular $(x_i(k)/y_i(k)) \cdot v \leq M_v$. Applying Lemma 10, $\sum_j p_{ij} x_j(k) \cdot v \leq M_v \sum_j p_{ij} y_j(k)$, so that $r_i(k+1) = \frac{x_i(k+1)}{y_i(k+1)}$ is an element of $\{x : x \cdot v \leq M_v\}$ as well. Since the half-space is arbitrary, and r_{k+1} is the intersection of all closed half-spaces containing it, $r_i(k+1) \in r_k$, and hence $r_{k+1} \subseteq r_k$. □

When $d = 1$, the convex hull is simply described, $\text{co}(r(k)) = [\min_j \{r_j(k)\}_{j=1}^n, \max_j \{r_j(k)\}_{j=1}^n]$, so that $\text{co}(r(k)) \supseteq \text{co}(r(k+1))$ gives the monotonicity results from [115], $\min_j \{r_j(k)\}_{j=1}^n \leq \min_j \{r_j(k+1)\}_{j=1}^n$ and $\max_j \{r_j(k)\}_{j=1}^n \geq \max_j \{r_j(k+1)\}_{j=1}^n$.

Norm Based Finite-time Termination:

Similar to the convex hull comprising all points (corresponding to each agent), the radius of a minimal norm ball in d dimension enclosing all the points can also be used as a termination criterion. Once the radius is within some bound ε , it can be easily shown that every agent's state is within 2ε of the consensus value. However, even in the p -norm case calculation of the minimum norm ball in a distributed manner is a difficult problem [125]. We provide an algorithm that distributedly finds an approximation of the minimal ball at each agent. We show that the minimal ball is enclosed in this approximation, thus if the approximate ball's radius is within ε then the minimal ball's radius is within ε as well. This is established in the next Lemma.

Lemma 11. *Let $\{r_i(k)\}$ be the sequence generated by the consensus protocol of (2.70). For all $i \in \mathcal{V}$, let*

$$R_i(k+1, k') := \max_{j \in \mathcal{N}_i^-} \{\|r_i(k'+k+1) - r_j(k'+k)\| + R_j(k, k')\} \quad (2.72)$$

with $R_i(0, k') := 0$ and $k' \geq 0$. Then

$$r_j(k') \in B\{R_i(\mathcal{D}, k'), r_i(k' + \mathcal{D})\}, \quad (2.73)$$

where, $B\{R_i(\mathcal{D}, k'), r_i(k' + \mathcal{D})\}$ is a ball of radius $R_i(\mathcal{D}, k')$ centered at $r_i(k' + \mathcal{D})$ and $\mathcal{D}$ is the diameter of the underlying graph topology.

Proof. We first prove the following claim.

$$r_j(k') \in B\{R_i(k, k'), r_i(k' + k)\} \quad (2.74)$$

for all $j \in \mathcal{V}$ such that the length of the shortest path from i to j is less than equal to k . Clearly above claim is sufficient to prove (2.73) as when $k = \mathcal{D}$, (2.74) is valid for all $j \in \mathcal{V}$. Let the length of the shortest path from i to j be denoted as $|path(i, j)|$. We prove this claim using induction. For $k = 1$,

$$R_i(1, k') = \max_{j \in \mathcal{N}_i^-} \{\|r_i(k'+1) - r_j(k')\|\}.$$

Then for all $j \in \mathcal{N}_i^-$, that is for all j such that $|path(i, j)| \leq 1$, we get

$$\|r_i(k'+1) - r_j(k')\| \leq R_i(1, k') \implies r_j(k') \in B\{R_i(1, k'), r_i(k'+1)\}.$$

Hence, true for $k = 1$.

Let us assume (2.74) is true for k . Let j be a node such that $|path(i, j)| \leq k + 1$. Let q be a neighbor of i on the shortest path from i to j , then $|path(q, j)| \leq k$. Then from the induction assumption,

$$r_j(k') \in B\{R_q(k, k'), r_q(k + k')\},$$

that is,

$$\|r_q(k' + k) - r_j(k')\| \leq R_q(k, k'). \quad (2.75)$$

From definition of $R_i(k+1, k')$,

$$\|r_i(k' + k + 1) - r_q(k' + k)\| + R_q(k, k') \leq R_i(k + 1, k').$$

From triangle inequality,

$$\|r_i(k' + k + 1) - r_j(k')\| \leq \|r_i(k' + k + 1) - r_q(k' + k)\| + \|r_q(k' + k) - r_j(k')\|.$$

Using (2.75),

$$\begin{aligned} \|r_i(k' + k + 1) - r_j(k')\| &\leq \|r_i(k' + k + 1) - r_q(k' + k)\| + R_q(k, k') \\ \implies \|r_i(k' + k + 1) - r_j(k')\| &\leq R_i(k + 1, k') \\ \implies r_j(k') &\in B\{R_i(k + 1, k'), r_i(k' + k + 1)\}. \end{aligned}$$

Hence, true for $k + 1$. and induction holds. $\square$

Lemma 11 provides an easy and distributed way to find a ball that encloses all the nodes. The only information needed by a node is the current radius of its neighbors (along with the states out ratio consensus protocol) and it can calculate the final radius within $\mathcal{D}$ iteration. Further, since the ball $B\{R_i(\mathcal{D}, k'), r_i(k' + \mathcal{D})\}$ encloses all the nodes, it also encloses the minimum ball, as mentioned earlier. We next present a framework that we use to prove that this radius converges to 0 and can be used as a distributed stopping criterion.

Let the element-wise maximum and minimum of the ratio state over all the agents at any time instant k be given by, $M(k) = [M_1(k) \ M_2(k) \ \dots \ M_d(k)]^\top$ and $m(k) = [m_1(k) \ m_2(k) \ \dots \ m_d(k)]^\top$ respectively. That is,

$$M_s(k) := \max_{i \in \mathcal{V}} r_{is}(k) \tag{2.76}$$

$$m_s(k) := \min_{i \in \mathcal{V}} r_{is}(k), \tag{2.77}$$

where, $M_s(k) \in \mathbb{R}$, $m_s(k) \in \mathbb{R}$ for all $s \in \{1, 2, \dots, d\}$ and $r_{is}(k)$ is the s -th element of $r_i(k)$. Then, for all time instants $k' \geq k$ and for all $i \in \mathcal{V}$ and $s \in \{1, 2, \dots, d\}$ [113],

$$m_s(k) \leq r_{is}(k') \leq M_s(k). \tag{2.78}$$

Further, for all $i \in \mathcal{V}$, $l \geq 0$ and $s \in \{1, 2, \dots, d\}$ [113],

$$M_s((l+1)\mathcal{D}) < M_s(l\mathcal{D}) \quad (2.79)$$

$$m_s((l+1)\mathcal{D}) > m_s(l\mathcal{D}). \quad (2.80)$$

By using (2.78), (2.79), and (2.80), we can prove the following theorem.

Theorem 12. *Consider the consensus protocol of (2.67), (2.68), and (2.70). Let Assumptions 7 and 8 hold. Then,*

$$\lim_{l \rightarrow \infty} M(l\mathcal{D}) = \lim_{l \rightarrow \infty} m(l\mathcal{D}) = \frac{1}{n} \sum_{i=1}^n x_i(0)$$

where, $M(k)$ and $m(k)$ are as defined earlier.

Proof. It follows from Theorem 10 that $\lim_{k \rightarrow \infty} r_i(k) = \frac{1}{n} \sum_{i=1}^n x_i(0)$ for all $i \in \mathcal{V}$. This implies that for all $i \in \mathcal{V}$, $s \in \{1, 2, \dots, d\}$ and given $\varepsilon > 0$, there exists a $L > 0$ such that for all $k \geq L$, $\left\| r_{is}(k) - \left[\frac{1}{n} \sum_{i=1}^n x_i(0) \right]_s \right\| < \varepsilon$. This implies that, there exists $N > 0$ such that for all $k \geq N$, $\left\| \max_i r_{is}(k) - \left[\frac{1}{n} \sum_{i=1}^n x_i(0) \right]_s \right\| < \varepsilon$. Similarly, $\left\| \min_i r_{is}(k) - \left[\frac{1}{n} \sum_{i=1}^n x_i(0) \right]_s \right\| < \varepsilon$. Thus, it follows that, $\lim_{k \rightarrow \infty} M_s(k) = \left[\frac{1}{n} \sum_{i=1}^n x_i(0) \right]_s$ and $\lim_{k \rightarrow \infty} m_s(k) = \left[\frac{1}{n} \sum_{i=1}^n x_i(0) \right]_s$.

As $\{M_s(l\mathcal{D})\}_{l=0,1,2,\dots}$ and $\{m_s(l\mathcal{D})\}_{l=0,1,2,\dots}$ are sub-sequences of convergent sequences $\{M_s(l)\}_{l=0,1,2,\dots}$ and $\{m_s(l)\}_{l=0,1,2,\dots}$ respectively, they converge to the same limit. Therefore,

$$\lim_{l \rightarrow \infty} M_s(l\mathcal{D}) = \left[\frac{1}{n} \sum_{i=1}^n x_i(0) \right]_s = \lim_{l \rightarrow \infty} m_s(l\mathcal{D}).$$

As the above result is true for all $s \in \{1, 2, \dots, d\}$ we have,

$$\lim_{l \rightarrow \infty} M(l\mathcal{D}) = \frac{1}{n} \sum_{i=1}^n x_i(0) = \lim_{l \rightarrow \infty} m(l\mathcal{D}).$$

□

Corollary 3. *Consider the consensus protocol of (2.67), (2.68), and (2.70). Let Assumptions 7 and 8 hold. Then,*

$$\lim_{l \rightarrow \infty} \|M(l\mathcal{D}) - m(l\mathcal{D})\| = 0.$$

Proof. The proof directly results from Theorem 12 and is left to the reader. $\square$

It is clear from Lemma 11 that at any instant k , all agents ratio states are within $2R_i(\mathcal{D}, k)$ of each other, that is,

$$\max_{i,j \in \mathcal{V}} \|r_i(k) - r_j(k)\| \leq 2R_i(\mathcal{D}, k). \quad (2.81)$$

Thus if $R_i(\mathcal{D}, k)$ is within a tolerance $\varepsilon/2$, all the agents' ratio states will be within ε of consensus. We next provide convergence result for $R_i(\mathcal{D}, k)$ as $k \rightarrow \infty$.

Theorem 13. *Consider the consensus protocol of (2.70) and update in (2.72). Let $\overline{R}_i(l) := R_i(\mathcal{D}, l\mathcal{D})$ for $l = 0, 1, 2, \dots$ and all $i \in \mathcal{V}$. Then*

$$\lim_{l \rightarrow \infty} \overline{R}_i(l) = 0 \quad (2.82)$$

for all $i \in \mathcal{V}$.

Proof. By definition,

$$R_i(1, l\mathcal{D}) = \max_{j \in \mathcal{N}_i^-} \{\|r_i(1 + l\mathcal{D}) - r_j(l\mathcal{D})\| + R_j(0, l\mathcal{D})\}$$

which implies,

$$R_i(1, l\mathcal{D}) \leq \max_{j \in \mathcal{N}_i^-} \|r_i(1 + l\mathcal{D}) - r_j(l\mathcal{D})\| + \max_{j \in \mathcal{N}_i^-} R_j(0, l\mathcal{D}) \quad (2.83)$$

Let $M(l\mathcal{D})$ and $m(l\mathcal{D})$ be as defined in Theorem 12. As $r_i(1 + l\mathcal{D}) \geq m(l\mathcal{D})$ and $r_j(l\mathcal{D}) \leq M(l\mathcal{D})$ from (2.78), we get

$$\|r_i(1 + l\mathcal{D}) - r_j(l\mathcal{D})\| \leq \|M(l\mathcal{D}) - m(l\mathcal{D})\|. \quad (2.84)$$

Then using (2.84) in (2.83) and observing $R_j(0, l\mathcal{D}) = 0$ for all $j \in \mathcal{V}$, we get

$$R_i(1, l\mathcal{D}) \leq \|M(l\mathcal{D}) - m(l\mathcal{D})\|. \quad (2.85)$$

Similarly,

$$R_i(2, l\mathcal{D}) \leq \max_{j \in \mathcal{N}_i^-} \|r_i(2 + l\mathcal{D}) - r_j(1 + l\mathcal{D})\| + \max_{j \in \mathcal{N}_i^-} R_j(1, l\mathcal{D}) \quad (2.86)$$

Again as $r_i(2 + l\mathcal{D}) \geq m(l\mathcal{D})$ and $r_j(1 + l\mathcal{D}) \leq M(l\mathcal{D})$, we have

$$\|r_i(2 + l\mathcal{D}) - r_j(1 + l\mathcal{D})\| \leq \|M(l\mathcal{D}) - m(l\mathcal{D})\| \quad (2.87)$$

Then using (2.85), (2.86) and (2.87), we get

$$R_i(2, l\mathcal{D}) \leq 2\|M(l\mathcal{D}) - m(l\mathcal{D})\|.$$

Following the same process, we have

$$\overline{R}_i(l) = R_i(\mathcal{D}, l\mathcal{D}) \leq \mathcal{D}\|M(l\mathcal{D}) - m(l\mathcal{D})\|. \quad (2.88)$$

Then from Corollary 3,

$$\lim_{l \rightarrow \infty} \overline{R}_i(l) = 0.$$

□

Note that $R_i(l)$ can be different for different nodes and each node might detect ε -convergence ($R_i(l) < \varepsilon$) at different time instants. But according to Lemma 11, once $R_i(l) < \varepsilon$ for any $i \in \mathcal{V}$, $\|r_i(l\mathcal{D}) - r_j(l\mathcal{D})\| < 2\varepsilon$, that is the ratio state is within 2ε of consensus value, and the consensus is achieved. Further, any node i that detects convergence can propagate a “converged flag” in the network. To take that into account, we run a separate 1-bit consensus algorithm (denoted as convergence consensus) for each node where each node maintains a convergence state $b_i(k)$ and shares it with neighbors. Each node initializes $b_i(k)$ at every $l\mathcal{D}$ iteration for $l \in \{0, 1, 2, \dots\}$ with 1 or 0 depending on whether the node has detected convergence or not, and updates its value on every iteration using,

$$b_i(k+1) = \bigcup_{j \in \mathcal{N}_i^-} b_j(k), \quad (2.89)$$

where $\bigcup$ denotes “OR” operation, $k \geq 0$ and $b_j(0) = 1$ if node j has detected convergence at initialization instant 0 and $b_j(0) = 0$ otherwise. Clearly, if $b_j(0) = 1$ for any $j \in \mathcal{V}$,

then $b_i(\mathcal{D}) = 1$ for all $i \in \mathcal{V}$ where $\mathcal{D}$ is the diameter. Thus each node can use $b_i(\mathcal{D})$ as a stopping criterion.

Using the above discussion and Theorem 13, we present an algorithm (see Algorithm 2) which calculates the radius $\overline{R}_i(l)$ for $l = 0, 1, 2, \dots$ and all $i \in \mathcal{V}$ and provides a finite-time stopping criterion for vector consensus.

Algorithm 2: Finite-time terminated higher-dimension d ratio consensus (at each node $i \in \mathcal{V}$)

```

Input:
|  $\varepsilon, x_i(0)$  ; // Initial condition
Initialize:
|  $k := 0; R_i(0) := 0; y_i(0) = 1; b_i(0) = 0; l := 1;$ 
Repeat:
| Input:
| |  $x_j(k), y_j(k), R_j(k), j \in \mathcal{N}_i^-$ 
| /* ratio consensus updates of node  $i$  given by (2.67), (2.68)
| | and (2.70) */
|  $x_i(k+1) := \sum_{j \in \mathcal{N}_i^- \cup \{i\}} p_{ji}(k)x_j(k); y_i(k+1) := \sum_{j \in \mathcal{N}_i^- \cup \{i\}} p_{ji}(k)y_j(k);$ 
|  $r_i(k+1) := \frac{1}{y_i(k+1)}x_i(k+1);$ 
| /* radius updates of node  $i$  given by (2.72) */
|  $R_i(k+1) := \max_{j \in \mathcal{N}_i^-} \{\|r_i(k+1) - r_j(k)\| + R_j(k)\}; b_i(k+1) = \bigcup_{j \in \mathcal{N}_i^-} b_j(k)$ 
| if  $k = lD$  then
| | if  $b_i(k+1) = 1$  then
| | | break
| | else
| | |  $\overline{R}_i(l) = R_i(k+1);$ 
| | | if  $(\overline{R}_i(l) < \varepsilon)$  then
| | | |  $b_i(k+1) = 1$ 
| | | else
| | | |  $R_i(k+1) = 0; b_i(k+1) = 0; l = l + 1;$ 
| | end
|  $k = k + 1;$ 

```

Theorem 14. *Algorithm 2 converges in a finite time simultaneously at each node.*

Proof. From Corollary 3, it follows that $\overline{R}_i(l) \rightarrow 0$ as $l \rightarrow \infty$. Thus, for any given $\varepsilon > 0$ and node $i \in \mathcal{V}$ there exists an integer $t(\varepsilon, i)$ such that for $l = t(\varepsilon, i)$, $\overline{R}_i(l) < \varepsilon$. As each

node has access to $\overline{R}_i(l)$, convergence can be detected by each node and the convergence bit $b_i(l\mathcal{D} + 1)$ will be set to 1. Thus $b_i(l\mathcal{D} + di + 1) = 1$ for all $i \in \mathcal{V}$ and the algorithm will stop simultaneously at each node. $\square$

Remark 12. Notice that using the above protocol, each node detects convergence simultaneously. Further, the only global parameter needed for Algorithm 2 is the knowledge of diameter $\mathcal{D}$. However, it should be noted that each node does not need to know the actual diameter $\mathcal{D}$ but some upper bound. In most applications, an upper bound on the diameter $\mathcal{D}$ is readily available.

Remark 13. It is to note here that for Algorithm 2, the only extra communication required between nodes is the passing of the current radius at each node, a scalar value, along with a single bit for convergence consensus. Therefore the total bandwidth required for each neighbor-neighbor interaction is $((d+1)B+1)$ where B is the bit length (usually 32) for floating point representation. Thus, the above protocol is suitable for ad-hoc communication networks where communication cost is high and bandwidth is limited.

A finite-time termination criterion for vector consensus was previously provided in [126]. There, each element of the ratio state required a maximum-minimum protocol (see (2.76) and (2.77)), with the stopping criterion given by,

$$\left\| \max_{s \in \{1,2,\dots,d\}} \max_{i \in \mathcal{V}} r_{is}(k) - \min_{s \in \{1,2,\dots,d\}} \min_{i \in \mathcal{V}} r_{is}(k) \right\| < \varepsilon$$

This maximum-minimum is a special case for finding a minimum convex set in the form of a hyper rectangle (box) which encompasses all the points. Here, at each iteration, two extra states are shared by each node, namely, one state for element-wise maximum and the other for element-wise minimum. Thus the total communication bandwidth required for this algorithm is $3Bd$. An example case where $d = 10, B = 32$, requires a bandwidth of 960 bits per interaction. For this example, Algorithm 2 only requires 353 bits per interaction, with a reduction of 63%. Thus for the applications with high dimensional vector consensus, the algorithm reported here provides a reliable distributed stopping criterion with significantly less communication bandwidth.

Numerical simulations:

In this section, we present simulation results to demonstrate the finite-time stopping criterion for high-dimensional ratio consensus. A network of 25 nodes is considered

which is represented by a randomly generated digraph (see Figure 2.14 with diameter $\mathcal{D} = 6$. Here the numerator state is chosen as a 10-dimensional vector and selected ran-

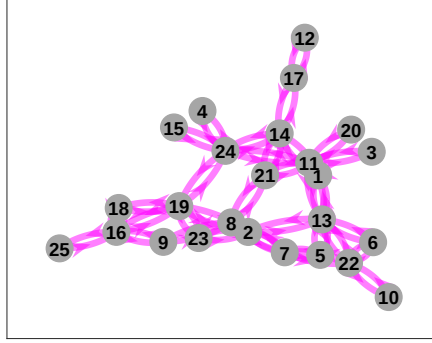


Figure 2.14: A communication network represented by a 25 node digraph.

domly for every node. Updates (2.67), (2.68) and (2.70) are implemented in MATLAB and simulated. 2-norm of each node's ratio state is plotted in Figure 2.15 achieving convergence in 60 iterations.

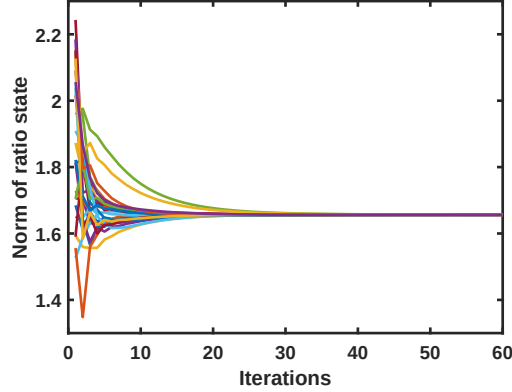


Figure 2.15: 2-norm of 10-dimensional ratio states of all the nodes (25) in the network.

Algorithm 2 is also implemented in MATLAB and the radius $\overline{R}_i(l)$ for all $i \in \mathcal{V}$ is plotted in Figure 2.16. It can be seen that the radius comes under some pre-specified tolerance (0.0166, 1% of the norm of the consensus vector) within 60 iterations and is used as a stopping criterion by each node. Figure 2.17 plots the two dimensional projection of the *norm* ball $B\{\overline{R}_i(l), r_i(l\mathcal{D} + \mathcal{D})\}$ for node 1 as l progresses over time. As expected, with the increase in l , balls shrink in size. A similar observation is seen for all the other nodes as well. Thus, we have established that Algorithm 2 can be successfully used as a finite-time termination criterion for distributed ratio vector consensus.

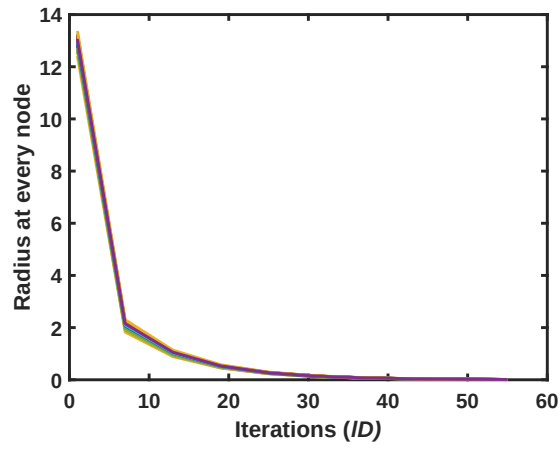


Figure 2.16: Radius $\overline{R}_i(l)$ at each node.

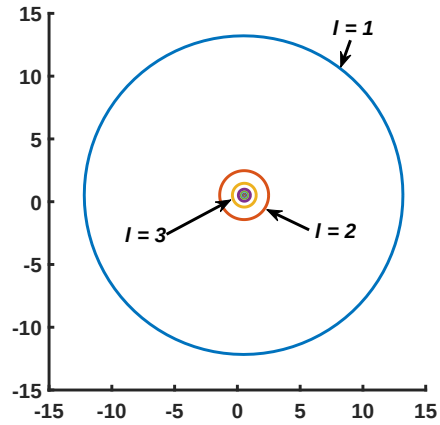


Figure 2.17: 2-dimensional projection of norm balls for node 1 with changing l .

Chapter 3

Resource Management in Networked Dynamical Systems: Distributed Optimization Over Networks

3.1 Gist of the Chapter

Distributed optimization has become a crucial approach in modern systems due to the limitations of centralized optimization methods in large-scale, decentralized environments. This approach allows for efficient, scalable, and robust solutions by distributing decision-making across multiple agents. These agents, equipped with local information, collaborate to solve complex optimization problems in networked systems, multi-agent coordination, and decentralized machine learning. Research in this field has led to the development of primal and dual methods for unconstrained and constrained optimization problems. However, challenges remain in achieving practical feasibility, especially when agents' estimates do not converge quickly enough for real-world applications like multi-robot rendezvous or decentralized deep learning. This chapter discusses these challenges, focusing on mismatched estimates between agents and the need for algorithms that guarantee feasible solutions within finite iterations.

Early approaches focused on primal and dual methods for unconstrained and constrained problems. However, existing algorithms often struggle with agent estimate

mismatches and finite-time feasibility, making them unsuitable for time-sensitive applications like robot coordination and distributed learning. This chapter aims to improve convergence and ensure practical solutions through finite-iteration algorithms and less restrictive assumptions. It further develops an efficient constrained distributed optimization algorithm known as **DC-DistADMM** algorithm.

The research work presented in this chapter is published in the articles [124], [127]–[129].

3.2 State-of-the-art on Distributed Optimization Methods

Distributed optimization emerges as a compelling paradigm in response to the increasingly complex and interconnected nature of modern systems. Traditional centralized optimization methods, while effective in many contexts, often face significant challenges when applied to large-scale networks or decentralized systems. These challenges include issues of scalability, communication overhead, and privacy concerns. In contrast, distributed optimization offers a decentralized approach where decision-making and computation are distributed among multiple agents or nodes, each possessing local information and decision-making capabilities. By harnessing the collective intelligence of distributed entities, this paradigm enables efficient, scalable, and robust optimization solutions across a wide range of applications. From networked systems and multi-agent coordination to decentralized machine learning and resource allocation, distributed optimization provides a powerful framework for addressing real-world challenges in a distributed and collaborative manner. The evolution towards distributed optimization has spurred a wealth of research aimed at developing innovative algorithms and protocols tailored to the unique demands of decentralized systems. In the sequel, we delve into the rich tapestry of literature surrounding distributed optimization, exploring its foundational framework and state-of-the-art algorithms.

Consider a group of n agents connected through a directed graph, $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the set of vertices and directed edges respectively. Each agent can transmit information to other agents restricted by the directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; an agent i can transmit to agent j if a directed link $i \rightarrow j$ exists in $\mathcal{E}$. The agents focus on solving the

following distributed optimization problem:

$$\begin{aligned}
& \underset{\tilde{x} \in \mathbb{R}^p}{\text{minimize}} && \tilde{f}(\tilde{x}) = \sum_{i=1}^n \tilde{f}_i(\tilde{x}) \\
& \text{subject to} && C_i \tilde{x} = c_i, \ D_i \tilde{x} \leq d_i, \ \forall i \in \mathcal{V}, \ \tilde{x} \in \bigcap_{i=1}^n \mathcal{X}_i,
\end{aligned} \tag{3.1}$$

where, $x \in \mathbb{R}^p$ is a global optimization variable. $\tilde{f}_i : \mathbb{R}^p \rightarrow \mathbb{R}$ is the local objective function of agent i . $\mathcal{X}_i$ is a convex constraint set associated with the variables of agent i . $C_i \tilde{x} = c_i$ with $C_i \in \mathbb{R}^{m_1 \times p}$, $c_i \in \mathbb{R}^{m_1}$ and $D_i \tilde{x} \leq d_i$ with $D_i \in \mathbb{R}^{m_2 \times p}$, $d_i \in \mathbb{R}^{m_2}$ are the local equality and inequality constraints of agent i . Define $\tilde{x}_s$ to be a slack variable and re-write the inequality constraint $D_i \tilde{x} \leq d_i$ as,

$$\begin{bmatrix} D_i & \mathbf{I} \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{x}_s \end{bmatrix} = d_i, \ \tilde{x}_s \succeq 0, \ \forall i \in \mathcal{V},$$

where, $\mathbf{I}$ is the identity matrix of appropriate dimension. Let $x = [\tilde{x}^\top \ \tilde{x}_s^\top]^\top \in \mathbb{R}^{(m+p)}$. Given a set S , the indicator function of set S is defined as:

$$\mathcal{I}_S(x) = \begin{cases} 0 & \text{if } x \in S \\ +\infty & \text{otherwise.} \end{cases} \tag{3.2}$$

Using the definition of the indicator function of $\mathbb{R}_{\geq 0}^m$ problem (3.1) can be rewritten as:

$$\begin{aligned}
& \underset{x \in \mathbb{R}^p}{\text{minimize}} && \sum_{i=1}^n [\tilde{f}_i(\tilde{x}) + \mathcal{I}_{\mathbb{R}_{\geq 0}^m}(\tilde{x}_s)] \\
& \text{subject to} && A_i x = b_i, \ \forall i \in \mathcal{V}, \ \tilde{x} \in \bigcap_{i=1}^n \mathcal{X}_i,
\end{aligned} \tag{3.3}$$

where, $A_i := \begin{bmatrix} C_i & 0 \\ D_i & I \end{bmatrix}$, $b_i := \begin{bmatrix} c_i \\ d_i \end{bmatrix}$ ¹. Problem (3.3) is recast by creating local copies x_i for all $i \in \mathcal{V}$, of the global variable x and imposing the agreement of the solutions of all the agents via consensus constraint. This leads to an equivalent formulation of (3.3) as

¹without loss of generality here we assume $m_1 = m_2 = m$.

described below:

$$\begin{aligned}
& \text{minimize} && \sum_{i=1}^n [\tilde{f}_i(\tilde{x}_i) + \mathcal{I}_{\mathbb{R}_{\geq 0}^m}(\tilde{x}_{s_i})] \\
& \text{subject to} && A_i x_i = b_i, \tilde{x}_i \in \mathcal{X}_i, \forall i \in \mathcal{V}, \quad x_i = x_j, \forall i, j \in \mathcal{V}.
\end{aligned} \tag{3.4}$$

The constraints $\tilde{x}_i \in \mathcal{X}_i$ can be integrated into the objective function using the indicator functions of the sets $\mathcal{X}_i$, as:

$$\begin{aligned}
& \text{minimize} && \sum_{i=1}^n f_i(x_i) \\
& \text{subject to} && A_i x_i = b_i, \forall i \in \mathcal{V}, \quad x_i = x_j, \forall i, j \in \mathcal{V},
\end{aligned} \tag{3.5}$$

where, $f_i(x_i) := \tilde{f}_i(\tilde{x}_i) + \mathcal{I}_{\mathbb{R}_{\geq 0}^m}(\tilde{x}_{s_i}) + \mathcal{I}_{\mathcal{X}_i}(\tilde{x}_i)$ and $\mathcal{I}_{\mathcal{X}_i}$ is the indicator function of set $\mathcal{X}_i$.

$$\text{Let } \mathcal{C}_\eta := \{\mathbf{y} = [y_1^\top \dots y_n^\top]^\top \in \mathbb{R}^{n(m+p)} \text{ such that } \|y_i - y_j\| \leq 2\eta, 1 \leq i, j \leq n\}, \tag{3.6}$$

to be a set of vectors $\mathbf{y} \in \mathbb{R}^{n(m+p)}$ such that the norm of the difference between any two $(m+p)$ dimensional sub-vectors of $\mathbf{y}$ is less than η . Using definition of set $\mathcal{C}_\eta$, (3.5) is equivalent to:

$$\begin{aligned}
& \text{minimize} && \sum_{i=1}^n f_i(x_i) \\
& \text{subject to} && A_i x_i = b_i, \forall i \in \mathcal{V}, \quad \mathbf{x} = [x_1^\top x_2^\top \dots x_n^\top]^\top, \quad \mathbf{x} = \mathbf{y}, \quad \mathbf{y} \in \mathcal{C}_0.
\end{aligned} \tag{3.7}$$

Problem (3.7) can be reformulated as:

$$\begin{aligned}
& \text{minimize} && \sum_{i=1}^n f_i(x_i) + \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}) \\
& \text{subject to} && A_i x_i = b_i, \forall i \in \mathcal{V}, \quad \mathbf{x} = [x_1^\top x_2^\top \dots x_n^\top]^\top, \quad \mathbf{x} = \mathbf{y},
\end{aligned} \tag{3.8}$$

where, $\mathcal{I}_{\mathcal{C}_0}$ is the indicator function of the set $\mathcal{C}_0$. The unconstrained version of the problem (3.8),

$$\text{minimize}_{x \in \mathbb{R}^p} f(x) = \sum_{i=1}^n f_i(x), \tag{3.9}$$

has received significant attention in the literature. Note, here $x \in \mathbb{R}^p$ is a global optimization variable, and each function $f_i : \mathbb{R}^p \rightarrow \mathbb{R}$ is a convex function known only to agent i .

Many problems in various engineering fields such as wireless systems, multi-agent coordination and control [130], and machine learning [21] can be posed in the form of problem (3.1) and (3.9). Early works on the distributed optimization problem can be found in seminal papers [29], [131]. Current approaches for solving optimization problems can be broadly classified as: (i) primal methods, that update the agent estimates of the solution by utilizing a step size rule based on the gradient information (at the current estimate) of the objective function to steer towards the optimal solution, and (ii) dual based optimization methods that employ Lagrange multipliers. For the unconstrained optimization problems the works [50], [116], [120], [132]–[140] (and references therein) take a primal-based approach, whereas [141]–[148] take a dual-based approach. The primal approaches utilize the distributed gradient descent (DGD) method [50], [132] and its modifications [133]–[137]. The authors in [134] proposed an exact first-order method that converges to the optimal solution while using a fixed step size in the gradient updates. The works in [136], [137], [146] have developed algorithms based on proximal-gradient to tackle (3.9). Motivated by the advantages of parallel operation and good convergence results (see [149]) of dual-based methods, articles [142]–[148] adopt the alternating direction method of multipliers (ADMM) dual-based approach; the reference [142] proposed ADMM based algorithm for convex objective functions with a sublinear rate of convergence, [143], [144] established linear rate of convergence with globally strongly convex objective functions.

The works in [150]–[164] consider constrained distributed optimization problems similar to (3.1). For solving constrained convex optimization problems of the form (3.1) distributively, many of the existing state-of-the-art algorithms [150]–[161] utilize primal algorithms based on projections onto the constraint set and the (sub)-gradient information of the individual objective functions. Here, [155]–[157], [159], [162]–[164] focus only on set constraints; equality constraints are considered in [154], [158] while inequality constraints are the focus in [150]–[154], [160], [161]. To establish convergence, the article [153] assumes the differentiability of the individual objective functions. The assumption of individual functions being Lipschitz continuous and convex quadratic over the constraint set is required in [156]. Articles [157] and [160] require the assumption of the set on which the gradient of the individual functions is zero to be bounded and a compact optimal solution set respectively. Twice continuous differentiability and

bounded Hessian matrix for the individual functions is assumed in [158]. The articles [159], [161], require bounded (sub)-gradients. Articles [150]–[161] establish convergence to an optimal solution; however, no convergence rate/iteration complexity estimates are determined. References [162]–[164] utilize a primal-dual approach for constrained problems. Article [163] requires bounded (sub)-gradients to establish an $O(k^{-1/4})$ (k is the iteration counter) rate of convergence for the objective function residual and article [164] does not establish the rate of convergence in the presence of constraints. As alluded to earlier, ADMM based approach (which is a Lagrangian dual-based approach) has several advantages ([141]); however, results on distributed constrained optimization problems using ADMM are sparse. [162] which considers a dual ADMM-like method does not provide a convergence rate/iteration complexity analysis. The algorithms in [150], [151], [153]–[156], [161], [163], [164] are continuous-time algorithms that pose the need for further analysis under discretization needed for implementation.

Most existing algorithms for unconstrained optimization [29], [50], [131]–[137], [141]–[148], [165], [166] and constrained optimization [150]–[160], [162]–[164], are designed using centralized information and under restrictive assumptions of bi-directional (or undirected) communication networks and doubly-stochastic weight matrices. The first work to propose a distributed optimization method for directed graphs appeared in the context of unconstrained optimization problems [167]. Subsequently, the authors in [120] incorporated the push-sum protocol [42] with the earlier DGD method [50] to propose the subgradient-push algorithm for unconstrained distributed optimization over time-varying directed graphs. Recent works [168] and [140] proposed distributed optimization schemes for directed networks that use row and column stochastic matrices in one algorithm iteration.

3.3 Research Questions

Research Focus 1:

While there is proliferating literature on developing distributed optimization methods for (3.9), the current methods suffer from the mismatch between the agents' estimates at any finite-time termination of the algorithm. Consider, an equivalent reformulation

of problem (3.9):

$$\begin{aligned} & \underset{x_1, \dots, x_n}{\text{minimize}} \quad \sum_{i=1}^n f_i(x_i) \\ & \text{subject to } x_i = x_j, \text{ for all } i, j \in \{1, 2, \dots, n\}, \end{aligned} \quad (3.10)$$

where, $x_i, x_j \in \mathbb{R}^p$ are local estimates (of the optimal solution) of agent i and j respectively and the equality constraints enforce agreement between the local estimates of the agents. Existing algorithms' designs focus on finding the solution to problem (3.10) emphasizing solving the unconstrained problem and reaching an agreement between the agents' estimates only asymptotically. In particular, at the intermediate iterations of the algorithm, the agents' estimates often have a considerable mismatch and hence, do not provide a practically feasible solution (where a specification is present to have control over the largest mismatch between the agent estimates) to problem (3.10). The disagreement of agents' estimates (also called consensus constraint violation) can be problematic for practical applications where obtaining a feasible solution with guarantees in finite time is crucial for adequate performance and stability of the system. Some examples of such applications include

A. Rendezvous of Multi-Robot Systems: Consider n mobile robots that have to rendezvous at a common position and a common time. The i^{th} robot has a hard constraint characterized by $u_i(p, t) \leq 0$ for a position p and time t . This constraint, for example, can impose restrictions such as not being able to reach position p by time t . The i^{th} robot also poses a cost $f_i(p, t)$ to the system planner for reaching position p at time t . The related optimization problem is:

$$\begin{aligned} & \underset{(p_1, t_1), \dots, (p_n, t_n)}{\text{minimize}} \quad \sum_{i=1}^n f_i(p_i, t_i) \\ & \text{subject to } p_i = p_j, t_i = t_j, \forall i, j \in \{1, 2, \dots, n\} \\ & \quad u_i(p_i, t_i) \leq 0, \forall i \in \{1, 2, \dots, n\} \end{aligned} \quad (3.11)$$

The optimal solution to the above optimization can be realized; however, the optimal with the current know-how [27], [169] can be reached only asymptotically. It is possible that the robots cannot wait beyond a certain time and need to start taking action before a pre-specified time. In such scenarios, algorithms that do not satisfy the common meeting time and position may remain practically infeasible for the operation when

stopped before convergence to the optimal is close (which may be hard to assess).

B. Decentralized Deep Learning: Consider, a distributed empirical risk minimization problem with n nodes [170], [171]. In this problem, the task is to obtain the best estimate x of a parameter of interest, from data. The estimation is carried out by n distributed agents communicating over a given network. Agent i has a data set D_i assigned to it. Agent i has a cost function to quantify the quality of fit when the parameter x is used to fit a data z in D_i given by $f_i(x, z)$ (for example, one such function is $\|x - z\|^2$). All agents should agree on the parameter estimate at the end of the learning process. The associated optimization problem (also see [170], [171]) is

$$\begin{aligned} & \underset{x_1, \dots, x_n}{\text{minimize}} && \sum_{i=1}^n \sum_{z \in D_i} f_i(x_i, z) \\ & \text{subject to} && x_i = x_j, \forall i, j \in \{1, 2, \dots, n\} \end{aligned} \tag{3.12}$$

In the context of problem (3.12), it is important to balance the individual model parameter optimality (via local optimization by the i^{th} agent) and the global constraint of consensus between the model parameter estimates of all the agents during the model parameter learning procedure (typically the neural network training procedure) to recover an estimate of the true model from the distributed datasets [170], [171]. Here, a framework to improve the inter-iteration behavior of the estimates to agree is important (see [172]). More applications requiring finite-iteration feasibility arise in power systems, where the reactive power control, economic dispatch, and optimal power flow problems are modeled as distributed optimization problems [173], [174].

Due to the mismatch between the agent estimates, at any finite iteration, the solution obtained by the existing algorithms is not a feasible solution to problem (3.10). In applications where getting a feasible solution is critical during every time-instant of operation (a few examples are presented above), the existing algorithms do not provide a viable solution. Thus, a new perspective on balancing/reducing the disagreement between the estimates of all agents during the algorithm iterations is needed. The current thesis chapter addresses this issue.

Research Focus 2:

Although methods for solving constrained distributed optimization exist, different algorithms lack in one or more aspects:

- (i) Most existing algorithms [150]–[160], [162]–[164], are designed using centralized information and under restrictive assumptions of bi-directional (or undirected) communication networks and doubly-stochastic weight matrices.
- (ii) Articles [155]–[157], [159], [162]–[164] focus only on set constraints; whereas only equality constraints are considered in [154], [158]. Inequality constraints are the focus in [150]–[154], [160], [161].
- (iii) Restrictions like the assumption on the differentiability of all the individual objective functions ([153]), the requirement of the gradient of the individual functions to be zero over a bounded set ([157]), the necessity of a compact optimal solution set ([160]), availability of individual functions with twice continuous differentiability and a bounded Hessian matrix ([158]) and bounded (sub)-gradients ([159], [161]) are present that doesn't provide a general analysis for the constrained distributed optimization problems that appear in applications.
- (iv) Finite-iteration based convergence guarantees for the constrained distributed optimization problems are not present in the literature, the state-of-the-art [150]–[161] establish an asymptotic convergence to an optimal solution.

Thus, a constrained optimization algorithm that can be synthesized distributive without centralized coordination and is applicable for general non-linear constrained distributed optimization problems is needed. Further, the analysis tools presented should resemble a bigger class of problems with less restrictive assumptions. The current thesis chapter focuses on developing a constrained distributed optimization algorithm with the above characteristics and strong finite-iteration convergence guarantees.

3.4 Contributions of the Chapter

The current chapter makes the following contributions toward the two research focuses outlined above:

3.4.1 Contributions Towards Research Focus 1:

1. We introduce a new framework called *optimize then agree* for solving distributed optimization problems. Our approach is to decouple the consensus and optimization steps, unlike the existing distributed optimization schemes to reduce the inaccuracy of the consensus step in the existing distributed optimization algorithms. Our framework provides a new perspective on solving distributed optimization

problems and presents guidelines for developing new algorithms. Based on this framework we develop a first-order distributed algorithm termed **GradConsensus** to solve distributed optimization problems.

2. The proposed **GradConsensus** algorithm has several nice features:
 - i) It is suitable for directed graphs allowing for non-symmetric communication topologies. This feature considerably widens the applicability as in most application scenarios, agents do not have the same range of communication.
 - ii) It utilizes *only* the knowledge of the out-degree of each agent in the graph. It is based on a column stochastic matrix which is amenable for distributed synthesis where centralized information of the network connectivity structure is not required.
 - iii) To the best of the author's knowledge, the **GradConsensus** algorithm has the best convergence rate under the assumption of individual functions f_i being not necessarily strongly convex and Lipschitz differentiable. We show that the **GradConsensus** algorithm converges at a novel Q-linear rate to a small neighborhood of the optimal solution. Most existing analyses of the first-order gradient-based distributed optimization algorithms under these assumptions have a sub-linear rate of convergence. The improved guarantees accrue with an overhead of within $\log k$ communication steps of the optimal $O(k)$ bound by the k^{th} iterate. The **GradConsensus** algorithm has a global R-linear rate of convergence to the optimal solution when the aggregate function is strongly-convex and Lipschitz differentiable.
 - iv) Empirical results corroborate that the proposed **GradConsensus** algorithm performs well in practice for the number of gradient computations and the total run-time required.

3.4.2 Contributions Towards Research Focus 2:

1. We develop an ADMM based algorithm called Directed Constrained-Distributed Alternating Direction Method of Multipliers (**DC-DistADMM**) that solves constrained distributed optimization problem (3.1).
2. **DC-DistADMM** is a Lagrangian dual-based algorithm with the following features:

- i) **DC-DistADMM** accommodates directed graphs allowing for non-symmetric communication topologies. This feature considerably widens the applicability as in most application scenarios, agents do not have the same range of communication.
 - ii) **DC-DistADMM** is amenable to distributed synthesis scenarios and extends the applicability of ADMM method to applications where a plug-and-play operation is required [16], [17]. The **DC-DistADMM** algorithm has column stochastic updates and does not require the agents to know global network interconnection information.
 - iii) To the best of the author's knowledge **DC-DistADMM** has the best convergence rates in the constrained optimization literature. The **DC-DistADMM** algorithm converges to an optimal solution of problem (3.1) under mild assumptions. The algorithm has a $O(1/k)$ rate of convergence, where k is the iteration counter, when the individual functions are convex but not-necessarily differentiable. It has a geometric rate of convergence to any arbitrary neighborhood of the optimal solution when the objective functions are smooth and restricted strongly convex at the optimal solution of problem. **DC-DistADMM** does not require the individual functions to be differentiable for convergence to an optimal solution neither bounded sub-gradients and globally strongly convex nor twice continuously differentiable is assumed in the analysis.
3. ADMM based approaches and more generally Lagrangian dual-based approaches for distributed constrained problems involve, in each iteration, an information mixing step among agents and a step that improves the objective. An important technical contribution of the current thesis chapter is the transformed problem (3.8)) that allows for carrying out almost *complete* mixing via *any* consensus strategy for the information mixing step to be followed by the objective improvement step.
 4. The total number of communication steps for **DC-DistADMM** by the k^{th} iterate is within a factor of $\log k$ of the optimal lower bound in obtaining a $O(1/k)$ rate of convergence. Empirical data corroborates that **DC-DistADMM**, in the computational performance, outperforms other distributed constrained and unconstrained optimization approaches.

3.5 Developed Solutions

3.5.1 GradConsensus: Linearly Convergent Algorithm for Reducing Disagreement in Multi-agent Optimization

The Optimize then Agree Framework:

In this section, we focus on problem (3.10) and present the proposed *optimize then agree* framework. We emphasize that the proposed framework can be applied to any existing algorithm in the literature that utilizes an optimization scheme in tandem with a consensus step for state updates. Here we utilize the classical DGD method [50] for an explanation. During the iteration k of DGD every agent i updates its local estimate $x_i(k) \in \mathbb{R}^p$ as follows:

$$x_i(k) = p_{ii}x_i(k-1) + \sum_{j \in \mathcal{N}_i^-} p_{ij}x_j(k-1) - \alpha \nabla f_i(x_i(k-1)), \quad (3.13)$$

where, the weights $0 \leq p_{ij} \leq 1$ are such that $\sum_j p_{ij} = \sum_i p_{ij} = 1$. The first two terms $p_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i^-} p_{ij}x_j(k-1)$ in (3.13) corresponds to a local consensus step and the second term $-\alpha \nabla f_i(x_i(k-1))$ denotes a local gradient step. Notice that due to a local consensus step the update (3.13) does not account for the agreement between the estimates over the entire network of agents during iteration k . If update (3.13) is terminated after some finite number of iterations the solution estimates disagree and do not satisfy the equality constraints in problem (3.10) and hence the (finite-time) solution generated is not feasible for the original problem. To address this issue we propose the *optimize then agree* framework where the optimization step (gradient-descent in this case) and the consensus scheme are decoupled from each other. The optimize then agree framework enforces the mismatch between the agent estimates to remain below any specified threshold (and hence controls the consensus constraint violation). Here, each agent employs an optimization scheme to steer towards the solution of its local sub-problem and utilizes a consensus protocol to achieve agreement with the local estimates of all the other agents. To describe the *optimize then agree* framework we define, the optimization scheme operator used by agent $i \in \mathcal{V}$ as the mapping $\mathcal{O}_i : \mathbb{R}^p \rightarrow \mathbb{R}^p$:

$$z_i(k) := \mathcal{O}_i(x_i(k-1)). \quad (3.14)$$

Similarly, define the consensus scheme as the operator mapping $\mathcal{C} : \mathbb{R}^{p \times n} \rightarrow \mathbb{R}^{p \times n}$:

$$\mathbf{x}(k) := \mathcal{C}(\mathbf{z}(k)), \quad (3.15)$$

where, $\mathbf{z}(k) = [z_1(k) \dots z_n(k)]$, $\mathbf{x}(k) = [x_1(k) \dots x_n(k)] \in \mathbb{R}^{p \times n}$, such that $x_1(k) = x_2(k) = \dots = x_n(k)$. Utilizing the above notation the *optimize then agree* framework leads to:

Algorithm 3: Optimize then agree framework

```

Repeat for  $k = 1, 2, \dots$ 
    /* state updates using operators (3.14) and (3.15) */
     $z_i(k) = \mathcal{O}_i(x_i(k-1)), \forall i \in \mathcal{V}$ 
     $\mathbf{x}(k) = \mathcal{C}(\mathbf{z}(k))$ 
until a stopping criterion is met

```

Choosing first-order gradient descent with a constant step-size α as the optimization scheme, Algorithm 3 at any iteration $k + 1$ results in the following updates:

$$\begin{aligned} z_i(k+1) &= x_i(k) - \alpha \nabla f_i(x_i(k)), \forall i \in \mathcal{V}, \\ \mathbf{x}(k+1) &= \mathcal{C}(\mathbf{z}(k+1)), \end{aligned} \quad (3.16)$$

with the consensus operator implemented in a centralized manner (with the entire vector $\mathbf{z}(k+1)$ as the input). However, to make Algorithm 3 applicable to distributed networks, we need to distributively realize (3.16). To this end, one natural choice is to employ a distributed average consensus protocol [42]. Although the average consensus protocols lead to agreement among the agents to the initial average, the result holds only asymptotically which is a potential problem for an efficient implementation of Algorithm 3. To alleviate this, the consensus scheme operator is chosen to be a distributed finite-time approximate consensus protocol, which we call the ε -consensus protocol. The ε -consensus protocol produces a desired level of agreement among the estimates of all agents which is determined by a parameter $\varepsilon > 0$ independent of problem (3.9) that can be chosen arbitrarily small. In the following section, we first discuss the ε -consensus protocol and then introduce the proposed **GradConsensus** algorithm.

Remark 14. *We emphasize that the updates in Algorithm 3 are general and do not depend on a particular choice of the optimization scheme or the consensus algorithm, thus the optimize then agree framework provides a guideline for further exploration of*

distributed optimization algorithms. The framework of Algorithm 3 can also be seen as an outer optimization loop and an inner consensus loop structure algorithm. Thus, existing schemes [175]–[178] are special cases of the optimize then agree framework with a particular choice of optimization scheme and the number of communication steps.

ε -consensus Protocol:

The ε -consensus protocol is a distributed finite-time terminated average consensus algorithm for agreement on vector-valued states. Here, each agent $i \in \mathcal{V}$ has a vector $z_i \in \mathbb{R}^p$. The objective is to find an approximate estimate of the average $\bar{z} = \frac{1}{n} \sum_{i=1}^n z_i \in \mathbb{R}^p$. To achieve this objective an iterative procedure is devised where each agent $j \in \mathcal{V}$ maintains a state variable $s_j(k) \in \mathbb{R}^p$ and an auxiliary variable $t_j(k) \in \mathbb{R}$, with the following initialization: $s_j(0) = z_j$, $t_j(0) = 1$, for all $j \in \mathcal{V}$. Every agent $j \in \mathcal{V}$ updates its state and auxiliary variables according to:

$$s_j(k+1) = p_{jj}s_j(k) + \sum_{\ell \in \mathcal{N}_j^-} p_{j\ell}s_\ell(k) \quad (3.17)$$

$$t_j(k+1) = p_{jj}t_j(k) + \sum_{\ell \in \mathcal{N}_j^-} p_{j\ell}t_\ell(k) \quad (3.18)$$

$$r_j(k+1) = \frac{1}{t_j(k+1)}s_j(k+1). \quad (3.19)$$

We make the following assumption on the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and the associated matrix $\mathbf{P} = [p_{ij}]$:

Assumption 9. *The directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is strongly-connected. The associated weighted adjacency matrix $\mathbf{P} = [p_{ij}]$ is a column-stochastic matrix. In particular, $0 \leq p_{ij} \leq 1$ and $\sum_{i=1}^n p_{ij} = 1$ for all $j \in \mathcal{V}$.*

One choice of weights that satisfy Assumption 9 is the equal neighbor weights rule [51]. The convergence of the state $r_j(k) := \frac{1}{t_j(k)}s_j(k)$, $j \in \mathcal{V}$ evolving according to (3.17)-(3.19), to the average $\bar{z} = \frac{1}{n} \sum_{i=1}^n z_i$ has been established rigorously [42], [179]. We state the following result for updates (3.17)-(3.19):

Theorem 15 ([179]). *Let Assumption 9 and the updates (3.17)-(3.19) hold. Let the initial conditions of the state and auxiliary variables s_j and t_j be $s_j(0) = z_j$, $t_j(0) = 1$, for all $j \in \mathcal{V}$. Then $\lim_{k \rightarrow \infty} r_j(k) = \frac{1}{n} \sum_{i=1}^n z_i$, for all $j \in \mathcal{V}$.*

We now provide a criterion for determining when consensus within a tolerance $\varepsilon > 0$ is reached. Let $\mathbf{r}(k) = [r_1(k) \ r_2(k) \ \dots \ r_n(k)] \in \mathbb{R}^{p \times n}$. At any iteration k , define the maximum $\bar{\mathbf{r}}(k)$ and minimum $\underline{\mathbf{r}}(k)$ state variable of the network over all the agents as:

$$\bar{\mathbf{r}}(k) := \left[\max_{1 \leq j \leq n} \{\mathbf{r}_{[1j]}(k)\} \dots \max_{1 \leq j \leq n} \{\mathbf{r}_{[pj]}(k)\} \right] \quad (3.20)$$

$$\underline{\mathbf{r}}(k) := \left[\min_{1 \leq j \leq n} \{\mathbf{r}_{[1j]}(k)\} \dots \min_{1 \leq j \leq n} \{\mathbf{r}_{[pj]}(k)\} \right], \quad (3.21)$$

where, $\mathbf{r}_{[ij]}(k)$ is i^{th} -row and j^{th} -column entry of $\mathbf{r}(k)$. Let $\{\bar{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ and $\{\underline{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ denote sub-sequences of the sequences $\bar{\mathbf{r}}(k)$ and $\underline{\mathbf{r}}(k)$ respectively, sampled at iterations of the form $u\mathcal{D}$, where, $u = 1, 2, \dots$ and $\mathcal{D}$ is an upper bound on the diameter of the graph $\mathcal{G}$. Under Assumption 9, sub-sequences $\bar{\mathbf{r}}(u\mathcal{D})$ and $\underline{\mathbf{r}}(u\mathcal{D})$ converge to the average of the initial values, $\bar{z} = \frac{1}{n} \sum_{i=1}^n z_i$. Further, the sub-sequences $\bar{\mathbf{r}}(u\mathcal{D})$ and $\underline{\mathbf{r}}(u\mathcal{D})$ are strictly monotonically decreasing and increasing in the index u (see [179], Theorem 4.1).

To distributively determine sub-sequences $\{\bar{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ and $\{\underline{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ each agent $j \in \mathcal{V}$, maintains two variables $M_j(k), m_j(k) \in \mathbb{R}^p$ at any iteration k that are initialized as

$$M_j(0) = m_j(0) = z_j, \quad \forall j \in \mathcal{V}, \quad (3.22)$$

and updated as:

$$M_j(k+1) = \left[\max_{\ell \in \mathcal{N}_j^- \cup \{j\}} M_{[\ell 1]}(k) \dots \max_{\ell \in \mathcal{N}_j^- \cup \{j\}} M_{[\ell p]}(k) \right] \quad (3.23)$$

$$m_j(k+1) = \left[\min_{\ell \in \mathcal{N}_j^- \cup \{j\}} m_{[\ell 1]}(k) \dots \min_{\ell \in \mathcal{N}_j^- \cup \{j\}} m_{[\ell p]}(k) \right], \quad (3.24)$$

where, $M_{[\ell i]}(k)$, and $m_{[\ell i]}(k)$ are the i^{th} entries of the vectors $M_\ell(k)$ and $m_\ell(k)$ respectively. It can be rigorously established that (see [114], Proposition 2.3) the variables $M_j(k)$ and $m_j(k)$ for all $j \in \mathcal{V}$, initialized via the rule (3.22) converge to $\bar{\mathbf{r}}(0)$ and $\underline{\mathbf{r}}(0)$ respectively in finite number of iterations $k_f \leq \mathcal{D}$.

The above discussion leads to a method for finding the sequences $\{\bar{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ and $\{\underline{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ distributedly: re-initializing the updates (3.23), (3.24) for all agents $j \in \mathcal{V}$, at every iteration of the form $k = u\mathcal{D}, u = 1, 2, \dots$, to the variable $r_j(u\mathcal{D})$ allow the variables M_j and m_j to converge to $\bar{\mathbf{r}}((u-1)\mathcal{D})$ and $\underline{\mathbf{r}}((u-1)\mathcal{D})$ respectively, after a finite number of iterations $k_f \leq \mathcal{D}$. Since, both the sequences $\{\bar{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$ and $\{\underline{\mathbf{r}}(u\mathcal{D})\}_{u \geq 1}$

converge to $\bar{z}$, the norm of the difference between the two, i.e., $\|\bar{\mathbf{r}}(u\mathcal{D}) - \mathbf{r}(u\mathcal{D})\|$ also converges to zero. Therefore, given $\varepsilon > 0$, the quantity $\|\bar{\mathbf{r}}(u\mathcal{D}) - \mathbf{r}(u\mathcal{D})\|$ eventually becomes less than ε . Based on the above observations we propose an algorithm that allows all agents to simultaneously converge to an ε -close estimate of $\bar{z}$ with ε being any arbitrary pre-specified tolerance:

For $u = 1, 2, \dots$, let $\gamma^j(u\mathcal{D}) := M_j(u\mathcal{D}) - m_j(u\mathcal{D})$. Each agent computes the norm $\|\gamma^j(u\mathcal{D})\|$ every $\mathcal{D}$ iterations and compares it with ε . If $\|\gamma^j(\tilde{u}\mathcal{D})\| := \|M_j(\tilde{u}\mathcal{D}) - m_j(\tilde{u}\mathcal{D})\| < \varepsilon$ at any iteration $\tilde{u}\mathcal{D}$, then agent j , knowing its estimate $r_j(\tilde{u}\mathcal{D})$ is ε -close to $\bar{z}$, terminates the consensus protocol updates (3.17)-(3.19) at $\tilde{u}\mathcal{D}$. The variable $r_j(\tilde{u}\mathcal{D})$ is the ε -close estimate of $\bar{z}$ available with agent j .

The following Lemma provides the number of communication steps required by the ε -consensus protocol to converge to an ε -close solution.

Lemma 12. *Given, consensus tolerance $\varepsilon > 0$, under Assumption 9, after $k \geq \left\lceil \frac{\log(1/\varepsilon)}{-\log(\lambda)} + \frac{\log(C\|\mathbf{s}(0)\|_1/\delta)}{-\log(\lambda)} \right\rceil$ iterations of the consensus protocol (3.17)-(3.19) we have:*

$$\left\| r_i(k) - \frac{1}{n} \sum_{l=1}^n s_l(0) \right\| \leq \varepsilon, \quad \forall i \in \mathcal{V},$$

where, $\mathbf{s}(0) := [s_1(0)^\top s_2(0)^\top \dots s_n(0)^\top]^\top \in \mathbb{R}^{n \times p}$ and $C, \delta, \lambda \in (0, 1)$ are constants related to the properties of the matrix $\mathbf{P}$ and graph $\mathcal{G}$.

Proof. Using (3.17) it is easy to see that, $k \geq 0$,

$$\mathbf{s}(k) = \mathbf{P}^k \mathbf{s}(0), \quad (3.25)$$

where, $\mathbf{s}(k) := [s_1(k)^\top s_2(k)^\top \dots s_n(k)^\top]^\top \in \mathbb{R}^{n \times p}$. Since, $\mathbf{P}$ is column stochastic, we have,

$$\mathbf{1}^\top \mathbf{s}(k) = \mathbf{1}^\top \mathbf{s}(0). \quad (3.26)$$

From Corollary 2, [120] there exists stochastic vector $\phi_k \in \mathbb{R}^n$ and constants C_1 and $\lambda \in (0, 1)$ such that for all $i, j = 1, \dots, n$,

$$|\mathbf{P}_{ij}^k - \phi_{kj}| \leq C_1 \lambda^k, \quad \text{for all } k \geq 0. \quad (3.27)$$

From (3.25) and (3.26) we have,

$$\begin{aligned}
(\mathbf{P} - \phi_k \mathbf{1}^\top) \mathbf{s}(k) &= (\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{s}(0) \\
\implies \mathbf{P} \mathbf{s}(k) &= \phi_k \mathbf{1}^\top \mathbf{s}(k) + (\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{s}(0) \\
\implies \mathbf{s}(k+1) &= \phi_k \mathbf{1}^\top \mathbf{s}(k) + (\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{s}(0)
\end{aligned} \tag{3.28}$$

Similarly, for (3.18) we can write,

$$\mathbf{t}(k+1) = \phi_k \mathbf{1}^\top \mathbf{t}(k) + (\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{t}(0), \tag{3.29}$$

where, $\mathbf{t}(k) := [t_1(k)^\top \ t_2(k)^\top \ \dots \ t_n(k)^\top]^\top \in \mathbb{R}^n$. Recall, $\mathbf{t}(0) = \mathbf{1}$. From (3.28) and (3.29) we obtain for every $k \geq 1$ and any agent $i \in \mathcal{V}$,

$$r_i(k+1) = \frac{\phi_{ki} \mathbf{1}^\top \mathbf{s}(k) + [(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{s}(0)]_i}{n\phi_{ki} + [(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{1}]_i}.$$

Therefore,

$$r_i(k+1) - \frac{\mathbf{1}^\top \mathbf{s}(k)}{n} = \frac{\phi_{ki} \mathbf{1}^\top \mathbf{s}(k) + [(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{s}(0)]_i}{n\phi_{ki} + [(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{1}]_i} - \frac{\mathbf{1}^\top \mathbf{s}(k)}{n}.$$

By rearranging the fraction, we get

$$r_i(k+1) - \frac{\mathbf{1}^\top \mathbf{s}(k)}{n} = \frac{n[(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{s}(0)]_i - \mathbf{1}^\top \mathbf{s}(k)[(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{1}]_i}{n(n\phi_{ki} + [(\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top) \mathbf{1}]_i)}.$$

Let $\delta := \min_k \min_{1 \leq i \leq n} \sum_j [\mathbf{P}^k]_{ij}$. Note, that there exists $\underline{\delta} > 0$ (see [119], [120]) such that $\delta > \underline{\delta}$. Therefore, for all i and $k \geq 0$, we have,

$$\left\| r_i(k+1) - \frac{\mathbf{1}^\top \mathbf{s}(k)}{n} \right\| \leq \frac{\sqrt{n} \max_j (|[\mathbf{P}^{k+1} - \phi_k \mathbf{1}^\top]_{ij}|) (\|\mathbf{s}(0)\|_1 + |\mathbf{1}^\top \mathbf{s}(k)|)}{\delta}. \tag{3.30}$$

Using, (3.27) and (3.30), with $C = 2\sqrt{n}C_1$ we get,

$$\left\| r_i(k) - \frac{\mathbf{1}^\top \mathbf{s}(k-1)}{n} \right\| \leq \frac{C \|\mathbf{s}(0)\|_1}{\delta} \lambda^k.$$

Note, that due to Assumption 9, $\frac{\mathbf{1}^\top \mathbf{s}(k-1)}{n} = \frac{\mathbf{1}^\top \mathbf{s}(0)}{n}$. Therefore, for $k \geq \left\lceil \frac{\log(1/\varepsilon)}{-\log(\lambda)} \right\rceil +$

$$\left\lceil \frac{\log(C\|s(0)\|_1/\delta)}{-\log(\lambda)} \right\rceil,$$

$$\left\| r_i(k) - \frac{1}{n} \sum_{l=1}^n s_l(0) \right\| \leq \varepsilon, \quad \forall i \in \mathcal{V}.$$

□

The GradConsensus Algorithm:

Here, each agent i maintains two estimates: an optimization variable $x_i(k) \in \mathbb{R}^p$ and a local update variable $z_i(k) \in \mathbb{R}^p$ at any iteration k . Every iteration k proceeds in two steps: first each agent i updates $z_i(k)$ via a gradient descent update based on its function f_i at $x_i(k-1)$. At the next step, the optimization variable $x_i(k)$ is updated to an estimate which is $\varepsilon(k)$ -close to the average value $\hat{z}(k) := \frac{1}{n} \sum_{i=1}^n z_i(k)$, i.e. $\|x_i(k) - \hat{z}(k)\| \leq \varepsilon(k)$, using the distributed ε -consensus protocol, initialized with $z_i(k)$ as the initial condition for the agent i and tolerance $\varepsilon(k)$. Thus, at each iteration k , every individual agent $i \in \mathcal{V}$ performs the following updates:

$$z_i(k) = x_i(k-1) - \alpha \nabla f_i(x_i(k-1)) \quad (3.31)$$

$$x_i(k) = r_i(\theta_k), \text{ with} \quad (3.32)$$

$$\|r_i(\theta_k) - \hat{z}(k)\| < \varepsilon(k), \quad \hat{z}(k) := \frac{1}{n} \sum_{i=1}^n z_i(k), \quad (3.33)$$

where, θ_k denotes the number of communication steps in the ε -consensus protocol at iteration k of the **GradConsensus** algorithm, and $r_i(\theta_k)$ is the approximate estimate of the average $\hat{z}(k)$ produced by the ε -consensus protocol. We make the following assumptions:

Assumption 10. For problem (3.9) the optimal objective function value f^* is finite and the optimal solution set $\mathcal{X}^*$ is non-empty.

Assumption 11. (i) For all $i \in \mathcal{V}$, f_i is a proper convex function.

(ii) Each f_i is Lipschitz differentiable with constant $L_{f_i} > 0$.

Proposition 3. Let assumptions 9-11 hold. Let $\alpha \leq 1/L_f$, where, $L_f := \sum_{i=1}^n L_{f_i}$. Assume, that the set of minimizing solutions of the function f_i is non-empty and bounded for all $i \in \mathcal{V}$. Then, there exists constant $\bar{R} < \infty$ such that $\|z_i(k)\| \leq \bar{R}$ for all k and $i = 1, \dots, n$.

Proof. Under Assumption 11, for any $x \in \mathbb{R}^p$ at any k ,

$$f_i(x) \leq f_i(x_i(k-1)) + \langle \nabla f_i(x_i(k-1)), x - x_i(k-1) \rangle + \frac{1}{2\alpha} \|x - x_i(k-1)\|^2$$

Substituting, $x = x_i(k-1) - \alpha \nabla f_i(x_i(k-1))$ we get,

$$f_i(z_i(k)) \leq f_i(x_i(k-1)), \text{ for all } k. \quad (3.34)$$

Without loss of generality assume, $x_i(0) = 0, f_i(0) < \infty$ and $\|\nabla f_i(0)\| \leq h_i < \infty$, for all $i \in \mathcal{V}$. Under Assumption 11,

$$f_i(x_i(k)) \leq f_i(0) + h_i \|x_i(k)\| + \frac{1}{2\alpha} \|x_i(k)\|^2, \forall k. \quad (3.35)$$

From (3.32) and (3.33), for all k ,

$$\|x_i(k)\| \leq \|\hat{z}(k)\| + \varepsilon(k). \quad (3.36)$$

Note that all the level sets of f_i are bounded since the set of minimizing solutions of f_i is non-empty and bounded (See [180], Proposition B.9). Next, we prove that the $\|z_i(k)\|$ is bounded for $i = 1, \dots, n$ and for all k . We use an induction argument. Assume, there exists, R_i^k such that $\|z_i(k)\| \leq R_i^k \implies \|\hat{z}(k)\| \leq R_i^k$. From, (3.36), $\|x_i(k)\| \leq R_i^k + \varepsilon(k) \leq R_i^k + \bar{\varepsilon}$, where, $\bar{\varepsilon} := \sup_k \varepsilon(k)$. From (3.35), $f_i(x_i(k)) \leq f_i(0) + h_i(R_i^k + \bar{\varepsilon}) + \frac{1}{2\alpha} \|R_i^k + \bar{\varepsilon}\|^2 < \infty$. Thus, from (3.34) and the fact that all the level sets of f_i are bounded $\|z_i(k+1)\| \leq R_i^{k+1}$, where, R_i^{k+1} is diameter of the $(f_i(0) + h_i(R_i^k + \bar{\varepsilon}) + \frac{1}{2\alpha} \|R_i^k + \bar{\varepsilon}\|^2)$ -level set of f_i . Note, that $f_i(x_i(0)) = f_i(0) < \infty \implies \|z_i(1)\| \leq R_i^1$. Thus, the induction hypothesis can be applied recursively. Hence, induction holds. Therefore,

$$\|z_i(k)\| \leq \bar{R}, \text{ for } i = 1, \dots, n, \text{ and for all } k,$$

where, $\bar{R} := \max_{1 \leq i \leq n, k} R_i^k < \infty$. □

Remark 15. Proposition 3 assumes that the functions f_i have bounded minimizing solutions. Note this is not a restriction as in most applications the gradient descent minimization step is performed over a large enough compact set. In that case, f_i being proper convex functions have bounded minimizing solutions. Often in practical applications, the solution that minimizes the objective function f_i in problem (3.9) is bounded.

Since, the **GradConsensus** algorithm utilizes the ε -consensus protocol at every iteration k with the initialization $s_i(0) = z_i(k)$, $t_i(0) = 1, \forall i \in \mathcal{V}$, the number of communication rounds per iteration can be determined using Lemma 12 and Proposition 3. In particular, the number of communication rounds at iteration k of the **GradConsensus** algorithm is given by,

$$\theta_k := \left\lceil \frac{\log(1/\varepsilon(k))}{-\log(\lambda)} + \frac{\log(nC\bar{R}/\delta)}{-\log(\lambda)} \right\rceil, \quad (3.37)$$

where, C, δ, λ and $\bar{R}$ are constants defined in Lemma 12 and Proposition 3 respectively. The **GradConsensus** algorithm is summarized in Algorithm 4.

Algorithm 4: GradConsensus Algorithm

Input:

| Consensus tolerances $\{\varepsilon(k)\}_{k \geq 0}$ and step-size α

Initialize:

| Each agent $j \in \mathcal{V}$ initializes its estimates as $x_j(0) = z_j(0) = 0$.

Repeat for $k = 1, 2, \dots$

for $j = 1, 2, 3, \dots, n$, (*In parallel*) **do**

 gradient descent iteration: $z_j(k) = x_j(k-1) - \alpha \nabla f_j(x_j(k-1))$

 consensus iterations: $x_j(k) \leftarrow \varepsilon(k)$ -consensus($z_j(k)$, $j \in \mathcal{V}$)

end

until a stopping criterion is met

Remark 16. *Note that **GradConsensus** algorithm (updates (3.17)-(3.19)) utilizes a column stochastic matrix which allows for a distributed synthesis of the protocol. This feature gives an advantage over existing schemes in the literature [50], [134], [175], [181] that utilize a doubly-stochastic weight matrix and cannot achieve tractable distributed synthesis in directed networks.*

Convergence Analysis:

We analyze the convergence of the **GradConsensus** algorithm under two scenarios: (i) when the aggregate function f is convex and Lipschitz differentiable and, (ii) when f is strongly convex and Lipschitz differentiable. We provide estimates of the rate of convergence of the iterates generated by Algorithm 4 to the optimal solution(s) of problem (3.9) and the communication steps performed by Algorithm 4 at each iteration. Let the average of the optimization variables at iteration k be denoted as: $\hat{x}(k) :=$

$\frac{1}{n} \sum_{i=1}^n x_i(k)$. Denote the gradient of the function f evaluated at individual optimization variables of all the agents and at the average $\hat{x}(k)$ as:

$$g(k) := \sum_{i=1}^n \nabla f_i(x_i(k)), \text{ and } \hat{g}(k) := \sum_{i=1}^n \nabla f_i(\hat{x}(k)) \quad (3.38)$$

respectively. A consequence of the ε -consensus protocol is that the difference between $g(k)$ and $\hat{g}(k)$ is bounded at every iteration k of the **GradConsensus** algorithm.

Lemma 13. *Under assumptions 9 and 11, at k^{th} iteration of Algorithm 4 with θ_k (defined in (3.37)) communication rounds per iteration of the consensus protocol,*

$$\|g(k) - \hat{g}(k)\| \leq 2nL_h\varepsilon(k),$$

where, $L_h = \max_{1 \leq i \leq n} L_{f_i}$ and $\varepsilon(k)$ is the consensus tolerance at iteration k .

Proof. Under Assumption 11, note that,

$$\begin{aligned} \|g(k) - \hat{g}(k)\| &= \left\| \sum_{i=1}^n \nabla f_i(x_i(k)) - \sum_{i=1}^n \nabla f_i(\hat{x}(k)) \right\| \\ &\leq L_h \sum_{i=1}^n \|x_i(k) - \hat{x}(k) + \hat{z}(k) - \hat{z}(k)\| \\ &\leq L_h \sum_{i=1}^n \|x_i(k) - \hat{z}(k)\| + \|\hat{z}(k) - \hat{x}(k)\| \leq 2nL_h\varepsilon(k), \end{aligned}$$

where, in the last step follows from (3.32) and (3.33). $\square$

From update (3.31),

$$\frac{1}{n} \sum_{i=1}^n z_i(k) = \frac{1}{n} \sum_{i=1}^n [x_i(k-1) - \alpha \nabla f_i(x_i(k-1))] = \hat{x}(k-1) - \frac{\alpha}{n} \sum_{i=1}^n \nabla f_i(x_i(k-1)).$$

Therefore, we have,

$$\begin{aligned} \hat{z}(k) &= \hat{x}(k-1) - \hat{\alpha}g(k-1) + \hat{x}(k) - \hat{x}(k) \\ \implies \hat{x}(k) &= \hat{x}(k-1) - \hat{\alpha}g(k-1) + v(k-1), \end{aligned} \quad (3.39)$$

with $v(k-1) := \hat{x}(k-1) - \hat{z}(k-1)$, $\|v(k-1)\| \leq \varepsilon(k-1)$ and $\hat{\alpha} = \frac{\alpha}{n}$. In the

centralized setting the information about the gradient of the function f , i.e. $\nabla f(x) = \sum_{i=1}^n \nabla f_i(x) = \hat{g}(x)$ is known to the central server. Here, an iteration of the (centralized) gradient descent is of the form: $\tilde{x} = x - \alpha \hat{g}(x)$, where, $\tilde{x}$ denotes the updated estimate of the optimal solution. Note that due to Lemma 13 the update (3.39) can be viewed as an inexact centralized gradient descent update performed at the average of all the agents' estimates for the function f . In particular,

$$\begin{aligned} \hat{x}(k+1) &= \hat{x}(k) - \hat{\alpha} \hat{g}(k) + u(k), \text{ with} \\ u(k) &:= \hat{\alpha}(\hat{g}(k) - g(k)) + v(k), \|u(k)\| \leq (2L_h \alpha + 1)\varepsilon(k). \end{aligned}$$

Therefore, the **GradConsensus** algorithm updates are equivalent to (approximate) centralized gradient descent updates applied to the average optimization variable $\hat{x}(k)$ at each iteration k . We define the solution residual $\hat{e}(k)$ and objective value residual $\hat{r}(k)$ at iteration k as:

For the optimal value $-\infty < f^* < \infty$ and the optimal solution set, $\mathcal{X}^*$

$$\hat{e}(k) := \hat{x}(k) - x^*, \text{ where, } x^* \in \mathcal{X}^*, \text{ and,} \quad (3.40)$$

$$\hat{r}(k) := f(\hat{x}(k)) - f^* = \sum_{i=1}^n f_i(\hat{x}(k)) - f^*. \quad (3.41)$$

Under Assumption 11 since all f_i are Lipschitz differentiable with parameter L_{f_i} , f also is Lipschitz differentiable with the constant $L_f = \sum_{i=1}^n L_{f_i}$. The following Lemma is a property of Lipschitz differentiable convex functions that is a standard result in convex analysis.

Lemma 14 ([182], Theorem 2.1.5). *Under Assumption 11, for all $x, y \in \mathbb{R}^p$,*

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \frac{1}{L_f} \|\nabla f(y) - \nabla f(x)\|^2.$$

The next result establishes that the **GradConsensus** algorithm converges to an optimal solution of the problem (3.9) whenever the function f is convex but not necessarily strongly convex and Lipschitz differentiable.

Theorem 16 (Convergence to an optimal solution). *Let $\{x_i(k)\}_{k \geq 1}$ denote the sequence of optimization variables generated by Algorithm 4. Let the sequence of tolerances $\varepsilon(k)$ satisfy: $\varepsilon(k) \in (0, 1), \forall k, \sum_{s=0}^{\infty} \varepsilon^2(s) < \infty$. Under assumptions 9-11, $x_i(k)$ for all $i \in \mathcal{V}$*

converges to a solution x^∞ of (3.9), i.e., $f(x^\infty) = f^*$ and $\nabla f(x^\infty) = 0$.

Proof. Under Assumption 11, at any iteration k ,

$$f(\hat{x}(k+1)) \leq f(\hat{x}(k)) + \langle \hat{g}(k), \hat{x}(k+1) - \hat{x}(k) \rangle + \frac{L_f}{2} \|\hat{x}(k+1) - \hat{x}(k)\|^2$$

Subtracting, f^* from both sides, and using (3.39),

$$\begin{aligned} \hat{r}(k+1) &\leq \hat{r}(k) + \langle \hat{g}(k), \hat{x}(k+1) - \hat{x}(k) \rangle + \frac{L_f}{2} \|\hat{x}(k+1) - \hat{x}(k)\|^2 \\ &= \hat{r}(k) - \hat{\alpha} \langle \hat{g}(k), g(k) \rangle + \frac{\hat{\alpha}^2 L_f}{2} \|g(k)\|^2 - \hat{\alpha} L_f \langle g(k), v(k) \rangle + \langle \hat{g}(k), v(k) \rangle + \frac{L_f}{2} \|v(k)\|^2 \\ &= \hat{r}(k) - \hat{\alpha} \langle \hat{g}(k), g(k) + \hat{g}(k) - \hat{g}(k) \rangle + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) + \hat{g}(k) - \hat{g}(k)\|^2 + \frac{L_f}{2} \|v(k)\|^2 \\ &\quad - \hat{\alpha} L_f \langle g(k) + \hat{g}(k) - \hat{g}(k), v(k) \rangle + \langle \hat{g}(k), v(k) \rangle \\ &= \hat{r}(k) - \hat{\alpha} \langle \hat{g}(k), \hat{g}(k) \rangle - \hat{\alpha} \langle \hat{g}(k), g(k) - \hat{g}(k) \rangle + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) - \hat{g}(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|\hat{g}(k)\|^2 \\ &\quad + \hat{\alpha}^2 L_f \langle g(k) - \hat{g}(k), \hat{g}(k) \rangle + (1 - \hat{\alpha} L_f) \langle \hat{g}(k), v(k) \rangle + \hat{\alpha} L_f \langle \hat{g}(k) - g(k), v(k) \rangle \\ &\quad + \frac{L_f}{2} \|v(k)\|^2 \\ &= \hat{r}(k) - \hat{\alpha} \langle \hat{g}(k), \hat{g}(k) \rangle + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) - \hat{g}(k)\|^2 + \frac{L_f}{2} \|v(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|\hat{g}(k)\|^2 \\ &\quad + (\hat{\alpha} - \hat{\alpha}^2 L_f) \langle \hat{g}(k), \hat{g}(k) - g(k) \rangle + (1 - \hat{\alpha} L_f) \langle \hat{g}(k), v(k) \rangle + \hat{\alpha} L_f \langle \hat{g}(k) - g(k), v(k) \rangle. \end{aligned}$$

Using the $\pm 2\langle a, b \rangle \leq \|a\|^2 + \|b\|^2$, $a, b \in \mathbb{R}^p$ inequality on the term $\langle \hat{g}(k), \hat{g}(k) - g(k) \rangle$ we get,

$$\begin{aligned} \hat{r}(k+1) &\leq \hat{r}(k) + \frac{L_f}{2} \|v(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) - \hat{g}(k)\|^2 + (1 - \hat{\alpha} L_f) \langle \hat{g}(k), v(k) \rangle \\ &\quad + \hat{\alpha} L_f \langle \hat{g}(k) - g(k), v(k) \rangle + \frac{(\hat{\alpha} - \hat{\alpha}^2 L_f)}{2} \|\hat{g}(k) - g(k)\|^2 + \frac{(\hat{\alpha} - \hat{\alpha}^2 L_f)}{2} \|\hat{g}(k)\|^2 \\ &\quad - \hat{\alpha} \|\hat{g}(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|\hat{g}(k)\|^2 \\ &= \hat{r}(k) + \frac{L_f}{2} \|v(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) - \hat{g}(k)\|^2 + \frac{(\hat{\alpha} - \hat{\alpha}^2 L_f)}{2} \|\hat{g}(k) - g(k)\|^2 \\ &\quad + (1 - \hat{\alpha} L_f) \langle \hat{g}(k), v(k) \rangle + \hat{\alpha} L_f \langle \hat{g}(k) - g(k), v(k) \rangle - \frac{\hat{\alpha}}{2} \|\hat{g}(k)\|^2. \end{aligned}$$

Application of the $\pm 2\langle a, b \rangle \leq \zeta \|a\|^2 + \frac{1}{\zeta} \|b\|^2$, $a, b \in \mathbb{R}^p$ with $\zeta = \frac{2}{\hat{\alpha}}$ on the term

$\langle \hat{g}(k), v(k) \rangle$ leads to

$$\begin{aligned}
\hat{r}(k+1) &\leq \hat{r}(k) + \frac{L_f}{2} \|v(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) - \hat{g}(k)\|^2 + \frac{(\hat{\alpha} - \hat{\alpha}^2 L_f)}{2} \|\hat{g}(k) - g(k)\|^2 \\
&\quad + \hat{\alpha} L_f \langle \hat{g}(k) - g(k), v(k) \rangle - \frac{\hat{\alpha}}{2} \|\hat{g}(k)\|^2 + (1 - \hat{\alpha} L_f) \left(\frac{\hat{\alpha}}{4}\right) \|\hat{g}(k)\|^2 \\
&\quad + (1 - \hat{\alpha} L_f) \frac{1}{\hat{\alpha}} \|v(k)\|^2 \\
&\leq \hat{r}(k) + \frac{L_f}{2} \|v(k)\|^2 + \frac{\hat{\alpha}^2 L_f}{2} \|g(k) - \hat{g}(k)\|^2 + \frac{(1 - \hat{\alpha} L_f)}{\hat{\alpha}} \|v(k)\|^2 \\
&\quad + \frac{(\hat{\alpha} - \hat{\alpha}^2 L_f)}{2} \|\hat{g}(k) - g(k)\|^2 + \hat{\alpha} L_f \langle \hat{g}(k) - g(k), v(k) \rangle - \frac{\hat{\alpha}}{4} \|\hat{g}(k)\|^2.
\end{aligned}$$

Further, using Cauchy-Schwarz inequality and dropping the negative terms,

$$\hat{r}(k+1) \leq \hat{r}(k) - \frac{\hat{\alpha}}{4} \|\hat{g}(k)\|^2 + \frac{1}{\hat{\alpha}} \|v(k)\|^2 + \frac{\hat{\alpha}}{2} \|g(k) - \hat{g}(k)\|^2 + \hat{\alpha} L_f \|\hat{g}(k) - g(k)\| \|v(k)\|$$

Using, Lemma 13, we get,

$$\hat{r}(k+1) \leq \hat{r}(k) - \frac{\hat{\alpha}}{4} \|\hat{g}(k)\|^2 + a_0 \varepsilon^2(k) \quad (3.42)$$

$$\leq \hat{r}(k) + a_0 \varepsilon^2(k). \quad (3.43)$$

where, $a_0 := (\frac{1}{\hat{\alpha}} + 2nL_h\hat{\alpha}(nL_h + L_f))$. Applying (3.43) recursively, we get,

$$\hat{r}(k) \leq \hat{r}(0) + a_0 \sum_{s=0}^{\infty} \varepsilon^2(s) < \infty. \quad (3.44)$$

Therefore, there exists sub-sequence $\hat{r}(k_l)$ such that $\lim_{l \rightarrow \infty} \hat{r}(k_l) < \infty$. From (3.42) it follows, $\lim_{l \rightarrow \infty} \hat{g}(k_l) = \lim_{l \rightarrow \infty} \nabla f(\hat{x}(k_l)) = 0$. From (3.39) and Lemma 13, $\|\hat{x}(k+1) - \hat{x}(k)\| \leq \hat{\alpha} \|\hat{g}(k)\| + (2nL_h\hat{\alpha} + 1)\varepsilon(k)$. Thus, taking the limit over appropriate sequences on both sides gives $\|\hat{x}(k_l+1) - \hat{x}(k_l)\| \rightarrow 0$. Thus, $\hat{x}(k_l) \rightarrow \hat{x}^\infty$. Note that for all $i \in \mathcal{V}$, $\|x_i(k) - \hat{x}(k)\| \leq \varepsilon(k)$. Therefore, there exists x^∞ such that $\lim_{l \rightarrow \infty} x_i(k_l) = x^\infty$ for all $i = 1, \dots, n$. Now, for any $i \in \mathcal{V}$, consider $\|\nabla f(\hat{x}(k)) - \nabla f(x_i(k))\| \leq L_f \|\hat{x}(k) - x_i(k)\|$. Taking, limits over appropriate sequences, $\|\nabla f(\hat{x}(k_l)) - \nabla f(x_i(k_l))\| \rightarrow 0$. Thus, $\nabla f(x^\infty) = \lim_{l \rightarrow \infty} \nabla f(x_i(k_l)) = 0$. Further, since, f is convex,

$$f(x^*) \geq f(x_i(k)) + \langle \nabla f(x_i(k)), x^* - x_i(k) \rangle.$$

Note that under the assumptions here $x_i(k)$ is bounded. Taking limit over appropriate sequences on both sides gives, $f^* = f(x^*) \geq f(x^\infty)$. Thus, $f(x^\infty) = f^*$. To complete the proof, we show that $x_i(k)$ has a unique limit point. Let x_1^∞ and x_2^∞ be any two limit points of $x_i(k)$. From (3.44) and the fact that f is convex we have, $\|x_i(k) - x_j^\infty\|$ is bounded for all $i \in \mathcal{V}, j = 1, 2$. Thus, $\lim_{k \rightarrow \infty} \|x_i(k) - x_j^\infty\|^2 = \psi_j < \infty$ exists for all $i \in \mathcal{V}, j = 1, 2$. Now consider,

$$\|x_i(k) - x_1^\infty\|^2 - \|x_i(k) - x_2^\infty\|^2 = \|x_1^\infty\|^2 - \|x_2^\infty\|^2 - 2(x_1^\infty - x_2^\infty)^\top x_i(k)$$

Taking limit both sides for each limit point we get,

$$\begin{aligned} \psi_1 - \psi_2 &= \|x_1^\infty\|^2 - \|x_2^\infty\|^2 - 2(x_1^\infty - x_2^\infty)^\top x_1^\infty = -\|x_1^\infty - x_2^\infty\|^2 \text{ and,} \\ \psi_1 - \psi_2 &= \|x_1^\infty\|^2 - \|x_2^\infty\|^2 - 2(x_1^\infty - x_2^\infty)^\top x_2^\infty = \|x_1^\infty - x_2^\infty\|^2. \end{aligned}$$

Therefore, $\|x_1^\infty - x_2^\infty\| = 0$. Hence, x^∞ is unique. $\square$

We now provide two estimates for the convergence rate for the **GradConsensus** algorithm.

A. Q-linear Rate of Convergence:

Here, the convergence rate estimate of the **GradConsensus** algorithm is derived when the function f is convex but not necessarily strongly convex and Lipschitz differentiable. Under these assumptions, Theorem 17 establishes a Q-linear rate of convergence to a small neighborhood of an optimal solution of (3.9).

Theorem 17 (Q-linear rate of convergence). *Let assumptions 9-11 hold. Let $\hat{\alpha} \leq \frac{2}{L_f}, \varepsilon(k) = \frac{\varepsilon_0}{k^{1+\eta}}$, where, ε_0, η are positive constants. Denote, the Riemann zeta function by $\zeta(\cdot)$ and let $\mathbf{e} := \|\hat{x}(0) - x^*\| + (2\alpha L_h + 1)\varepsilon_0\zeta(1 + \eta)$. Then, while $\hat{r}(k) > 2\mathbf{e}\varepsilon_0\sqrt{\frac{2}{\hat{\alpha}}(\frac{1}{\hat{\alpha}} + 2nL_h\hat{\alpha}(nL_h + L_f))}$, $\hat{r}(k)$ decreases at a Q-linear rate. In particular, there exists $\beta \in (0, 1)$, such that for all $k \geq 0$, $\hat{r}(k + 1) \leq \beta\hat{r}(k)$.*

Proof. We show that the solution residual $\|\hat{e}(k)\|$ is bounded. Consider,

$$\begin{aligned} \|\hat{x}(k) - \hat{\alpha}\hat{g}(k) - x^*\|^2 &= \|\hat{x}(k) - x^*\|^2 + \hat{\alpha}^2\|\hat{g}(k)\|^2 - 2\hat{\alpha}\langle\hat{g}(k), \hat{x}(k) - x^*\rangle \\ &\leq \|\hat{x}(k) - x^*\|^2 + \hat{\alpha}(\hat{\alpha} - \frac{2}{L_f})\|\hat{g}(k)\|^2. \end{aligned}$$

Therefore,

$$\|\hat{x}(k) - \hat{\alpha}\hat{g}(k) - x^*\| \leq \|\hat{x}(k) - x^*\|, \quad (3.45)$$

where, we used Lemma 14 and the facts $\hat{g}(x^*) = 0, \hat{\alpha} \leq 2/L_f$. Using (3.39), Lemma 13 and (3.45),

$$\begin{aligned} \|\hat{x}(k+1) - x^*\| &= \|\hat{x}(k) - \hat{\alpha}g(k) - x^* + v(k)\| \\ &= \|\hat{x}(k) - \hat{\alpha}g(k) + \hat{\alpha}\hat{g}(k) - \hat{\alpha}\hat{g}(k) - x^* + v(k)\| \\ &\leq \|\hat{x}(k) - \hat{\alpha}\hat{g}(k) - x^*\| + \hat{\alpha}\|g(k) - \hat{g}(k)\| + \varepsilon(k) \\ &\leq \|\hat{x}(k) - x^*\| + \frac{(2\alpha L_h + 1)\varepsilon_0}{k^{1+\eta}} \\ &\leq \|\hat{x}(0) - x^*\| + (2\alpha L_h + 1)\varepsilon_0 \sum_{s=0}^k \frac{1}{k^{1+\eta}} \\ &\leq [\|\hat{x}(0) - x^*\| + (2\alpha L_h + 1)\varepsilon_0 \zeta(1 + \eta)] =: \mathbf{e}, \end{aligned} \quad (3.46)$$

where, $\zeta(\cdot)$ is the Riemann zeta function. Note that, $\hat{r}(k) = f(\hat{x}(k)) - f^* \leq \langle \hat{g}(k), \hat{x}(k) - x^* \rangle = \langle \hat{g}(k), \hat{e}(k) \rangle$. Using (3.46) we obtain,

$$\|\hat{g}(k)\| \geq \|\hat{g}(k)\| \frac{\|\hat{x}(k) - x^*\|}{\mathbf{e}} \geq \frac{\langle \hat{g}(k), \hat{x}(k) - x^* \rangle}{\mathbf{e}} \geq \frac{\hat{r}(k)}{\mathbf{e}}.$$

Thus, from (3.42), we have,

$$\hat{r}(k+1) \leq \hat{r}(k) - \frac{\hat{\alpha}}{4\mathbf{e}^2} \hat{r}^2(k) + a_0 \varepsilon^2(k).$$

Therefore, while $\hat{r}(k) > 2\mathbf{e}\sqrt{\frac{2a_0}{\hat{\alpha}}}\varepsilon_0$, we have,

$$\begin{aligned} \hat{r}(k+1) &\leq \hat{r}(k) - \frac{\hat{\alpha}}{8\mathbf{e}^2} \hat{r}^2(k) \\ &\leq \hat{r}(k) - \frac{\hat{\alpha}}{8\mathbf{e}^2} \hat{r}(k) 2n\mathbf{e}\varepsilon_0 L_h = \left(1 - \frac{\alpha}{4\mathbf{e}} L_h \varepsilon_0\right) \hat{r}(k) = \beta \hat{r}(k). \end{aligned}$$

Note, $\alpha L_h \varepsilon_0 < 4\mathbf{e} \implies \beta \in (0, 1)$. □

Remark 17. From Theorem 16, there exists $\bar{B} < \infty$ such that $\|x_i(k)\| \leq \bar{B}$ for all $i \in \mathcal{V}$ and $k \geq 0$. Therefore, under Assumption 11, $\|\nabla f_i(x_i(k))\| \leq (L_h \bar{B} +$

$\max_{1 \leq i \leq n} \|\nabla f_i(x_i(0))\| =: h_m < \infty$ for all $i \in \mathcal{V}$. Thus,

$$\begin{aligned} f(x_i(k)) - f^* &\leq \hat{r}(k) + \langle \nabla f(x_i(k)), x_i(k) - \hat{x}(k) \rangle \\ &\leq \hat{r}(k) + \sum_{j=1}^n \|\nabla f(x_i(k))\| \|x_i(k) - \hat{x}(k)\| \\ &\leq \hat{r}(k) + h_m n \varepsilon(k) = \hat{r}(k) + h_m n \frac{\varepsilon_0}{k^{1+\eta}}. \end{aligned}$$

Hence, using the result of Theorem 17 we conclude that $f(x_i(k)) - f^*$, similar to $\hat{r}(k)$ decreases geometrically, with respect to the outer gradient descent iterations of the Algorithm 4, until reaching a small $O(\varepsilon_0)$ neighborhood. The parameter ε_0 is a user-specified algorithm parameter that can be chosen appropriately to get solutions close to the optimal solution.

Remark 18. For the result in Theorem 17, the communication rounds per iteration k for Algorithm 4 are given by $\left\lceil \frac{\log(k^{1+\eta}/\varepsilon_0)}{-\log(\lambda)} + \frac{\log(nC\bar{R}/\delta)}{-\log(\lambda)} \right\rceil = O(\log(k/(\varepsilon_0)^{\frac{1}{\eta+1}}))$, where C, δ, λ and $\bar{R}$ are constants defined in Lemma 12 and Proposition 3 respectively. The Algorithm 4 with $\varepsilon_0 = 1$ (in the worst case) utilizes $O(k \log k)$ number of communication steps along with k gradient descent iterations. This communication complexity is within a factor of $\log k$ of the optimal lower bound $O(k)$. The additional communication steps provide improved convergence guarantees for the **GradConsensus** algorithm over the existing algorithms in the literature. In particular, the Q -linear rate of convergence in Theorem 17 is stronger than the sub-linear convergence rate present in the literature [50], [166], [134] under Assumption 11. Moreover, empirically it is seen that the total number of gradient descent (computation) steps is smaller with our algorithm. Thus it is well suited to situations where computations are expensive.

B. R-linear Rate of Convergence:

In this subsection, we make an additional assumption:

Assumption 12. $f = \sum_{i=1}^n f_i$ is strongly convex with parameter $\sigma_f > 0$.

Note that Assumption 12 does not require all f_i 's to be necessarily strongly convex. However, when this is the case, Assumption 12 holds naturally. We show in Theorem 18 that under Assumption 12 the solution estimates of every agent converge exactly to the optimal solution of problem (3.9) at an R-linear rate. We utilize the following standard result in the proof of Theorem 18.

Lemma 15 ([182], Theorem 2.1.12). *Under assumptions 11 and 12 for all $x, y \in \mathbb{R}^p$,*

$$\langle x - y, \nabla f(x) - \nabla f(y) \rangle \geq \frac{\sigma_f L_f}{\sigma_f + L_f} \|x - y\|^2 + \frac{1}{\sigma_f + L_f} \|\nabla f(x) - \nabla f(y)\|^2.$$

Theorem 18. *Let assumptions 9-12 hold. Let $\hat{\alpha} \leq \frac{2}{(\sigma_f + L_f)}$. Define, $\rho := \sqrt{1 - \frac{2\hat{\alpha}\sigma_f L_f}{\sigma_f + L_f}}$. Let $\varepsilon(k) \leq \mu^k$, where, $\mu \in [\rho, 1)$. Then the agent estimates converge at an R -linear rate to the optimal solution,*

$$\|x_i(k) - x^*\| \leq \Gamma \mu^k, \text{ for all } i \in \mathcal{V},$$

where, $\Gamma := \|\hat{x}(0) - x^*\| + (2\alpha L_h + 1) \frac{\mu}{\mu - \rho} + 1$.

Proof. Consider,

$$\begin{aligned} \|\hat{x}(k) - \hat{\alpha}\hat{g}(k) - x^*\|^2 &= \|\hat{x}(k) - x^*\|^2 + \hat{\alpha}^2 \|\hat{g}(k)\|^2 - 2\hat{\alpha} \langle \hat{g}(k), \hat{x}(k) - x^* \rangle \\ &\leq \left(1 - \frac{2\hat{\alpha}\sigma_f L_f}{\sigma_f + L_f}\right) \|\hat{x}(k) - x^*\|^2 + \hat{\alpha} \left(\hat{\alpha} - \frac{2}{\sigma_f + L_f}\right) \|\hat{g}(k)\|^2 \\ &\leq \rho^2 \|\hat{x}(k) - x^*\|^2, \end{aligned} \tag{3.47}$$

where, we used Lemma 15 and the facts $\hat{g}(x^*) = 0, \hat{\alpha} \leq \frac{2}{(\sigma_f + L_f)}$. From (3.39),

$$\begin{aligned} \|\hat{x}(k+1) - x^*\| &= \|\hat{x}(k) - \hat{\alpha}g(k) - x^* + v(k)\| \\ &\leq \|\hat{x}(k) - \hat{\alpha}\hat{g}(k) - x^*\| + \hat{\alpha} \|\hat{g}(k) - g(k)\| + \varepsilon(k). \end{aligned} \tag{3.48}$$

Combining (3.47), (3.48) and the result of Lemma 13 we get,

$$\|\hat{x}(k+1) - x^*\| \leq \rho \|\hat{x}(k) - x^*\| + (2\alpha L_h + 1) \varepsilon(k) = \rho \|\hat{x}(k) - x^*\| + (2\alpha L_h + 1) \mu^k.$$

Applying the above inequality recursively we get,

$$\|\hat{x}(k) - x^*\| \leq \rho^k \|\hat{x}(0) - x^*\| + (2\alpha L_h + 1) \sum_{s=0}^{k-1} \rho^s \mu^{k-s}.$$

Using the triangle inequality and the above relation we get,

$$\begin{aligned}
\|x_i(k) - x^*\| &\leq \|\hat{x}(k) - x^*\| + \|x_i(k) - \hat{x}(k)\| \\
&\leq \rho^k \|\hat{x}(0) - x^*\| + (2\alpha L_h + 1) \sum_{s=0}^{k-1} \rho^s \mu^{k-s} + \mu^k \\
&\leq \rho^k \|\hat{x}(0) - x^*\| + (2\alpha L_h + 1) \mu^k \sum_{s=0}^{k-1} \left(\frac{\rho}{\mu}\right)^s + \mu^k \\
&\leq \rho^k \|\hat{x}(0) - x^*\| + \frac{(2\alpha L_h + 1)\mu}{\mu - \rho} \mu^k + \mu^k \\
&\leq \left[\|\hat{x}(0) - x^*\| + (2\alpha L_h + 1) \frac{\mu}{\mu - \rho} + 1 \right] \mu^k = \Gamma \mu^k.
\end{aligned}$$

□

Remark 19. For the result in Theorem 18, the communication rounds per iteration k for Algorithm 4 are given by $\left\lceil \frac{k \log(\mu)}{-\log(\lambda)} + \frac{\log(nC\bar{R}/\delta)}{-\log(\lambda)} \right\rceil = O(k)$, where C, δ, λ and $\bar{R}$ are constants defined in Lemma 12 and Proposition 3 respectively. Algorithm 4 (in the worst case) utilizes $O(k^2)$ number of communication steps along with k gradient descent iterations for a global R -linear rate of convergence to the exact optimal solution. The per node work complexity (sum of the total number of gradient computations and the total number of communication steps) to achieve an accuracy of $\tau > 0$, i.e., $\|x_i(k) - x^*\| \leq \tau$ is given by: $O(\log(1/\tau))$ computation steps + $O(\log(1/\tau)^2)$ communication steps.

From Theorems 17 and 18, the consensus parameter $\varepsilon(k)$ can be chosen suitably to get near-optimal solutions. Theorem 18 shows that controlling the disagreement between the agents (and hence the infeasibility of the algorithm iterates) to a small value using an appropriately chosen $\varepsilon(k)$, results in strong convergence guarantees (global R -linear rate of convergence) for the **GradConsensus** algorithm. However, there exists a trade-off as tight regulation of the disagreement between agents' estimates would require performing a larger number of total communication steps ($O(k^2)$ in the worst case) until any outer gradient descent iteration k , compared to $O(k \log k)$ (in the worst case) that gives a Q -linear rate (Theorem 17).

Numerical Simulations and Results:

We consider a network of 1000 agents with the interconnection topology generated using the Erdos-Rényi model [121] with connectivity probability 0.005. The interconnection

topology is shown in Figure 3.1. The diameter of the 1000 node graph is 10. In the simulation study, we choose the parameter $\mathcal{D}$ in the **GradConsensus** algorithm to be 11. The weight matrix $\mathbf{P}$ is chosen using the equal neighbor model [51]. To demonstrate

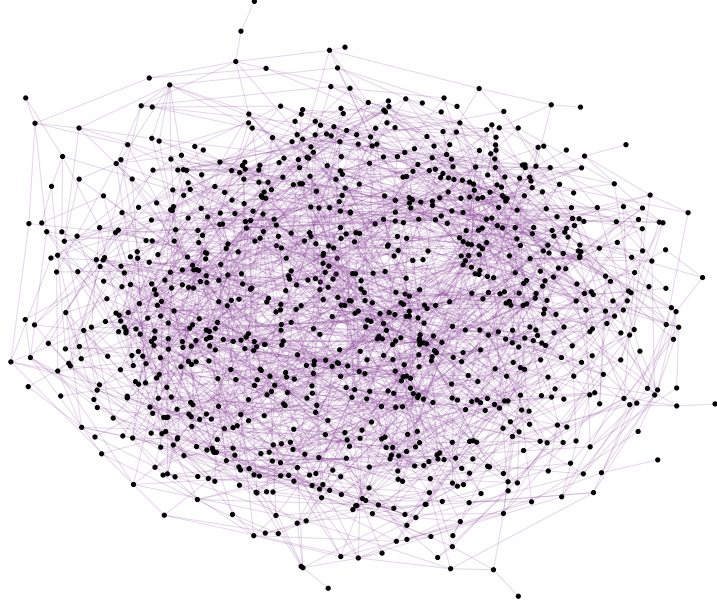


Figure 3.1: 1000-node test network topology

the performance, we solve the distributed logistic regression problem,

$$\underset{x \in \mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{1000} \frac{1}{n_i} \sum_{j=1}^{n_i} \log(1 + \exp(-(A_{ij}x)y_{ij})), \quad (3.49)$$

where, each agent $i \in \mathcal{V}$ has its training data $(A_{ij}, y_{ij}) \in \mathbb{R}^p \times \{-1, +1\}$, $j = 1, 2, \dots, n_i$, with feature variables A_{ij} and binary outcomes y_{ij} . Here, $(p = 35, n_i = 1500)$. For our simulations, we generate an artificial data set of feature vectors A_{ij} with outcome $y_{ij} = 1$ from a standard normal distribution, and with output $y_{ij} = -1$ from another normal distribution with mean 0 and standard deviation 0.707. We compare the performance of the proposed **GradConsensus** algorithm with the well-known Distributed Gradient Descent (DGD) [50] method, and the NEAR-DGD methods presented in [178] that utilize multiple communication steps at each iteration. Two versions of the **GradConsensus**, with $\varepsilon(k) = 0.01$, and $\varepsilon(k) = 1/k^{1.1}, \forall k$ are used for comparison. Two variants of the NEAR-DGD algorithm are implemented: (i) NEAR-DGD(1, 150, 2) where we start with

1 round of communication and the number of communication rounds are doubled after every 150 algorithm iterations, (ii) NEAR-DGD⁺ where the number of communication steps is increased linearly against algorithm iteration (at algorithm iteration k the number of communication steps performed is k).

Let $x_1(k), x_2(k), \dots, x_n(k)$ denote the algorithm estimates produced by the corresponding algorithm and f^* denote the optimal value of problem (3.49) solved via a centralized solver. We provide a performance comparison between the **GradConsensus** and the baseline methods (DGD and NEAR-DGD) by plotting the objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ against (i) the number gradient computations performed by the methods, (ii) total communication steps required by the methods, and (iii) the CPU run-time taken by the methods. We also provide a practical estimate of communication complexity by plotting the number of communication steps required at each algorithm iteration (denoted as the counter k in the above sections) for all the methods.

All the numerical examples in this section are implemented in MATLAB and run on a desktop computer with 16 GB RAM and an Intel Core i7 processor running at 1.90 GHz. The parameters used in the simulations for all the algorithms are reported in Table 3.1. In choosing the step-sizes of the algorithms in Table 3.1 we hand-optimized the hyper-parameters that produce a good performance for these algorithms while maintaining the stability of the algorithmic estimates for the logistic regression problem. We choose the same step size of 0.02 for all the compared algorithms.

Table 3.1: Algorithm Parameters of **GradConsensus**

Algorithm	Parameters
GradConsensus	$\varepsilon(k) = 0.01, 1/k^{1.1}$, step-size $\alpha = 0.02$
DGD [50]	step-size $\alpha = 0.02$
NEAR-DGD(1,150,2) [178]	step-size $\alpha = 0.02$
NEAR-DGD ⁺ [178]	step-size $\alpha = 0.02$

A. Comparison concerning gradient computations:

Figure 3.2 provides a comparison against the number of gradient computations. The **GradConsensus** algorithm with $\varepsilon(k) = 1/k^{1.1}$ gives the best performance among the compared algorithms converging to an objective value residual less than 10^{-3} after around 300 gradient computations. The NEAR-DGD⁺ method has the second-best performance, requiring more than 1100 iterations to achieve an objective value residual of 10^{-3} . Observe that **GradConsensus** with $\varepsilon(k) = 0.01$ performs better than the NEAR-DGD(1,150,2) method with objective value residual closer to 10^{-2} after 1100

gradient computations. The total number of gradient computation steps for the proposed **GradConsensus** algorithm with $\varepsilon(k) = 1/k^{1.1}$ is lesser than the other algorithms. This shows an advantage of using the **GradConsensus** algorithm in applications where the computation of the gradients is cost-intensive.

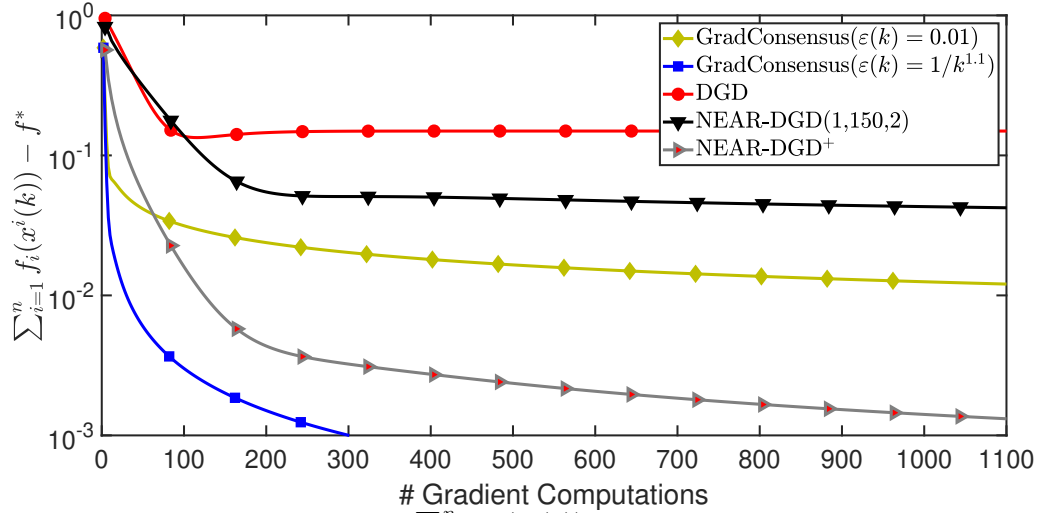


Figure 3.2: Objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ against the number of gradient computations.

B. Communication Complexity during algorithm iterations:

A comparison between the communication complexity of the variants of the proposed **GradConsensus** algorithm and the baselines (DGD and NEAR-DGD methods) is presented in Figure 3.3. The theoretical worst-case bounds on the communication steps for the **GradConsensus** algorithm is given in (3.37). The communication steps required at each iteration by the algorithms while solving the distributed logistic regression problem (3.107) are presented in Figure 3.3. The DGD method has the minimum number of communication steps required at each iteration with only one round of communication per algorithm iteration. The NEAR-DGD⁺ algorithm requires the number of communication steps to linearly increase with the number of iterations which leads to the worst total communication burden among the compared methods. The two variants of the **GradConsensus** algorithm and the NEAR-DGD(1,150,2) method incur a communication load that lies between the communication load of DGD and NEAR-DGD⁺ algorithms. The **GradConsensus** algorithm with $\varepsilon(k) = 0.01$ takes almost a constant number (50 to 52) of communication steps at each iteration throughout the simulation run. The **GradConsensus** algorithm with $\varepsilon(k) = 1/k^{1.1}$ has a near logarithmic increase in the number of communication steps taken at each iteration. The NEAR-DGD(1,150,2) method starts with one communication step load but has a geometric increase after every 150 algorithm iteration. The number of communication steps required by the NEAR-DGD(1,150,2) method per algorithm iteration becomes more than the two variants of the **GradConsensus** algorithm after 1050 algorithm iterations, implying worse communication complexity per iteration (than the **GradConsensus** algorithm variants) as the algorithm progresses.

C. Comparison in terms of communication steps:

Figure 3.4 compares the objective value residuals against the total communication steps taken by the algorithms. One can see that the **GradConsensus** algorithm starts slow due to the communication steps required by the ε -consensus protocol to maintain a feasible solution to the consensus constraint. Thus, for smaller total communication steps (less than 100) the **GradConsensus** algorithm doesn't have the best performance. In this region (total communication steps less than 100), the NEAR-DGD and DGD methods perform well. However, maintaining consensus constraint feasibility in the **GradConsensus** algorithm pays off as the algorithm progresses. For a higher total number of communication steps, the **GradConsensus** with $\varepsilon(k) = 1/k^{1.1}$ has the best

performance with the objective value residual value less than 10^{-2} after 900 communication steps. The NEAR-DGD⁺ method has a higher communication requirement (a linear increase in the communication steps compared to the logarithmic increase for the **GradConsensus** algorithm with $\varepsilon(k) = 1/k^{1.1}$) which deteriorates its eventual performance. NEAR-DGD(1,150,2) method on the other hand has a lesser communication requirement which leads to its initial good performance. However, the objective value residual plateaus and the **GradConsensus** algorithm with $\varepsilon(k) = 0.01$ (with consensus constraint feasibility) having started behind the NEAR-DGD(1,150,2) catches up with the objective value residual less than 10^{-1} around 1000 total communication steps. Thus, utilizing the ε -consensus protocol that uses an appropriate number of communication steps (while maintaining consensus constraint feasibility) based on the network structure rather than an ad-hoc increase in the communication steps leads to better performance of the **GradConsensus** algorithm.

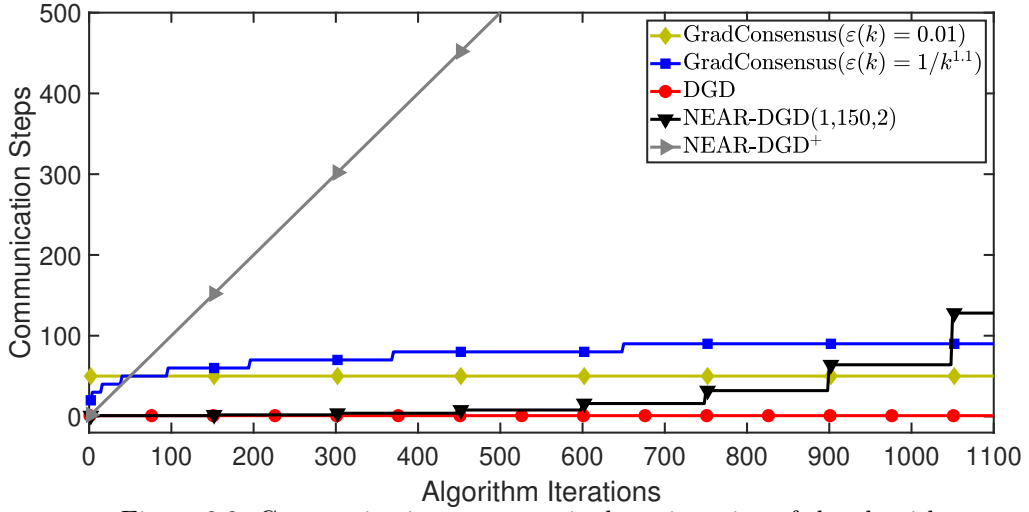


Figure 3.3: Communication steps required per iteration of the algorithm.

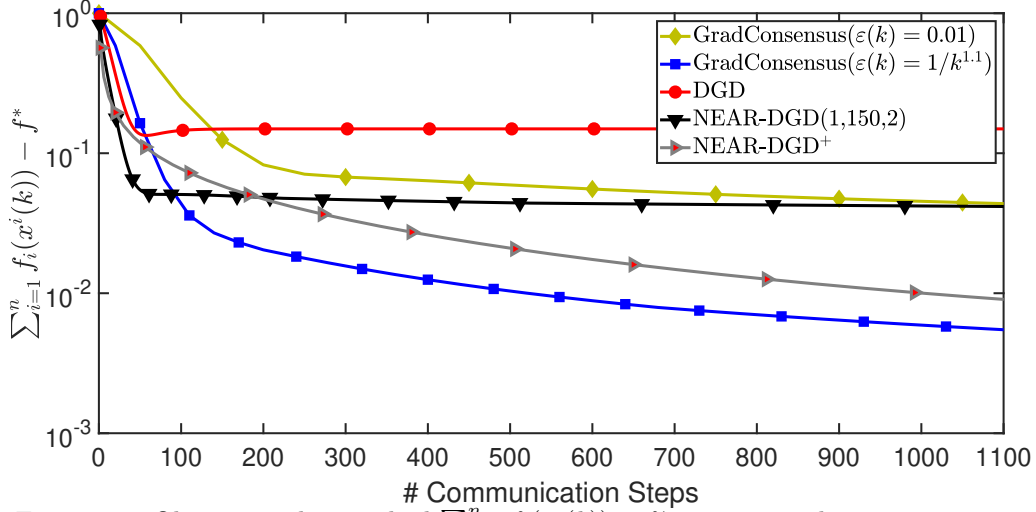


Figure 3.4: Objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ against total communication steps.

D. Comparison in terms of run time:

Next, we present a performance comparison between the proposed **GradConsensus** algorithm and other methods regarding the total run time required. For this simulation, during each iteration of the algorithms, a communication delay is introduced uniformly at random taking values between 0 and 1 ms (maximum delay of 1 ms). Figure 3.5 presents the objective value residual for the total run-time (in seconds). The ε -consensus protocol incurs additional communication steps, however, the information mixing steps (3.17)-(3.18) and (3.23)-(3.24) have a low computational footprint and do not significantly increase the overall run time of the **GradConsensus** algorithm. The objective value residual plots in Figure 3.5 demonstrate this. The **GradConsensus** algorithm with $\varepsilon(k) = 1/k^{1.1}$ has the best decrease in the objective value residual with a value of 10^{-3} . The **GradConsensus** with $\varepsilon(k) = 0.01$ also has good performance.

*E. Discussion on scalability of the **GradConsensus**:*

We remark that the proposed **GradConsensus** algorithm is distributedly synthesized and implemented. Each agent i only needs to know its neighborhood information. This is pivotal since the algorithm doesn't need to be set up centrally, and thus when the number of nodes in the network increases agents only need to adjust their local weights in the algorithm. Further, the information mixing over networks depends on the diameter of the directed graph which is a measure of connectivity in a graph. **GradConsensus**

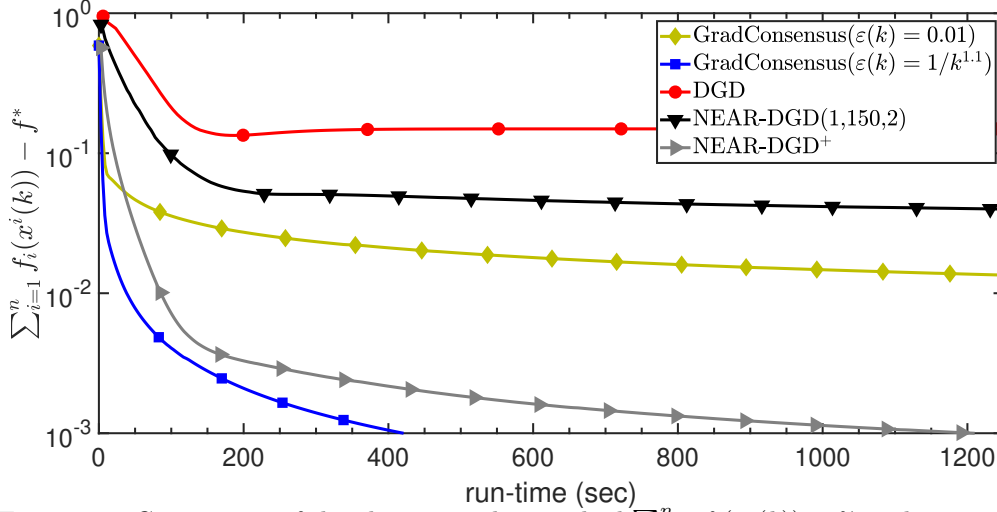


Figure 3.5: Comparison of the objective value residual $\sum_{i=1}^n f_i(x_i(k)) - f^*$ with respect to total run-time.

utilizes an upper bound $\mathcal{D}$ on the diameter. We emphasize that in many modern applications $\mathcal{D} \ll n$, the diameter of the graph is of a much smaller order than the number of nodes n in the graph [183]. Even though a thorough mathematical investigation of the relationship between a graph's size and its diameter is well outside the scope of the current thesis, as a heuristic, let us mention for example that in an Erdos-Rényi graph $G(n, p)$ (with n being the number of nodes and p being the probability of a link between two nodes), where p is an order larger than $\frac{\log n}{n}$, then the graph is connected almost surely, and its diameter is of the order $\log(n)/\log(np)$ (see [184]). In short, in large graphs, when connections between the nodes are random, the diameter of the graph is of a much smaller order than the size of the graph, even under sparse connectivity. Thus, the **GradConsensus** algorithm: (i) requiring only local neighborhood information by the agents for setup and implementation, and (ii) dependence on the graph diameter (or an upper bound $\mathcal{D}$) rather than the number of nodes, scales well with the increase in the number of nodes.

Summarizing, the simulation results indicate the **GradConsensus** the algorithm has better performance compared to the baselines for the number of gradient computations and the run time and remains acceptable for the total communication steps. Further, **GradConsensus** is scalable due to distributed synthesis and local computation and communication.

3.5.2 DC-DistADMM:ADMM Algorithm for Constrained Optimization over Directed Graphs

We start with a brief review of the standard ADMM method [141] utilized to solve optimization problems of the following form:

$$\begin{aligned} & \underset{x \in \mathbb{R}^p, y \in \mathbb{R}^q}{\text{minimize}} && f(x) + g(y) \\ & \text{subject to} && Sx + Ty = c, \end{aligned} \quad (3.50)$$

where, $S \in \mathbb{R}^{n \times p}$, $T \in \mathbb{R}^{n \times q}$, and $c \in \mathbb{R}^n$ is provided. Consider, the augmented Lagrangian with the Lagrange multiplier λ and positive scalar γ ,

$$\mathcal{L}_\gamma(x, y, \lambda) = f(x) + g(y) + \lambda^\top (Sx + Ty - c) + \frac{\gamma}{2} \|Sx + Ty - c\|^2. \quad (3.51)$$

The primal and dual variable updates in ADMM are given as: starting with the initial guess (x^0, y^0, λ^0) , at each iteration,

$$x^{k+1} = \underset{x}{\operatorname{argmin}} \mathcal{L}_\gamma(x, y^k, \lambda^k) \quad (3.52)$$

$$y^{k+1} = \underset{y}{\operatorname{argmin}} \mathcal{L}_\gamma(x^{k+1}, y, \lambda^k) \quad (3.53)$$

$$\lambda^{k+1} = \lambda^k + \gamma(Sx^{k+1} + Ty^{k+1} - c). \quad (3.54)$$

Note that step-size in the dual variable update (3.54) is the same as the parameter γ in the augmented Lagrangian function.

The Proposed DC-DistADMM Method:

Define $\mathbf{F}(\mathbf{x}) := \sum_{i=1}^n f_i(x_i)$, $\mathbf{A} := \mathbf{I} \otimes A_i, i = 1, \dots, n, \in \mathbb{R}^{nm \times n(m+p)}$, where $\mathbf{I}$ is $n \times n$ identity matrix and $\otimes$ denote the matrix Kronecker product, and $\mathbf{b} := [b_1^\top \dots b_n^\top]^\top \in \mathbb{R}^{nm}$. The Lagrangian function $\mathcal{L}$ for problem (3.8) is given by:

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda, \mu) = \mathbf{F}(\mathbf{x}) + \mathcal{I}_{C_0}(\mathbf{y}) + \lambda^\top (\mathbf{x} - \mathbf{y}) + \mu^\top (\mathbf{Ax} - \mathbf{b}). \quad (3.55)$$

The associated standard augmented Lagrangian at any iteration k is given by:

$$\mathcal{L}_\gamma(\mathbf{x}^k, \mathbf{y}^k, \lambda^k, \mu^k) = \mathcal{L}(\mathbf{x}^k, \mathbf{y}^k, \lambda^k, \mu^k) + \frac{\gamma}{2} \|\mathbf{x}^k - \mathbf{y}^k\|^2 + \frac{\gamma}{2} \|\mathbf{Ax}^k - \mathbf{b}\|^2. \quad (3.56)$$

Based on ADMM iterations (3.52)-(3.54) the primal and dual updates corresponding to augmented Lagrangian (3.56) are:

$$\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \mathbf{F}(\mathbf{x}) + \lambda^{k\top} (\mathbf{x} - \mathbf{y}^k) + \mu^{k\top} (\mathbf{A}\mathbf{x} - \mathbf{b}) + \frac{\gamma}{2} \|\mathbf{x} - \mathbf{y}^k\|^2 + \frac{\gamma}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right\}. \quad (3.57)$$

Note that the above $\mathbf{x}^{k+1}$ update can be obtained in a decentralized manner by computation at each agent as follows: $\forall i \in \mathcal{V}$,

$$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) + \lambda_i^{k\top} (x_i - y_i^k) + \mu_i^{k\top} (A_i x_i - b_i) + \frac{\gamma}{2} \|x_i - y_i^k\|^2 + \frac{\gamma}{2} \|A_i x_i - b_i\|^2 \right\}. \quad (3.58)$$

The update for the primal variable $\mathbf{y}$ is given as:

$$\begin{aligned} \bar{\mathbf{y}}^{k+1} &= \underset{\mathbf{y}}{\operatorname{argmin}} \left\{ \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}) + \lambda^{k\top} (\mathbf{x}^{k+1} - \mathbf{y}) + \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{y}\|^2 \right\} \\ &= \underset{\mathbf{y}}{\operatorname{argmin}} \left\{ \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}) + \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{y} + \frac{1}{\gamma} \lambda^k\|^2 \right\}, \end{aligned} \quad (3.59)$$

Using definition of the set $\mathcal{C}_0$ update (3.59) can be written as:

$$\begin{aligned} \bar{\mathbf{y}}^{k+1} &= \underset{\mathbf{y}}{\operatorname{argmin}} \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{y} + \frac{1}{\gamma} \lambda^k\|^2 \\ &\text{subject to } y_i = y_j, \forall i, j \in \mathcal{V}. \end{aligned}$$

Let $y_i = y_j = u, \forall i, j \in \mathcal{V}$. As the objective is in a block separable form we have the following equivalent problem,

$$\underset{u}{\operatorname{minimize}} \frac{\gamma}{2} \sum_{i=1}^n \|x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k - u\|^2.$$

Differentiating the objective function and setting it equal to zero gives the optimal solution. In particular, $\sum_{i=1}^n [x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k - u] = 0$ which implies $u = \frac{1}{n} \sum_{i=1}^n [x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k]$.

Therefore, $y_i = y_j = u, \forall i, j \in \mathcal{V}$. Hence,

$$\begin{aligned} \bar{\mathbf{y}}^{k+1} &= [\bar{y}^{k+1} \dots \bar{y}^{k+1}]^\top \in \mathbb{R}^{n(m+p)}, \text{ where,} \\ \bar{y}^{k+1} &= \frac{1}{n} \sum_{i=1}^n [x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k], \end{aligned} \quad (3.60)$$

is the solution to (3.59). However, obtaining the optimal solution (the exact average of the local variables) in a distributive manner over a directed network where each agent only has access to its own local information is a challenge. Hence, the exact consensus constraint is relaxed to a requirement of a η_{k+1} -closeness among the variables of all the agents, i.e., the solution $\mathbf{y}^{k+1}$ is allowed to lie in the set $\mathcal{C}_{\eta_{k+1}}$ and satisfy $\|y_i^{k+1} - y_j^{k+1}\| \leq 2\eta_{k+1}, \forall i, j \in \mathcal{V}$. The parameter η_{k+1} can be interpreted as a specified tolerance on the quality of consensus among the agent variables and can be chosen appropriately to find a corresponding near-optimal solution of (3.59). A distributed finite-time terminated approximate consensus protocol is employed to find an *inexact* solution of (3.59), such that the obtained solution $\mathbf{y}^{k+1} \in \mathcal{C}_{k+1}$. The “approximate” consensus protocol is discussed in detail in the later sections. After the primal variables, each agent updates the dual variables in the following manner:

$$\lambda_i^{k+1} = \lambda_i^k + \gamma(x_i^{k+1} - y_i^{k+1}), \forall i \in \mathcal{V}, \quad (3.61)$$

$$\mu_i^{k+1} = \mu_i^k + \gamma(A_i x_i^{k+1} - b_i), \forall i \in \mathcal{V}. \quad (3.62)$$

Define $\bar{\lambda}_i^{k+1}$ based on the solution $\bar{\mathbf{y}}^{k+1}$:

$$\bar{\lambda}_i^{k+1} = \lambda_i^k + \gamma(x_i^{k+1} - \bar{y}_i^{k+1}), \forall i \in \mathcal{V}. \quad (3.63)$$

Next, the ε -consensus protocol is described.

Finite-time ε -consensus protocol:

Here, a finite-time “approximate” consensus protocol, the ε -consensus protocol, is presented. Consider, a set of n agents connected via a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Every agent i has a vector $u_i^0 \in \mathbb{R}^p$. Let $\tilde{u} = \frac{1}{n} \sum_{i=1}^n u_i^0$ denote the average of the vectors. The objective is to design a distributed protocol such that the agents can compute an approximate estimate of $\tilde{u}$ in finite time. This approximate estimate is parametrized by a tolerance ε that can be chosen arbitrarily small to make the estimate as precise as

needed. To this end, the agents maintain state variables $u_i^k \in \mathbb{R}^p, v_i^k \in \mathbb{R}$ that undergo the following update: for $k \geq 0$,

$$u_i^{k+1} = p_{ii}u_i^k + \sum_{j \in \mathcal{N}_i^-} p_{ij}u_j^k \quad (3.64)$$

$$v_i^{k+1} = p_{ii}v_i^k + \sum_{j \in \mathcal{N}_i^-} p_{ij}v_j^k \quad (3.65)$$

$$w_i^{k+1} = \frac{1}{v_i^{k+1}} u_i^{k+1}, \quad (3.66)$$

where, $w_i^0 = u_i^0, v_i^0 = 1$ for all $i \in \mathcal{V}$ and $\mathcal{N}_i^-$ denotes the set of in-neighbors of agent i . The updates (3.64)-(3.66) are based on the push-sum (or ratio consensus) updates (see [42] or [185]). The following assumption on the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and weight matrix $\mathbf{P}$ is made:

Assumption 13. *The directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is strongly-connected. Let the weighted adjacency matrix $\mathbf{P} = [p_{ij}]$, associated with the digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, be column-stochastic.*

Note that $\mathbf{P}$ being a column stochastic matrix allows for a distributed synthesis of the consensus protocol. The variable $w_i^k \in \mathbb{R}^p$ is an estimate of $\tilde{u}$ with each agent i at any iteration k . It is established in prior work that the vector estimates w_i^k converge to the average $\tilde{u}$ asymptotically.

Theorem 19. *Let Assumption 13 hold. Let $\{w_i^k\}_{k \geq 0}$ be the sequence generated by (3.66) at each agent $i \in \mathcal{V}$. Then w_i^k asymptotically converges to $\tilde{u} = \frac{1}{n} \sum_{i=1}^n u_i^0$ for all $i \in \mathcal{V}$,*

$$\lim_{k \rightarrow \infty} w_i^k = \frac{1}{n} \sum_{i=1}^n u_i^0, \text{ for all } i \in \mathcal{V}.$$

Proof. Refer [47], [42] for the proof. □

The ε -consensus protocol is a distributed algorithm to determine when the agent states w_i^k , for all $i \in \mathcal{V}$ are ε -close to each other and hence from Theorem 19, ε -close to $\tilde{u}$. To this end, each agent maintains a scalar value R_i termed as the radius of agent i . The motivation behind maintaining such a radius variable is as follows: consider an open ball at iteration k that encloses all the agent states, w_i^k , for all $i \in \mathcal{V}$, with a minimal radius (the existence of such a ball can be shown due to bounded nature of the updates (3.64)-(3.66), see Lemma 4.2 in [122]). Theorem 19 guarantees that all

the agent states w_i^k converge to a single vector, which implies that the minimal ball enclosing all the states will also shrink with iteration k and eventually the radius will become zero. The radius variable R_i is designed to track the radius of this minimal ball. Starting at an iteration s , the radius $R_i^k(s)$ is updated to $R_i^{k+1}(s)$ as follows: for all $k = 0, 1, 2, \dots$, with $R_i^0(s) := 0$,

$$R_i^{k+1}(s) = \max_{j \in \mathcal{N}_i^-} \{ \|w_i^{s+k+1} - w_j^{s+k}\| + R_j^k(s) \}, \quad (3.67)$$

for all $i \in \mathcal{V}$. Denote, $\mathcal{B}(w_i^{s+k}, R_i^k(s))$ as the ball of radius $R_i^k(s)$ centered at w_i^{s+k} . It is established in [47] that, after $\mathcal{D}$ number of iterations of the update (3.67) the ball $\mathcal{B}(w_i^{s+\mathcal{D}}, R_i^{\mathcal{D}}(s))$ encloses the states w_i^s of all the agents $i \in \mathcal{V}$.

Lemma 16. *Let $\{w_i^k\}_{k \geq 0}$ be the sequence generated by (3.66) at each agent $i \in \mathcal{V}$. Given, $s \geq 0$, let $R_i^k(s)$ be updated as in (3.67) for all $k \geq 0$ and $i \in \mathcal{V}$. Under Assumption 13,*

$$w_j^s \in \mathcal{B}(w_i^{\mathcal{D}+s}, R_i^{\mathcal{D}}(s)), \text{ for all } j, i \in \mathcal{V}.$$

Proof. Define, $d_{i,j}$ to be the length of the shortest path connecting node i to node j . We use an induction argument on $d_{i,j}$. Base case: $d_{i,j} = 1$, Given, $s \geq 0$ for all $i \in \mathcal{V}$,

$$R_i^1(s) = \max_{j \in \mathcal{N}_i^-} \{ \|w_i^{s+1} - w_j^s\| + R_j^0(s) \}.$$

Therefore, for all $j \in \mathcal{N}_i^-$, $w_j^s \in \mathcal{B}(w_i^{s+1}, R_i^1(s))$. Assume the claim holds for $d_{i,j} = k$, that is for all j such that $d_{i,j} \leq k$, $w_j^s \in \mathcal{B}(w_i^{s+k}, R_i^k(s))$. For any node $\ell \in \mathcal{N}_i^-$, on the path between node i and j with $d_{i,j} \leq k+1$, we have $d_{\ell,j} \leq k$. By the induction hypothesis, $w_j^s \in \mathcal{B}(w_\ell^{s+k}, R_\ell^k(s))$. It implies, $\|w_\ell^{s+k} - w_j^s\| \leq R_\ell^k(s)$. From (3.67),

$$R_i^{k+1}(s) \geq \|w_i^{s+k+1} - w_\ell^{s+k}\| + R_\ell^k(s).$$

Using, the triangle inequality,

$$\begin{aligned} \|w_i^{s+k+1} - w_j^s\| &\leq \|w_i^{s+k+1} - w_\ell^{s+k}\| + \|w_\ell^{s+k} - w_j^s\| \\ &\leq \|w_i^{s+k+1} - w_\ell^{s+k}\| + R_\ell^k(s) \leq R_i^{k+1}(s). \end{aligned}$$

Therefore, for any j on the path with $d_{i,j} \leq k+1$, $w_j^s \in \mathcal{B}(w_i^{s+k+1}, R_i^{k+1}(s))$. Therefore,

induction holds. $k = \mathcal{D}$, gives the desired result. $\square$

Let

$$\hat{R}_i^m := R_i^{\mathcal{D}}(m\mathcal{D}), \quad m = 0, 1, 2, \dots, \quad (3.68)$$

where, $R_i^k(s)$ follows the update rule (3.67) for any $s \geq 0$. The next result establishes the fact that the sequence of radii $\{\hat{R}_i^{\mathcal{D}}(m\mathcal{D})\}_{m \geq 0}$, for $i \in \mathcal{V}$ converges to zero as $m \rightarrow \infty$.

Theorem 20. *Let $\{w_i^k\}_{k \geq 0}, \{\hat{R}_i^m\}_{m \geq 0}$ be the sequences generated by (3.66) and (3.68) respectively. Under Assumption 13,*

$$\lim_{m \rightarrow \infty} \hat{R}_i^m = 0, \quad \text{for all } i \in \mathcal{V}.$$

Further, $\lim_{m \rightarrow \infty} R_i^m = 0$ if and only if $\lim_{m \rightarrow \infty} \max_{i,j \in \mathcal{V}} \|w_i^{m\mathcal{D}} - w_j^{m\mathcal{D}}\| = 0$.

Proof. Define the element-wise maximum $\bar{\mathbf{w}}^k$ and minimum $\underline{\mathbf{w}}^k$ state variable of the network over all the agents as

$$\begin{aligned} \bar{\mathbf{w}}^k &:= \left[\max_{1 \leq j \leq n} \{w_{1[j]}^k\} \dots \max_{1 \leq j \leq n} \{w_{n[j]}^k\} \right]^\top, \\ \underline{\mathbf{w}}^k &:= \left[\min_{1 \leq j \leq n} \{w_{1[j]}^k\} \dots \min_{1 \leq j \leq n} \{w_{n[j]}^k\} \right]^\top, \end{aligned}$$

where, $w_{i[j]}^k$ is the j^{th} entry of w_i^k . It is proven in [114], [47] that $\lim_{m \rightarrow \infty} \bar{\mathbf{w}}^{m\mathcal{D}} = \lim_{m \rightarrow \infty} \underline{\mathbf{w}}^{m\mathcal{D}} = \tilde{u} = \frac{1}{n} \sum_{i=1}^n u_i^0$. It implies $\lim_{m \rightarrow \infty} \|\bar{\mathbf{w}}^{m\mathcal{D}} - \underline{\mathbf{w}}^{m\mathcal{D}}\| = 0$. Proceeding in the same manner as in [47], Theorem 4.2, we get $\hat{R}_i^m \leq \mathcal{D} \|\bar{\mathbf{w}}^{m\mathcal{D}} - \underline{\mathbf{w}}^{m\mathcal{D}}\|$. Taking $\lim_{m \rightarrow \infty}$ both sides we conclude for all $i \in \mathcal{V}$, $\lim_{m \rightarrow \infty} \hat{R}_i^m = 0$. This gives the first result. From Lemma 16, $\max_{i,j \in \mathcal{V}} \|w_i^{m\mathcal{D}} - w_j^{m\mathcal{D}}\| \leq 2\hat{R}_i^m$. Letting $m \rightarrow \infty$ gives the second claim. $\square$

Theorem 20 gives a criterion for termination of the consensus iterations (3.64)-(3.66) by utilizing the radius updates at each agent $i \in \mathcal{V}$ given by (3.67). In particular, by tracking the radii $\hat{R}_i^m, m = 0, 1, 2 \dots$ each agent can determine an estimate of the radius of the minimal ball enclosing all the agent states (Lemma 16) distributively. Further, by monitoring the value of the $\hat{R}_i^m, m = 0, 1, 2 \dots$ each agent can know the state of consensus among all the agents. A protocol to determine ε -consensus among the agents is given in Algorithm 5. Algorithm 5 is initialized with $w_i^0 = u_i^0, v_i^0 = 1$ and $R_i^0 = 0$ respectively for all $i \in \mathcal{V}$. Each agent follows rules (3.64)-(3.66) and (3.67)

Algorithm 5: Finite-time ε -consensus protocol at each node $i \in \mathcal{V}$ [47]

Input:
 | Pre-specified tolerance $\varepsilon > 0$;
 | Diameter upper bound $\mathcal{D}$;
Initialize:
 | $w_i^0 = u_i^0$; $v_i^0 = 1$; $R_i^0 = 0$;
 | $m := 1$
Repeat for $k = 0, 1, 2, \dots$
 | **/* consensus updates (3.64)-(3.66) */**
 | $u_i^{k+1} = p_{ii}u_i^k + \sum_{j \in \mathcal{N}_i^-} p_{ji}u_j^k$
 | $v_i^{k+1} = p_{ii}v_i^k + \sum_{j \in \mathcal{N}_i^-} p_{ji}v_j^k$
 | $w_i^{k+1} = \frac{1}{v_i^{k+1}}u_i^{k+1}$
 | **/* radius update (3.67) */**
 | $R_i^{k+1} = \max_{j \in \mathcal{N}_i^-} \left\{ \|w_i^{k+1} - w_j^k\| + R_j^k \right\}$
 | **if** $k = m\mathcal{D} - 1$ **then**
 | | $\hat{R}_i^{m-1} = R_i^{k+1}$;
 | | **if** $\hat{R}_i^{m-1} < \varepsilon$ **then**
 | | | **break** ; // ε -consensus achieved
 | | **else**
 | | | $R_i^{k+1} = 0$;
 | | | $m = m + 1$;
 | **end**

to update its state and radius variables respectively. R_i^k is reset at each iteration of the form $k = m\mathcal{D}, m = 1, 2, \dots$ to have a value equal to 0. The sequence of radii $\{\hat{R}_i^m\}_{m \geq 0}$ is determined by setting the value $\hat{R}_i^{m-1}$ equal to R_i^k for all iterations of the form $k = m\mathcal{D}, m = 1, 2, \dots$, i.e., $\hat{R}_i^{m-1} = R_i^{m\mathcal{D}}$ for all $m \geq 1$. From Lemma 16, the ball $\mathcal{B}(w_i^{m\mathcal{D}}, \hat{R}_i^{m-1})$ will contain all estimates $w_i^{(m-1)\mathcal{D}}$. This ball is the estimate of the minimal ball enclosing all agent states $w_i^{(m-1)\mathcal{D}}$. Therefore, as a method to detect ε -consensus, at every iteration of the form $m\mathcal{D}$, for $m = 1, 2, \dots$, $\hat{R}_i^{m-1}$ is compared to the parameter ε , if $\hat{R}_i^{m-1} < \varepsilon$ then all the agent states $w_i^{(m-1)\mathcal{D}}$ were ε -close to $\tilde{u}$ (from Theorem 19) and the iterations (3.64)-(3.66) are terminated. Proposition 4 establishes that the ε -consensus converges in finite number of iterations.

Proposition 4. *Under the Assumption 13, ε -consensus is achieved in finite number of iterations at each agent $i \in \mathcal{V}$.*

Proof. Note, $\hat{R}_i^m \rightarrow 0$ as $m \rightarrow \infty$. Thus, given $\varepsilon > 0, i \in \mathcal{V}$ there exists finite $\bar{k}_i^\varepsilon$ such that for $m \geq \bar{k}_i^\varepsilon, \hat{R}_i^m < \varepsilon$. □

Note, that the radius estimates $\widehat{R}_i^m$ can be different for some of the agents. Thus, the detection of ε -consensus can happen at different iterations for some nodes. To, have a global detection, each agent can generate a “converged flag” indicating its detection. Such a flag signal can then be combined utilizing distributed one-bit consensus updates (see [47]), thus allowing the agents to achieve global ε -consensus.

y variable updates in DC-DistADMM:

The finite-time ε -consensus protocol discussed above is utilized to determine an inexact solution to the update (3.59). At any iteration $k \geq 0$ of the **DC-DistADMM** algorithm, each agent $i \in \mathcal{V}$ runs an ε -consensus protocol with the tolerance $\varepsilon = \eta_{k+1}$ and the following initialization:

$$u_i^0 = x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k, \quad v_i^0 = 1, \quad \text{and} \quad w_i^0 = u_i^0. \quad (3.69)$$

From Proposition 4, there exists finite t_k such that $\forall i, j \in \mathcal{V}$,

$$\|w_i^{t_k} - w_j^{t_k}\| \leq \eta_{k+1}, \quad \text{and} \quad \|w_i^{t_k} - \bar{y}^{k+1}\| \leq \eta_{k+1}, \quad (3.70)$$

where, $\bar{y}^{k+1} = \frac{1}{n} \sum_{i=1}^n [x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k]$. The variable y_i^{k+1} of agent i is then set equal to the consensus variable $w_i^{t_k}$, i.e. $y_i^{k+1} = w_i^{t_k}$. Recall, the solution to (3.59) is $\bar{\mathbf{y}}^{k+1} = [u^\top \dots u^\top]^\top$, where, $u := \frac{1}{n} \sum_{i=1}^n [x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k]$. The ε -consensus algorithm at iteration k of the **DC-DistADMM** algorithm, with the initial condition $u_i^0 = x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k$, $v_i^0 = 1, \forall i \in \mathcal{V}$ and $\varepsilon = \eta_{k+1}$ yields a solution of the form $[u^\top u^\top \dots u^\top]^\top + e^{k+1}$, where $\|e^{k+1}\| \leq \sqrt{n} \eta_{k+1}$. Therefore, the obtained vector $\mathbf{y}^{k+1} = [y_1^{k+1}^\top, \dots, y_n^{k+1}^\top]^\top \in \mathbb{R}^{n(m+p)}$ is an *inexact* solution to the update (3.59). Further, $\mathbf{y}^{k+1}$ is $\sqrt{n} \eta_{k+1}$ -close to the exact solution $\bar{\mathbf{y}}^{k+1}$, i.e.,

$$\mathbf{y}^{k+1} = \bar{\mathbf{y}}^{k+1} + e^{k+1}, \quad \text{with} \quad \|e^{k+1}\| \leq \sqrt{n} \eta_{k+1}. \quad (3.71)$$

The **DC-DistADMM** algorithm is presented in Algorithm 6.

Algorithm 6: DC-DistADMM Algorithm

Input:

| choose $\gamma > 0, \{\eta_k\}_{k \geq 0}$

Initialize:

| $x_i^0 \in \mathbb{R}^{(m+p)}, y_i^0 \in \mathbb{R}^{(m+p)}, \lambda_i^0 = \mathbf{0}_{m+p} \in \mathbb{R}^{(m+p)}, \mu_i^0 \in \mathbb{R}^m, \forall i \in \mathcal{V}$

Repeat for $k = 0, 1, 2, \dots$

for $i = 1, 2, \dots, n$, (*In parallel*) **do**

$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}} \left\{ f_i(x_i) + \lambda_i^{k\top} (x_i - y_i^k) + \frac{\gamma}{2} \|x_i - y_i^k\|^2 + \mu_i^{k\top} (A_i x_i - b_i) + \frac{\gamma}{2} \|A_i x_i - b_i\|^2 \right\},$

$y_i^{k+1} \xleftarrow{\eta_{k+1} \text{ - consensus protocol}} (x_i^{k+1} + \frac{1}{\gamma} \lambda_i^k)$

$\lambda_i^{k+1} = \lambda_i^k + \gamma(x_i^{k+1} - y_i^{k+1})$

$\mu_i^{k+1} = \mu_i^k + \gamma(A_i x_i^{k+1} - b_i)$

end

until a stopping criterion is met

Convergence Results for DC-DistADMM:

The convergence analysis for the proposed **DC-DistADMM** algorithm is presented here. Let $\mathcal{X} := \mathcal{X}_1 \times \dots \times \mathcal{X}_n, \mathbb{R}_{\geq 0}^m := \underbrace{\mathbb{R}_{\geq 0}^m \times \dots \times \mathbb{R}_{\geq 0}^m}_{n \text{ times}}$ and $x^* \in \mathbb{R}^{(m+p)}$ denote an optimal solution of problem (3.1). The following assumptions hold:

Assumption 14. $\mathcal{X}_i$ s are a closed bounded convex set with diameter $\operatorname{diam}_{\|\cdot\|}(\mathcal{X}_i) = \mathcal{M}_i$.

Note that Assumption 14 can be relaxed if the set of optimal solutions of the original distributed problem (3.1) is bounded (see [180]); in this case, the existence of a bound on the local variables can be inferred from the bound on optimal solutions. Here this route is not taken, as in many multi-agent distributed optimization applications, the variables are required to remain within specified bounds. Assumption 14 is motivated by the constraint set requirement in many practical applications.

Assumption 15. The function $\tilde{f}_i, i \in \{1, \dots, n\}$, is a proper closed convex function, which is not necessarily differentiable.

Assumption 16. There exists a saddle point $(\mathbf{x}^*, \mathbf{y}^*, \lambda^*, \mu^*)$ for the Lagrangian function $\mathcal{L}$ defined in (3.55), i.e., for all $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{n(m+p)} \times \mathbb{R}^{n(m+p)}, (\lambda, \mu) \in \mathbb{R}^{n(m+p)} \times \mathbb{R}^{nm}$,

$$\mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda, \mu) \leq \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda^*, \mu^*) \leq \mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda^*, \mu^*).$$

Note that since (3.8) is a convex optimization problem, the existence of dual optimal solutions is guaranteed if a constraint qualification condition like Slater's CQ holds for problem (3.8) [3]. Note that the indicator function of a set is a convex but not differentiable function. For further analysis of the **DC-DistADMM** algorithm consider the following relation: for $x_1, x_2, x_3, x_4 \in \mathbb{R}^p$

$$(x_1 - x_2)^\top (x_3 - x_4) = \frac{(\|x_1 - x_4\|^2 - \|x_1 - x_3\|^2)}{2} + \frac{(\|x_2 - x_3\|^2 - \|x_2 - x_4\|^2)}{2} \quad (3.72)$$

Under Assumption 15, first-order optimality and primal feasibility conditions for (3.8) are,

$$-\lambda_i^* - A_i^\top \mu_i^* \in \partial f_i(x_i^*), \quad i = 1, 2, \dots, n, \quad (3.73)$$

$$\lambda^* \in \partial \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}^*), \quad (3.74)$$

$$x_i^* = y_i^*, \quad i = 1, 2, \dots, n, \quad (3.75)$$

$$A_i x_i^* - b_i = 0, \quad i = 1, 2, \dots, n, \quad (3.76)$$

where, $\partial f_i(x_i^*)$ and $\partial \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}^*)$ are the set of all the sub-gradients of f_i at x_i^* , and $\mathcal{I}_{\mathcal{C}_0}$ at $\mathbf{y}^*$ respectively. Similarly, for the **DC-DistADMM** updates (3.58), and (3.59) by the first-order optimality condition it follows that, for all $i \in \{1, \dots, n\}$:

$$-(\lambda_i^k + A_i^\top \mu_i^k + \gamma(x_i^{k+1} - y_i^k) + \gamma A_i^\top (A_i x_i^{k+1} - b_i)) \in \partial f_i(x_i^{k+1}), \quad (3.77)$$

$$-(\lambda_i^{k+1} + \gamma(y_i^{k+1} - y_i^k) + A_i^\top \mu_i^{k+1}) \in \partial f_i(x_i^{k+1}), \quad (3.78)$$

$$\begin{aligned} \gamma(\mathbf{x}^{k+1} - \bar{\mathbf{y}}^{k+1}) + \lambda^k &\in \partial \mathcal{I}_{\mathcal{C}_0}(\bar{\mathbf{y}}^{k+1}) \\ \implies \bar{\lambda}^{k+1} &\in \partial \mathcal{I}_{\mathcal{C}_0}(\bar{\mathbf{y}}^{k+1}), \end{aligned} \quad (3.79)$$

where, $\partial \mathcal{I}_{\mathcal{C}_0}(\bar{\mathbf{y}}^{k+1})$ is the set of all sub-gradients of $\mathcal{I}_{\mathcal{C}_0}$ at $\bar{\mathbf{y}}^{k+1}$. The relation (3.78) results from combining (3.77) with the update (3.61), and in (3.79) (3.71) is employed.

Lemma 17. *Let $\{\mathbf{x}^k\}_{k \geq 1}$, $\{\mathbf{y}^k\}_{k \geq 1}$, $\{\lambda^k\}_{k \geq 1}$, and $\{\mu^k\}_{k \geq 1}$, be the sequences generated by Algorithm 6. Let $\{\eta_k\}_{k \geq 1}$ be the sequence of tolerances in Algorithm 6. Under assumptions 13 and 14, there exists $\mathbf{Q}$, independent of k , such that,*

$$\|\lambda^k\| \leq k\mathbf{Q}, \quad \text{for all } k.$$

Proof. Note that from the initialization in Algorithm 6 $\lambda_i = 0, \forall i \in \mathcal{V}$. From (3.60),

(3.61), (3.63), and, (3.71), for all $i \in \mathcal{V}$,

$$\begin{aligned}
\lambda_i^{k+1} &= \lambda_i^k + \gamma(x_i^{k+1} - \bar{y}^{k+1}) - \gamma e_i^{k+1} \\
&= \gamma \sum_{s=0}^k (x_i^{s+1} - \bar{y}^{s+1}) - \gamma \sum_{s=0}^k e_i^{s+1} \\
&= \gamma \sum_{s=0}^k \left[x_i^{s+1} - \frac{1}{n} \sum_{j=1}^n (x_j^{s+1} + \frac{1}{\gamma} \lambda_j^s) \right] - \gamma \sum_{s=0}^k e_i^{s+1} \\
&= -\frac{1}{n} \sum_{s=0}^k \sum_{j=1}^n \lambda_j^s + \gamma \sum_{s=0}^k \left[x_i^{s+1} - \frac{1}{n} \sum_{j=1}^n x_j^{s+1} \right] - \gamma \sum_{s=0}^k e_i^{s+1}.
\end{aligned}$$

Let $l_i^{k+1} := \gamma \sum_{s=0}^k \left[x_i^{s+1} - \frac{1}{n} \sum_{j=1}^n x_j^{s+1} \right] - \gamma \sum_{s=0}^k e_i^{s+1}$. Therefore,

$$\lambda_i^{k+1} = -\frac{1}{n} \sum_{s=0}^k \sum_{j=1}^n \lambda_j^s + l_i^{k+1}$$

Similarly,

$$\lambda_i^k = -\frac{1}{n} \sum_{s=0}^{k-1} \sum_{j=1}^n \lambda_j^s + l_i^k$$

Taking the difference of the above two equations we get,

$$\lambda_i^{k+1} = \lambda_i^k - \frac{1}{n} \sum_{j=1}^n \lambda_j^k + \gamma \left[x_i^{k+1} - \frac{1}{n} \sum_{j=1}^n x_j^{k+1} \right] - \gamma e_i^{k+1}.$$

Let $r_i^{k+1} = \gamma \left[x_i^{k+1} - \frac{1}{n} \sum_{j=1}^n x_j^{k+1} \right] - \gamma e_i^{k+1}$. Taking z -transform both sides of the above equation we get,

$$z\hat{\lambda}_i = \hat{\lambda}_i - \frac{1}{n} \sum_{j=1}^n \hat{\lambda}_j + \hat{r}_i,$$

where, $(\hat{\cdot})$ denotes the z -transform of the corresponding time-domain signal. Let $\hat{\lambda} :=$

$[\hat{\lambda}_1^\top \ \hat{\lambda}_2^\top \ \dots \ \hat{\lambda}_n^\top]^\top$ and $\hat{r} := [\hat{r}_1^\top \ \hat{r}_2^\top \ \dots \ \hat{r}_n^\top]^\top$. Writing, the above n equations compactly,

$$\underbrace{\begin{bmatrix} (z-1+\frac{1}{n})\mathbf{I}_p & \frac{1}{n}\mathbf{I}_p & \cdots & \frac{1}{n}\mathbf{I}_p \\ \frac{1}{n}\mathbf{I}_p & (z-1+\frac{1}{n})\mathbf{I}_p & \cdots & \frac{1}{n}\mathbf{I}_p \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n}\mathbf{I}_p & \frac{1}{n}\mathbf{I}_p & \cdots & (z-1+\frac{1}{n})\mathbf{I}_p \end{bmatrix}}_{T(z)} \hat{\lambda} = \hat{r},$$

where $\mathbf{I}_p$ is the p -dimensional identity matrix. Note, that the matrix $T(z)$ has eigenvalues z with algebraic multiplicity p and $(z-1)$ with algebraic multiplicity $(n-1)p$. Therefore, there exists a (constant) matrix Q such that

$$\hat{\lambda} = T^{-1}\hat{r} = Q^{-1} \begin{bmatrix} \frac{1}{z}\mathbf{I}_p & \mathbf{0}_{p \times p} & \cdots & \mathbf{0}_{p \times p} \\ \mathbf{0}_{p \times p} & \frac{1}{z-1}\mathbf{I}_p & \cdots & \mathbf{0}_{p \times p} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{p \times p} & \frac{1}{n}\mathbf{I}_p & \cdots & \frac{1}{z-1}\mathbf{I}_p \end{bmatrix} Q\hat{r}.$$

Let $Q\hat{\lambda} =: \hat{q} = [\hat{q}_1^\top \ \hat{q}_2^\top \ \dots \ \hat{q}_n^\top]^\top$, and $Q\hat{r} =: \hat{t} = [\hat{t}_1^\top \ \hat{t}_2^\top \ \dots \ \hat{t}_n^\top]^\top$. Therefore,

$$\hat{q}_1 = \frac{1}{z}\hat{t}_1, \quad \hat{q}_i = \frac{1}{z-1}\hat{t}_i, \quad \text{for } i = 2, 3, \dots, n.$$

Taking inverse z -transform on both sides of the above equations we get,

$$q_1^{k+1} = t_1^k, \quad q_i^{k+1} = q_i^k + t_i^k, \quad \text{for } i = 2, 3, \dots, n,$$

$$\text{Therefore, } q_1^{k+1} = t_1^k, \quad q_i^{k+1} = q_i^0 + \sum_{s=0}^k t_i^s, \quad \text{for } i = 2, 3, \dots, n.$$

Let $Q_i \in \mathbb{R}^{np}$ denote the i^{th} row of Q . Note, that $\|t_i^s\| = \|Q_i r^s\| = \|Q_i \gamma[[x_1^{s+1} - \frac{1}{n} \sum_{j=1}^n x_j^{s+1} - e_1^{s+1}]^\top \dots [x_1^{s+1} - \frac{1}{n} \sum_{j=1}^n x_j^{s+1} - e_1^{s+1}]^\top]^\top\| \leq \gamma\sqrt{n}\|Q_i\|(\mathcal{M} + \eta_{s+1}) \leq \gamma\sqrt{n}\|Q_i\|(\mathcal{M} + \eta_0), \forall i \in \mathcal{V}$. Let, $\|Q\|_{\max} := \max_{1 \leq i \leq n} \|Q_i\|$.

Therefore, $\|\lambda^k\| = \|Q^{-1}q^k\| \leq kn\gamma\|Q^{-1}\|\|Q\|_{\max}(\mathcal{M} + \eta_0)$. Note, that here

$$\mathbf{Q} := n\gamma\|Q^{-1}\|\|Q\|_{\max}(\mathcal{M} + \eta_0) < \infty, \quad (3.80)$$

associated with, $\|\lambda^k\| \leq k\mathbf{Q}$. □

Lemma 18 (Iteration complexity of the $\mathbf{y}^{k+1}$ update (3.70)). *Under assumptions 13 and 14 at iteration k of Algorithm 6, after*

$$\overline{t}_k := \left\lceil \frac{\log\left(\frac{1}{\eta_{k+1}}\right)}{-\log \alpha} + \frac{\log\left(\frac{8((\mathcal{M} + \|x^0\|)n + k\sqrt{n}\mathbf{Q}/\gamma)}{\beta}\right)}{-\log \alpha} \right\rceil,$$

iterations of the consensus protocol (updates (3.64)-(3.66)) with the initial condition, $u_i^0 = x_i^{k+1} + \frac{1}{\gamma}\lambda_i^k$, $v_i^0 = 1$, for all $i \in \mathcal{V}$ and $\varepsilon = \eta_{k+1}$ it follows that:

$$\|y_i^{k+1} - \bar{y}_i^{k+1}\| \leq \eta_{k+1}, \quad \forall i \in \mathcal{V}.$$

Here, $\mathcal{M} := \max_{1 \leq i \leq n} \mathcal{M}_i$, η_{k+1} denotes the inaccuracy introduced in solving relaxed version of problem (3.59), $\mathbf{Q}$ is a finite constant as given in Lemma 17, and $\beta > 0$, $\alpha \in (0, 1)$ are parameters of the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ satisfying $\beta \geq \frac{1}{n^n}$, $\alpha \leq (1 - \frac{1}{n^n})$.

Proof. To begin, we will present a modification of an existing result (Lemma 1 [120]). The result in Lemma 1 [120], with the perturbation term being zero, reduces to a convergence result for the push-sum protocol. Using this property, we conclude that the updates (3.64)-(3.66) converges at a geometric rate to the average of the initial values. Note that the **DC-DistADMM** algorithm at every iteration k utilizes ε -consensus protocol (updates (3.64)-(3.66)) with the initial values $x_i^{k+1} + \frac{1}{\gamma}\lambda_i^k$. Therefore, we conclude,

$$\|y_i^{k+1} - \bar{y}_i^{k+1}\| \leq \frac{8\sqrt{n}\alpha^{t_k}}{\beta} \|\mathbf{z}^k\|, \quad \forall i \in \mathcal{V}, \quad (3.81)$$

where, $\mathbf{z}^k := \mathbf{x}^{k+1} + \frac{1}{\gamma}\lambda^k$ and, $\beta > 0, \alpha \in (0, 1)$ satisfy: $\beta \geq \frac{1}{n^n}, \alpha \leq (1 - \frac{1}{n^n})$. Here, the variables α and β are parameters of the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The parameter α measures the speed at which the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ diffuses the information among the agents over time.

Further, the parameter, β measures the imbalance of influences in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ [120]. Under Assumption 14 using the result in Lemma 17, $\|\mathbf{z}^k\| \leq \sqrt{n}(\mathcal{M} + \|x^0\|) + k\frac{\mathbf{Q}}{\gamma}$. If $\frac{\beta\eta_{k+1}}{8\sqrt{n}\alpha^{t_k}} = \|\mathbf{z}^k\| \leq \sqrt{n}(\mathcal{M} + \|x^0\|) + k\frac{\mathbf{Q}}{\gamma}$ it implies that, $\frac{\eta_{k+1}\beta}{8((\mathcal{M} + \|x^0\|)n + k\frac{\sqrt{n}\mathbf{Q}}{\gamma})} \leq \alpha^{t_k}$. Therefore, we have,

$$t_k \leq \frac{-1}{\log \alpha} \left[\log\left(\frac{1}{\eta_{k+1}}\right) + \log\left(\frac{8((\mathcal{M} + \|x^0\|)n + k\frac{\sqrt{n}\mathbf{Q}}{\gamma})}{\beta}\right) \right] := \overline{t}_k.$$

Therefore, at the k^{th} iteration of Algorithm 6, after $\overline{t_k}$ number of iterations of the consensus protocol we have, $\|y_i^{k+1} - \overline{y}_i^{k+1}\| \leq \eta_{k+1}$, for all $i \in \mathcal{V}$. $\square$

Let the sequence of accuracy tolerances $\{\eta_k\}_{k \geq 1}$ in Algorithm 6 be such that

$$\text{for all } k \geq 0, \eta_{k+1} \leq \eta_k, \sum_{k=1}^{\infty} k\eta_k < \infty. \quad (3.82)$$

Some examples of sequences that satisfy (3.82) include:

$$\text{i) } \eta_k = \frac{1}{k^{2+q}}, q > 0, \text{ ii) } \eta_k = \rho^k, \rho \in (0, 1), \text{ iii) } \eta_k = \frac{\rho^k}{k!}.$$

In the following, the global convergence of the proposed **DC-DistADMM** algorithm to a solution of problem (3.8) is established.

Theorem 21 (Convergence of iterates generated by DC-DistADMM algorithm to an optimal solution). *Let $\{\mathbf{x}^k\}_{k \geq 1}$, $\{\mathbf{y}^k\}_{k \geq 1}$, $\{\lambda^k\}_{k \geq 1}$, $\{\mu^k\}_{k \geq 1}$, be the sequences generated by Algorithm 6. Let the consensus tolerance sequence $\{\eta_k\}_{k \geq 1}$ satisfy (3.82). Under assumptions 13-16, $(\mathbf{x}^k, \mathbf{y}^k, \lambda^k, \mu^k)$ converges to a solution $(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty)$ of (3.8), i.e., $\mathbf{F}(\mathbf{x}^\infty) = \mathbf{F}(\mathbf{x}^*)$, $\mathbf{y}^\infty \in \mathcal{C}_0$, $\mathbf{x}^\infty = \mathbf{y}^\infty$, and $\mathbf{A}\mathbf{x}^\infty = \mathbf{b}$.*

Proof. From (3.73), (3.78),

$$\begin{aligned} 0 &\leq [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [d\mathbf{F}(\mathbf{x}^{k+1}) - d\mathbf{F}(\mathbf{x}^*)] \\ &= [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [-\lambda^{k+1} - \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k) - \mathbf{A}^\top \mu^{k+1} + \lambda^* + \mathbf{A}^\top \mu^*] \\ &= [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\lambda^* - \lambda^{k+1}] + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] - \gamma[\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k], \end{aligned}$$

$d\mathbf{F}(\mathbf{x}^{k+1})$ is a sub-gradient of $\mathbf{F}$ at $\mathbf{x}^{k+1}$. Since, $\mathcal{I}_{\mathcal{C}_0}$ is convex,

$$\mathcal{I}_{\mathcal{C}_0}(\mathbf{y}_1) \geq \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}_2) + d^\top(\mathbf{y}_2)(\mathbf{y}_1 - \mathbf{y}_2),$$

where, $d(\mathbf{y}_2)$ is any sub-gradient of $\mathcal{I}_{\mathcal{C}_0}$ at $\mathbf{y}_2$. Using (3.79) and $\mathbf{y}_1 = \mathbf{y}^*$ and $\mathbf{y}_2 = \overline{\mathbf{y}}^{k+1}$,

$$\overline{\lambda}^{k+1 \top} (\overline{\mathbf{y}}^{k+1} - \mathbf{y}^*) \geq 0. \quad (3.83)$$

Similarly, for $\mathbf{y}_1 = \overline{\mathbf{y}}^{k+1}$, $\mathbf{y}_2 = \mathbf{y}^*$ and using (3.74),

$$-\lambda^{* \top} (\overline{\mathbf{y}}^{k+1} - \mathbf{y}^*) \geq 0. \quad (3.84)$$

Therefore, using (3.83) and (3.84) gives,

$$\begin{aligned}
0 &\leq [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\lambda^* - \lambda^{k+1}] + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] - \gamma [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] \\
&\quad + [\bar{\lambda}^{k+1} - \lambda^*]^\top [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*] \\
&= [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\lambda^* - \lambda^{k+1}] + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] - \gamma [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] \\
&\quad + [\lambda^{k+1} - \lambda^*]^\top [\mathbf{y}^{k+1} - \mathbf{y}^*] + \gamma e^{k+1\top} [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*] - e^{k+1\top} [\lambda^{k+1} - \lambda^*] \\
&= [\lambda^* - \lambda^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{y}^{k+1}] + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] + \gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] \\
&\quad + \gamma e^{k+1\top} [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*] - e^{k+1\top} [\lambda^{k+1} - \lambda^*] \\
&= \frac{1}{\gamma} [\lambda^{k+1} - \lambda^*]^\top [\lambda^k - \lambda^{k+1}] + \frac{1}{\gamma} [\mu^* - \mu^{k+1}]^\top [\mu^{k+1} - \mu^k] + \gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] \\
&\quad + \gamma e^{k+1\top} [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*] - e^{k+1\top} [\lambda^{k+1} - \lambda^*],
\end{aligned}$$

where, the first equality used (3.71), and the last equality utilized (3.61) and (3.62). Using (3.72) for $\frac{1}{\gamma} [\lambda^{k+1} - \lambda^*]^\top [\lambda^k - \lambda^{k+1}]$, $\frac{1}{\gamma} [\mu^* - \mu^{k+1}]^\top [\mu^{k+1} - \mu^k]$ and $\gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k]$ yields,

$$\begin{aligned}
&\frac{\gamma}{2} \|\mathbf{y}^{k+1} - \mathbf{x}^*\|^2 + \frac{1}{2\gamma} \|\lambda^{k+1} - \lambda^*\|^2 + \frac{1}{2\gamma} \|\mu^{k+1} - \mu^*\|^2 \\
&\leq \frac{\gamma}{2} \|\mathbf{y}^k - \mathbf{x}^*\|^2 + \frac{1}{2\gamma} \|\lambda^k - \lambda^*\|^2 + \frac{1}{2\gamma} \|\mu^k - \mu^*\|^2 - \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{y}^k\|^2 - \frac{1}{2\gamma} \|\mu^{k+1} - \mu^k\|^2 \\
&\quad + \gamma e^{k+1\top} [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*] - e^{k+1\top} [\lambda^{k+1} - \lambda^*].
\end{aligned} \tag{3.85}$$

Let $R_1 := \gamma e^{k+1\top} [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*] = \gamma (\mathbf{y}^{k+1} - e^{k+1} - \mathbf{y}^*)^\top e^{k+1} = \gamma \mathbf{y}^{k+1\top} e^{k+1} - \gamma \|e^{k+1}\|^2 - \gamma \mathbf{y}^{*\top} e^{k+1}$. Therefore, $\|R_1\| \leq \gamma \|\mathbf{y}^{k+1}\| \|e^{k+1}\| + \gamma \|\mathbf{y}^*\| \|e^{k+1}\|$. Using, Lemma 17, $\|R_1\| \leq n\gamma(\mathcal{M} + \|x^*\|)\eta_{k+1} + 2\sqrt{n}\mathbf{Q}(k+1)\eta_{k+1} + \gamma\sqrt{n}\|\mathbf{y}^*\|\eta_{k+1}$. and $-e^{k+1\top} [\lambda^{k+1} - \lambda^*]$. Let $R_2 := -e^{k+1\top} [\lambda^{k+1} - \lambda^*] = -e^{k+1\top} \lambda^{k+1} + e^{k+1\top} \lambda^*$. Therefore, using, Lemma 17, $\|R_2\| \leq \|e^{k+1}\| \|\lambda^{k+1}\| + \|e^{k+1}\| \|\lambda^*\| \leq \sqrt{n}\mathbf{Q}(k+1)\eta_{k+1} + \|\lambda^*\|\eta_{k+1}$. Let $\mathbf{R} := n\gamma(\mathcal{M} + \|x^*\|) + \gamma\sqrt{n}\|\mathbf{y}^*\| + \|\lambda^*\|$. Therefore, using upper bounds on R_1, R_2 ,

$$a_{k+1} \leq a_k + \mathbf{R}\eta_{k+1} + 3\sqrt{n}\mathbf{Q}(k+1)\eta_{k+1} \tag{3.86}$$

$$\leq a_0 + \mathbf{R} \sum_{s=0}^{\infty} \eta_s + 3\sqrt{n}\mathbf{Q} \sum_{s=0}^{\infty} s\eta_s, \tag{3.87}$$

where, $a_k := \frac{\gamma}{2} \|\mathbf{y}^k - \mathbf{x}^*\|^2 + \frac{1}{2\gamma} \|\lambda^k - \lambda^*\|^2 + \frac{1}{2\gamma} \|\mu^k - \mu^*\|^2$. The condition (3.82)

implies that a_k is bounded. Therefore, by the boundedness of a_k (3.87) there exists sub-sequence a_{k_t} such that, $\lim_{t \rightarrow \infty} a_{k_t} < \infty$ exists. Therefore, using (3.85), $\|\mathbf{x}^{k_t+1} - \mathbf{y}^{k_t}\| \rightarrow 0, \|\mu^{k_t+1} - \mu_t^k\| \rightarrow 0$. Note that, due to (3.87) $\|\mathbf{y}^{k_t+1} - \mathbf{y}_{k_t}\| \rightarrow 0, \|\mathbf{x}^{k_t+1} - \mathbf{x}_{k_t}\| = \|\mathbf{x}^{k_t+1} - \mathbf{y}^{k_t+1} + \mathbf{y}^{k_t+1} - \mathbf{y}^{k_t} + \mathbf{y}^{k_t} - \mathbf{x}_{k_t}\| \leq \|\mathbf{x}^{k_t+1} - \mathbf{y}^{k_t+1}\| + \|\mathbf{y}^{k_t+1} - \mathbf{y}^{k_t}\| + \|\mathbf{y}^{k_t} - \mathbf{x}_{k_t}\| \rightarrow 0$. Further, $\|\lambda^{k_t+1} - \lambda^{k_t}\| = \gamma \|\mathbf{x}^{k_t+1} - \mathbf{y}^{k_t+1}\| \rightarrow 0$ and $\|\mathbf{x}^{k_t+1} - \bar{\mathbf{y}}^{k_t+1}\| \leq \|\mathbf{x}^{k_t+1} - \mathbf{y}^{k_t+1}\| + \|\mathbf{y}^{k_t+1} - \bar{\mathbf{y}}^{k_t+1}\| \leq \|\mathbf{x}^{k_t+1} - \mathbf{y}^{k_t+1}\| + \sqrt{n}\eta_{k+1} \rightarrow 0$. Therefore, there exists a sub-sequence $(\mathbf{x}^{k_t}, \mathbf{y}^{k_t}, \lambda^{k_t}, \mu^{k_t})$ that converges to a limit point $(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty)$. Next, we show that the limit point $(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty)$ satisfy the constraints in problem (3.8) and the objective function value $\mathbf{F}(\mathbf{x}^\infty) = \mathbf{F}(\mathbf{x}^*)$. To this end, we will utilize the following Lemma.

Lemma 19. *Let $\phi : \mathbb{R}^p \rightarrow \mathbb{R}$ be a convex function. Given, $\tilde{z} \in \mathbb{R}^p$ and a positive number $\gamma > 0$, if $\hat{z}$ is a proximal minimization point, i.e., $\hat{z} := \underset{z}{\operatorname{argmin}} \phi(z) + \frac{1}{2\gamma} \|z - \tilde{z}\|^2$, then we have, for any $z \in \mathbb{R}^p$*

$$2\gamma(\phi(\hat{z}) - \phi(z)) \leq \|\tilde{z} - z\|^2 - \|\hat{z} - z\|^2 - \|\hat{z} - \tilde{z}\|^2. \quad (3.88)$$

Proof. Let $\Phi(z) := \phi(z) + \frac{1}{2\gamma} \|z - \tilde{z}\|^2$. Since, ϕ is convex it follows that $\Phi(z)$ is strongly convex with modulus $\frac{1}{\gamma}$. By the definition of the proximal minimization step, we have $0 \in \partial\Phi(\hat{z})$. Therefore,

$$\Phi(z) \geq \Phi(\hat{z}) + \frac{\gamma}{2} \|z - \hat{z}\|^2 \implies 2\gamma(\phi(\hat{z}) - \phi(z)) \leq \|\tilde{z} - z\|^2 - \|\hat{z} - z\|^2 - \|\hat{z} - \tilde{z}\|^2.$$

□

Using Lemma 19 for the updates (3.58) and (3.59) with $\hat{z} = \mathbf{x}^{k+1}, \tilde{z} = \mathbf{y}^k, z = \mathbf{x}$ and $\hat{z} = \bar{\mathbf{y}}^{k+1}, \tilde{z} = \mathbf{x}^{k+1}, z = \mathbf{y}$ we obtain,

$$\begin{aligned} & \frac{2}{\gamma} (\mathcal{L}(\mathbf{x}^{k+1}, \mathbf{y}^k, \lambda^k, \mu^k) + \frac{\gamma}{2} \|\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}\|^2 - \mathcal{L}(\mathbf{x}, \mathbf{y}^k, \lambda^k, \mu^k) - \frac{\gamma}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2) \\ & \leq \|\mathbf{y}^k - \mathbf{x}\|^2 - \|\mathbf{x}^{k+1} - \mathbf{x}\|^2 - \|\mathbf{x}^{k+1} - \mathbf{y}^k\|^2, \text{ and,} \\ & \frac{2}{\gamma} (\mathcal{L}(\mathbf{x}^{k+1}, \bar{\mathbf{y}}^{k+1}, \lambda^k, \mu^k) - \mathcal{L}(\mathbf{x}^{k+1}, \mathbf{y}, \lambda^k, \mu^k)) \\ & \leq \|\mathbf{x}^{k+1} - \mathbf{y}\|^2 - \|\bar{\mathbf{y}}^{k+1} - \mathbf{y}\|^2 - \|\bar{\mathbf{y}}^{k+1} - \mathbf{x}^{k+1}\|^2. \end{aligned}$$

Adding the above two inequalities we obtain,

$$\begin{aligned} & \frac{2}{\gamma}(\mathcal{L}(\mathbf{x}^{k+1}, \bar{\mathbf{y}}^{k+1}, \lambda^k, \mu^k) + \frac{\gamma}{2}\|\mathbf{Ax}^{k+1} - \mathbf{b}\|^2 - \mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda^k, \mu^k) - \frac{\gamma}{2}\|\mathbf{Ax} - \mathbf{b}\|^2) \\ & \leq \|\mathbf{y}^k - \mathbf{x}\|^2 + \|\mathbf{x}^{k+1} - \mathbf{y}\|^2 - \|\mathbf{x}^{k+1} - \mathbf{x}\|^2 - \|\bar{\mathbf{y}}^{k+1} - \mathbf{y}\|^2 - \|\mathbf{x}^{k+1} - \mathbf{y}^k\|^2 - \|\bar{\mathbf{y}}^{k+1} - \mathbf{x}^{k+1}\|^2. \end{aligned}$$

Taking limit over the appropriate sub-sequences on both sides of the above inequality, we get,

$$\mathcal{L}(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty) \leq \mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda^\infty, \mu^\infty) + \frac{\gamma}{2}\|\mathbf{Ax} - \mathbf{b}\|^2. \quad (3.89)$$

Note that the updates (3.62) and (3.63) can be written as the following minimization updates: $\forall i \in \mathcal{V}$,

$$\mu_i^{k+1} = \underset{\mu_i}{\operatorname{argmin}}\{-\lambda_i^{k\top}(x_i^{k+1} - y_i^{k+1}) - \mu_i^\top(A_i x_i^{k+1} - b_i) + \frac{1}{2\gamma}\|\mu_i - \mu_i^k\|^2\}, \quad (3.90)$$

$$= \mu_i^k + \gamma(A_i x_i^{k+1} - b_i),$$

$$\bar{\lambda}_i^{k+1} = \underset{\lambda_i}{\operatorname{argmin}}\{-\lambda_i^\top(x_i^{k+1} - \bar{y}_i^{k+1}) + \frac{1}{2\gamma}\|\lambda_i - \lambda_i^k\|^2\}, \quad (3.91)$$

$$= \lambda_i^k + \gamma(x_i^{k+1} - \bar{y}_i^{k+1}).$$

Applying Lemma 19 to the updates (3.90) and (3.91) with $\hat{z} = \mu^{k+1}$, $z = \mu$, $\tilde{z} = \mu^k$ and $\hat{z} = \bar{\lambda}^{k+1}$, $z = \lambda$, $\tilde{z} = \lambda^k$ respectively and adding the two inequalities result in,

$$\begin{aligned} & 2\gamma(\mathcal{L}(\mathbf{x}^{k+1}, \bar{\mathbf{y}}^{k+1}, \lambda, \mu) - \mathcal{L}(\mathbf{x}^{k+1}, \bar{\mathbf{y}}^{k+1}, \bar{\lambda}^{k+1}, \mu^{k+1})) \\ & \leq \|\mu^k - \mu\|^2 + \|\lambda^k - \lambda\|^2 - \|\mu^{k+1} - \mu\|^2 - \|\mu^{k+1} - \mu^k\|^2 - \|\bar{\lambda}^{k+1} - \lambda\|^2 - \|\bar{\lambda}^{k+1} - \lambda^k\|^2. \end{aligned}$$

Taking limit over the appropriate sub-sequences on both sides of the above inequality, we get,

$$\mathcal{L}(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda, \mu) \leq \mathcal{L}(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty). \quad (3.92)$$

Using (3.89) with $\mathbf{x} = \mathbf{x}^*$, $\mathbf{y} = \mathbf{y}^*$, (3.92) with $\lambda = \lambda^*$, $\mu = \mu^*$, the saddle point relation

in Assumption 16 and (3.76), we get,

$$\begin{aligned}\mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda^*, \mu^*) &\leq \mathcal{L}(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^*, \mu^*) \leq \mathcal{L}(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty) \leq \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda^\infty, \mu^\infty) \\ &\leq \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda^*, \mu^*).\end{aligned}$$

$$\text{Therefore, } \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda^*, \mu^*) = \mathcal{L}(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty). \quad (3.93)$$

This implies that $\mathbf{F}(\mathbf{x}^\infty) + \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}^\infty) + \lambda^{\infty\top}(\mathbf{x}^\infty - \mathbf{y}^\infty) + \mu^{\infty\top}(\mathbf{A}\mathbf{x}^\infty - \mathbf{b}) = \mathbf{F}(\mathbf{x}^*) < \infty$. Therefore, $\mathbf{y}^\infty \in \mathcal{C}_0$. Further, since, $\mathbf{x}^\infty = \mathbf{y}^\infty$ as $\|\lambda^{k+1} - \lambda^k\| \rightarrow 0$ and $\mathbf{A}\mathbf{x}^\infty = \mathbf{b}$ since, $\|\mu^{k+1} - \mu^k\| \rightarrow 0$. Thus, $\mathbf{F}(\mathbf{x}^\infty) = \mathbf{F}(\mathbf{x}^*)$. To complete the proof, we show that the $(\mathbf{x}^k, \mathbf{y}^k, \lambda^k, \mu^k)$ has a unique limit point; we utilize the same argument as in [186], Theorem 1. Let $r_1 = (\mathbf{x}_1^\infty, \mathbf{y}_1^\infty, \lambda_1^\infty, \mu_1^\infty)$ and $r_2 = (\mathbf{x}_2^\infty, \mathbf{y}_2^\infty, \lambda_2^\infty, \mu_2^\infty)$ be any two limit points of $(\mathbf{x}^k, \mathbf{y}^k, \lambda^k, \mu^k)$. As shown above, both r_1 and r_2 are saddle points of $\mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda, \mu)$. Then, from (3.87) for appropriate sub-sequences we have $\lim_{k_t \rightarrow \infty} \|\mathbf{x}^{k_t} - \mathbf{x}_i^\infty\|^2 + \|\mathbf{y}^{k_t} - \mathbf{y}_i^\infty\|^2 + \|\lambda^{k_t} - \lambda_i^\infty\|^2 + \|\mu^{k_t} - \mu_i^\infty\|^2 = \hat{r}_j < 0$, for $j = 1, 2$. Consider,

$$\begin{aligned}\|\mathbf{x}^k - \mathbf{x}_1^\infty\|^2 + \|\mathbf{y}^k - \mathbf{y}_1^\infty\|^2 + \|\lambda^k - \lambda_1^\infty\|^2 + \|\mu^k - \mu_1^\infty\|^2 - \|\mathbf{x}^k - \mathbf{x}_2^\infty\|^2 - \|\mathbf{y}^k - \mathbf{y}_2^\infty\|^2 \\ - \|\lambda^k - \lambda_2^\infty\|^2 - \|\mu^k - \mu_2^\infty\|^2 = \|r_1\|^2 - \|r_2\|^2 - 2(r_1 - r_2)^\top (\mathbf{x}^{k\top}, \mathbf{y}^{k\top}, \lambda^{k\top}, \mu^{k\top}).\end{aligned}$$

Taking the limit on both sides for each limit point we obtain,

$$\begin{aligned}\hat{r}_1 - \hat{r}_2 &= \|r_1\|^2 - \|r_2\|^2 - 2(\mathbf{x}_1^{\infty\top}(\mathbf{x}_1^\infty - \mathbf{x}_2^\infty) + \mathbf{y}_1^{\infty\top}(\mathbf{y}_1^\infty - \mathbf{y}_2^\infty) + \lambda_1^{\infty\top}(\lambda_1^\infty - \lambda_2^\infty) \\ &\quad + \mu_1^{\infty\top}(\mu_1^\infty - \mu_2^\infty)) = -\|r_1 - r_2\|^2, \text{ and,} \\ \hat{r}_1 - \hat{r}_2 &= \|r_1\|^2 - \|r_2\|^2 - 2(\mathbf{x}_2^{\infty\top}(\mathbf{x}_1^\infty - \mathbf{x}_2^\infty) + \mathbf{y}_2^{\infty\top}(\mathbf{y}_1^\infty - \mathbf{y}_2^\infty) + \lambda_2^{\infty\top}(\lambda_1^\infty - \lambda_2^\infty) \\ &\quad + \mu_2^{\infty\top}(\mu_1^\infty - \mu_2^\infty)) = \|r_1 - r_2\|^2.\end{aligned}$$

Therefore, we must have $\|r_1 - r_2\| = 0$ and hence, $(\mathbf{x}^\infty, \mathbf{y}^\infty, \lambda^\infty, \mu^\infty)$ is unique. This completes the proof. $\square$

Next, two estimates of rate of convergence for the proposed **DC-DistADMM** algorithm are provided.

Sub-linear Rate of Convergence:

Here, the convergence of the proposed **DC-DistADMM** algorithm for the case when individual functions $\tilde{f}_i$ are convex but not necessarily differentiable is analyzed.

Theorem 22 (Sub-linear rate of convergence). Let $\{\mathbf{x}^k\}_{k \geq 1}$, $\{\mathbf{y}^k\}_{k \geq 1}$, $\{\lambda^k\}_{k \geq 1}$, and $\{\mu^k\}_{k \geq 1}$, be the sequences generated by Algorithm 6. Let $\eta_k = \frac{1}{k^{2+q}}$, $q \in (0, 1)$. Let assumptions 13-16 hold, then for all $k \geq 1$,

$$\mathbf{F}(\hat{\mathbf{x}}^k) - \mathbf{F}(\mathbf{x}^*) = O(1/k), \quad \|\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k\| = O(1/k),$$

$$\text{and, } \|\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}\| = O(1/k).$$

where, $\hat{\mathbf{x}}^k := \frac{1}{k} \sum_{s=0}^{k-1} \mathbf{x}^{s+1}$, $\hat{\mathbf{y}}^k := \frac{1}{k} \sum_{s=0}^{k-1} \mathbf{y}^{s+1}$.

Proof. From (3.78) we have, for $i = 1, \dots, n$,

$$df_i(x_i^{k+1}) + \lambda_i^{k+1} + A_i^\top \mu_i^{k+1} + \gamma(y_i^{k+1} - y_i^k) = 0,$$

where, $df_i(x_i^{k+1})$ is a sub-gradient of f_i at x_i^{k+1} . Writing the above n inequalities compactly,

$$d\mathbf{F}(\mathbf{x}^{k+1}) + \lambda^{k+1} + \mathbf{A}^\top \mu^{k+1} + \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k) = 0,$$

where, $d\mathbf{F}(\mathbf{x}^{k+1})$ is a vector with subgradients $df_i(x_i^{k+1})$ stacked together respectively. Further, using (3.79) $d(\bar{\mathbf{y}}^{k+1}) + \bar{\lambda}^{k+1} = 0$, where $d(\bar{\mathbf{y}}^{k+1})$ is a sub-gradient of $\mathcal{I}_{\mathcal{C}_0}$ at $\bar{\mathbf{y}}^{k+1}$. Noticing that $\mathbf{F}$ and $\mathcal{I}_{\mathcal{C}_0}$ are convex functions and using (3.63) we have,

$$\begin{aligned} \mathbf{F}(\mathbf{x}^{k+1}) - \mathbf{F}(\mathbf{x}^*) + \mathcal{I}_{\mathcal{C}_0}(\bar{\mathbf{y}}^{k+1}) - \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}^*) &\leq -[\mathbf{x}^* - \mathbf{x}^{k+1}]^\top d\mathbf{F}(\mathbf{x}^{k+1}) - [\mathbf{y}^* - \bar{\mathbf{y}}^{k+1}]^\top d(\bar{\mathbf{y}}^{k+1}) \\ &\leq [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\lambda^{k+1} + \mathbf{A}^\top \mu^{k+1} + \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k)] - [\mathbf{y}^* - \bar{\mathbf{y}}^{k+1}]^\top \bar{\lambda}^{k+1}. \end{aligned}$$

Note that $\mathcal{I}_{\mathcal{C}_0}(\bar{\mathbf{y}}^{k+1}) = \mathcal{I}_{\mathcal{C}_0}(\mathbf{y}^*) = 0$. Using (3.61) and (3.63),

$$\begin{aligned} \mathbf{F}(\mathbf{x}^{k+1}) - \mathbf{F}(\mathbf{x}^*) &\leq [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\lambda^{k+1} + \mathbf{A}^\top \mu^{k+1} + \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k)] \\ &\quad - [\mathbf{y}^* - \bar{\mathbf{y}}^{k+1}]^\top [\lambda^{k+1} + \gamma(\mathbf{y}^{k+1} - \bar{\mathbf{y}}^{k+1})] \\ &\leq [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\lambda^{k+1} + \mathbf{A}^\top \mu^{k+1} + \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k)] \\ &\quad - [\mathbf{y}^* - \bar{\mathbf{y}}^{k+1}]^\top \lambda^{k+1} + \gamma e^{k+1\top} (\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*) \\ &= [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\lambda^{k+1} + \mathbf{A}^\top \mu^{k+1} + \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k)] \\ &\quad - [\mathbf{y}^* - \bar{\mathbf{y}}^{k+1}]^\top \lambda^{k+1} + e^{k+1\top} (-\lambda^{k+1} + \gamma(\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*)) \\ &= [\mathbf{y}^{k+1} - \mathbf{x}^{k+1}]^\top \lambda^{k+1} + \gamma[\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] \\ &\quad + [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top \mathbf{A}^\top \mu^{k+1} + e^{k+1\top} (-\lambda^{k+1} + \gamma(\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*)). \end{aligned}$$

Therefore, for any λ and μ , using (3.76) we have

$$\begin{aligned} \mathbf{F}(\mathbf{x}^{k+1}) - \mathbf{F}(\mathbf{x}^*) + \lambda^\top (\mathbf{x}^{k+1} - \mathbf{y}^{k+1}) + \mu^\top (\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}) &\leq [\lambda^{k+1} - \lambda]^\top [\mathbf{y}^{k+1} - \mathbf{x}^{k+1}] \\ &+ \gamma[\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] + [\mu - \mu^{k+1}]^\top \mathbf{A}(\mathbf{x}^{k+1} - \mathbf{b}) + e^{k+1\top} (-\lambda^{k+1} + \gamma(\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*)). \end{aligned}$$

Using (3.61) and (3.62), we get,

$$\begin{aligned} \mathbf{F}(\mathbf{x}^{k+1}) - \mathbf{F}(\mathbf{x}^*) + \lambda^\top (\mathbf{x}^{k+1} - \mathbf{y}^{k+1}) + \mu^\top (\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}) &\leq \frac{1}{\gamma}[\lambda - \lambda^{k+1}]^\top [\lambda^{k+1} - \lambda^k] \quad (3.94) \\ &+ \gamma[\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] + \frac{1}{\gamma}[\mu - \mu^{k+1}]^\top [\mu^{k+1} - \mu^k] + e^{k+1\top} (-\lambda^{k+1} + \gamma(\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*)). \end{aligned}$$

Using the identity (3.72) for any $s \geq 0$ we get,

$$\frac{1}{\gamma}[\lambda - \lambda^{s+1}]^\top [\lambda^{s+1} - \lambda^s] = \frac{1}{2\gamma}(\|\lambda - \lambda^s\|^2 - \|\lambda - \lambda^{s+1}\|^2 - \|\lambda^{s+1} - \lambda^s\|^2), \quad (3.95)$$

$$\begin{aligned} \gamma[\mathbf{x}^* - \mathbf{x}^{s+1}]^\top [\mathbf{y}^{s+1} - \mathbf{y}^s] &= \frac{\gamma}{2}(\|\mathbf{x}^{s+1} - \mathbf{y}^{s+1}\|^2 - \|\mathbf{x}^{s+1} - \mathbf{y}^s\|^2 + \|\mathbf{x}^* - \mathbf{y}^s\|^2 \\ &\quad - \|\mathbf{x}^* - \mathbf{y}^{s+1}\|^2), \end{aligned} \quad (3.96)$$

$$\frac{1}{\gamma}[\mu - \mu^{k+1}]^\top [\mu^{k+1} - \mu^k] = \frac{1}{2\gamma}(\|\mu - \mu^k\|^2 - \|\mu - \mu^{k+1}\|^2 - \|\mu^{k+1} - \mu^k\|^2). \quad (3.97)$$

Note that from Lemma 17, (3.61) and (3.71), under Assumption 14, $e^{k+1\top}(-\lambda^{k+1} + \gamma(\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*)) = -e^{k+1\top}\lambda^{k+1} + \gamma e^{k+1\top}\mathbf{y}^{k+1} - \gamma\|e^{k+1}\|^2 - e^{k+1\top}\mathbf{y}^* \leq \|e^{k+1}\|\|\lambda^{k+1}\| + \gamma\|e^{k+1}\|\|\mathbf{y}^{k+1}\| + \|e^{k+1}\|\|\mathbf{y}^*\| \leq (k+1)\mathbf{Q}\sqrt{n}\eta_{k+1} + (\gamma n(\mathcal{M} + \|x^*\|) + 2(k+1)\sqrt{n}\mathbf{Q})\eta_{k+1} + \sqrt{n}\|\mathbf{y}^*\|\eta_{k+1}$. Let $\mathbf{M} := \gamma n(\mathcal{M} + \|x^*\|) + \sqrt{n}\|\mathbf{y}^*\|$. From (3.94), (3.95), (3.96) and (3.97) for any λ, μ and $s \geq 0$ we get,

$$\begin{aligned} \mathbf{F}(\mathbf{x}^{s+1}) - \mathbf{F}(\mathbf{x}^*) + \lambda^\top (\mathbf{x}^{s+1} - \mathbf{y}^{s+1}) + \mu^\top (\mathbf{A}\mathbf{x}^{s+1} - \mathbf{b}) &\leq \frac{1}{2\gamma}(\|\lambda - \lambda^s\|^2 - \|\lambda - \lambda^{s+1}\|^2) + \\ &\frac{\gamma}{2}(\|\mathbf{x}^* - \mathbf{y}^s\|^2 - \|\mathbf{x}^* - \mathbf{y}^{s+1}\|^2) + \frac{1}{2\gamma}(\|\mu - \mu^s\|^2 - \|\mu - \mu^{s+1}\|^2) + \frac{\mathbf{M}}{s^{2+q}} + \frac{3\sqrt{n}\mathbf{Q}(s+1)}{(s+1)^{2+q}}. \end{aligned}$$

Summing from $s = 0$ to $k-1$ and dividing by $1/k$ gives,

$$\begin{aligned} \frac{1}{k} \sum_{s=0}^{k-1} [\mathbf{F}(\mathbf{x}^{s+1}) - \mathbf{F}(\mathbf{x}^*) + \lambda^\top (\mathbf{x}^{s+1} - \mathbf{y}^{s+1}) + \mu^\top (\mathbf{A}\mathbf{x}^{s+1} - \mathbf{b})] &\leq \frac{\gamma\|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} \\ &+ \frac{\|\lambda - \lambda^0\|^2}{2\gamma k} + \frac{\|\mu - \mu^0\|^2}{2\gamma k} + \frac{\mathbf{M} \sum_{s=0}^{k-1} \frac{1}{s^{2+q}}}{k} + \frac{3\sqrt{n}\mathbf{Q} \sum_{s=0}^{k-1} (s+1) \frac{1}{(s+1)^{2+q}}}{k}. \end{aligned}$$

Let $\hat{\mathbf{x}}^k := \frac{1}{k} \sum_{s=0}^{k-1} \mathbf{x}^{s+1}$, and $\hat{\mathbf{y}}^k := \frac{1}{k} \sum_{s=0}^{k-1} \mathbf{y}^{s+1}$. Since, $\mathbf{F}$ is a convex function we have, $\frac{1}{k} \sum_{s=0}^{k-1} \mathbf{F}(\mathbf{x}^{s+1}) \geq \mathbf{F}(\hat{\mathbf{x}}^k)$. Therefore,

$$\begin{aligned} \mathbf{F}(\hat{\mathbf{x}}^k) - \mathbf{F}(\mathbf{x}^*) + \lambda^\top (\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k) + \mu^\top (\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}) &\leq \frac{\gamma \|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} + \frac{\|\lambda - \lambda^0\|^2}{2\gamma k} \\ &+ \frac{\|\mu - \mu^0\|^2}{2\gamma k} + \frac{\mathbf{M}\zeta(2+q) + 3\sqrt{n}\mathbf{Q}\zeta(1+q)}{k}, \end{aligned} \quad (3.98)$$

where, $\zeta(\cdot)$ is the Riemann zeta function. Let $\mu = 0, \lambda = 0$, therefore, we get,

$$\mathbf{F}(\hat{\mathbf{x}}^k) - \mathbf{F}(\mathbf{x}^*) \leq \frac{\gamma \|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} + \frac{\|\lambda^0\|^2 + \|\mu^0\|^2}{2\gamma k} + \frac{\mathbf{M}\zeta(2+q) + 3\sqrt{n}\mathbf{Q}\zeta(1+q)}{k} = O(1/k).$$

Define, $\hat{\bar{\mathbf{y}}}^k := \frac{1}{k} \sum_{s=0}^{k-1} \bar{\mathbf{y}}^{s+1}$, where, $\bar{\mathbf{y}}^{s+1}$ is as defined in (3.60). Since, $(\mathbf{x}^*, \mathbf{y}^*, \lambda^*, \mu^*)$ is a saddle point we have, $\mathcal{L}(\mathbf{x}^*, \mathbf{y}^*, \lambda, \mu) \leq \mathcal{L}(\hat{\mathbf{x}}^k, \hat{\bar{\mathbf{y}}}^k, \lambda^*, \mu^*)$. Therefore, $\mathbf{F}(\hat{\mathbf{x}}^k) - \mathbf{F}(\mathbf{x}^*) + \lambda^{*\top} (\hat{\mathbf{x}}^k - \hat{\bar{\mathbf{y}}}^k) + \lambda^{*\top} (\hat{\bar{\mathbf{y}}}^k - \hat{\mathbf{y}}^k) + \mu^{*\top} (\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}) \geq 0$. Let $\lambda = \lambda^* + \frac{\hat{\mathbf{x}}^k - \hat{\bar{\mathbf{y}}}^k}{\|\hat{\mathbf{x}}^k - \hat{\bar{\mathbf{y}}}^k\|}$ and $\mu = \mu^*$. Therefore,

$$\begin{aligned} \|\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k\| &\leq \mathbf{F}(\hat{\mathbf{x}}^k) - \mathbf{F}(\mathbf{x}^*) + \lambda^{*\top} (\hat{\mathbf{x}}^k - \hat{\bar{\mathbf{y}}}^k) + \mu^{*\top} (\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}) + \lambda^{*\top} (\hat{\bar{\mathbf{y}}}^k - \hat{\mathbf{y}}^k) + \|\hat{\mathbf{x}}^k - \hat{\bar{\mathbf{y}}}^k\| \\ &\leq \frac{\gamma \|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} + \frac{1 + \|\lambda^0 - \lambda^*\|^2}{\gamma k} + \frac{\|\mu^* - \mu^0\|^2}{2\gamma k} + \frac{\mathbf{M}\zeta(2+q) + 3\sqrt{n}\mathbf{Q}\zeta(1+q)}{k} \\ &\quad + \lambda^{*\top} (\hat{\bar{\mathbf{y}}}^k - \hat{\mathbf{y}}^k). \end{aligned}$$

Note that, $\|\lambda^{*\top} (\hat{\bar{\mathbf{y}}}^k - \hat{\mathbf{y}}^k)\| \leq \|\lambda^*\| \|\frac{1}{k} \sum_{s=0}^{k-1} [\mathbf{y}^{s+1} - \bar{\mathbf{y}}^{s+1}]\| = \frac{\|\lambda^*\|}{k} \sum_{s=0}^{k-1} \|\mathbf{y}^{s+1} - \bar{\mathbf{y}}^{s+1}\| = \frac{\|\lambda^*\|}{k} \sum_{s=0}^{k-1} \|e^{s+1}\| \leq \frac{\|\lambda^*\|}{k} \sum_{s=0}^{k-1} \sqrt{n}\eta_{s+1} \leq \frac{\sqrt{n}\|\lambda^*\|}{k} \sum_{s=0}^{\infty} \eta_{s+1} = \frac{\sqrt{n}\|\lambda^*\|}{k} \zeta(2+q)$. Therefore,

$$\begin{aligned} \|\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k\| &\leq \frac{\gamma \|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} + \frac{1 + \|\lambda^0 - \lambda^*\|^2}{\gamma k} + \frac{\|\mu^* - \mu^0\|^2}{2\gamma k} \\ &\quad + \frac{(\mathbf{M} + \sqrt{n}\|\lambda^*\|)\zeta(2+q) + 3\sqrt{n}\mathbf{Q}\zeta(1+q)}{k} = O(1/k). \end{aligned}$$

Alternatively, let, $\mu = \mu^* + \frac{\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}}{\|\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}\|}$ and $\lambda = \lambda^*$, we get,

$$\begin{aligned}
\|\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}\| &\leq \mathbf{F}(\hat{\mathbf{x}}^k) - \mathbf{F}(\mathbf{x}^*) + \lambda^{*\top}(\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k) + \mu^{*\top}(\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}) + \lambda^{*\top}(\hat{\mathbf{y}}^k - \hat{\mathbf{y}}^k) + \|\mathbf{A}\hat{\mathbf{x}}^k - \mathbf{b}\| \\
&\leq \frac{\gamma\|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} + \frac{\|\lambda^0 - \lambda^*\|^2}{2\gamma k} + \frac{1 + \|\mu - \mu^0\|^2}{\gamma k} + \frac{\mathbf{M}\zeta(1+q)}{k} + \lambda^{*\top}(\hat{\mathbf{y}}^k - \hat{\mathbf{y}}^k) \\
&\leq \frac{\gamma\|\mathbf{x}^* - \mathbf{y}^0\|^2}{2k} + \frac{\|\lambda^0 - \lambda^*\|^2}{2\gamma k} + \frac{1 + \|\mu - \mu^0\|^2}{\gamma k} \\
&\quad + \frac{(\mathbf{M} + \sqrt{n}\|\lambda^*\|)\zeta(2+q) + 3\sqrt{n}\mathbf{Q}\zeta(1+q)}{k} = O(1/k).
\end{aligned}$$

□

Theorem 22 establishes that the objective function evaluated at the ergodic average of the optimization variables obtained by the proposed **DC-DistADMM** algorithm converges to the optimal value. In particular, the objective function value evaluated at the ergodic average converges to the optimal value at a rate of $O(1/k)$.

Remark 20. Let, $\hat{x}_i^k = \frac{1}{k} \sum_{s=0}^{k-1} x_i^s$ and $\hat{y}_i^k = \frac{1}{k} \sum_{s=0}^{k-1} y_i^s$ denote the ergodic averages of the variables x_i^k and y_i^k . Since, $\|\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k\| = O(1/k)$ it implies that for all $i \in \mathcal{V}$, $\|\hat{x}_i^k - \hat{y}_i^k\| = O(1/k)$. Further, since, for all $k \geq 0$ and $i, j \in \mathcal{V}$, $\|y_i^k - y_j^k\| \leq \eta_k$ (see (3.70)) it implies that $\|\hat{y}_i^k - \hat{y}_j^k\| \leq \frac{1}{k} \sum_{s=0}^{k-1} \|y_i^s - y_j^s\| \leq \frac{1}{k} \sum_{s=0}^{k-1} \eta_s \leq \frac{\zeta(2+q)}{k}$. Thus, $\|\hat{y}_i^k - \hat{y}_j^k\| = O(1/k)$. Therefore, for all $i, j \in \mathcal{V}$,

$$\|\hat{x}_i^k - \hat{x}_j^k\| = O(1/k) \text{ and } \lim_{k \rightarrow \infty} \|\hat{x}_i^k - \hat{x}_j^k\| = 0.$$

Thus, in practice, the **DC-DistADMM** algorithm can be implemented with an additional variable $\hat{x}_i^k$ tracking the ergodic average of x_i^k at each agent $i \in \mathcal{V}$ to achieve a consensual solution at the $O(1/k)$ rate of convergence given in Theorem 22 in a fully distributed manner. Note, that the variable $\hat{x}_i^k$ is computed locally by each agent without any additional cost of communication and a minor addition in cost of computation.

Remark 21. By Lemma 18 the total number of communication iterations performed until iteration k of the **DC-DistADMM** algorithm in Theorem 22, is upper bounded by $\mathcal{K} := \sum_{s=1}^k \bar{t}_s := O(k \log k)$. The communication complexity is within a factor of $\log k$ of the optimal lower bound $O(k)$.

Geometric rate of Convergence:

Here, a geometric rate of convergence for the proposed **DC-DistADMM** algorithm under the following assumption is established:

Assumption 17. *Each function $\tilde{f}_i$ is Lipschitz differentiable with constant L_{f_i} and restricted strongly convex with respect to the optimal solution x^* on $\mathcal{X}_i$ with parameter $\sigma_i > 0$.*

Remark 22. *Under Assumption 17 the constrained distributed problem (3.1) has a unique optimal solution. However, Assumption 17 is less restrictive than the standard global strong convexity assumption and makes the analysis presented here applicable to a bigger class of functions [187]. For example, the widely used logistic regression objective function is restricted strongly convex but not globally strongly convex [188].*

Theorem 23 (Geometric Rate of convergence). *Let $\{\mathbf{x}^k\}_{k \geq 1}$, $\{\mathbf{y}^k\}_{k \geq 1}$, $\{\lambda^k\}_{k \geq 1}$, and $\{\mu^k\}_{k \geq 1}$, be the sequences generated by Algorithm 6. Let assumptions 13,14,16, and 17 hold. Let $\Delta := (1 - \frac{1}{\delta}) \min\{1, \nu_{\min}(\mathbf{A}\mathbf{A}^\top)\}$ for any $\delta \in (1, 1 + \frac{2\sigma\gamma}{L^2 + \gamma^2}]$, where $L := \max_{1 \leq i \leq n} L_{f_i}$, $\sigma := \min_{1 \leq i \leq n} \sigma_i$, and $\nu_{\min}(\mathbf{A}\mathbf{A}^\top)$ is the minimum eigenvalue of $\mathbf{A}\mathbf{A}^\top$. Let $\eta_k = \rho^k$, where, $\rho \in [\frac{1}{1+\Delta}, 1)$. Let $x_i^k \in \text{relint}(\mathcal{X}_i \times \mathbb{R}_{\geq 0}^m), \forall i \in \mathcal{V}, k \geq 0$. Then the agent solution residual $s_k := \frac{\gamma}{2} \|\mathbf{x}^k - \mathbf{x}^*\|^2 + \frac{1}{2\gamma} \|\lambda^k - \lambda^*\|^2 + \frac{1}{2\gamma} \|\mu^k - \mu^*\|^2$ has the following relation: for any $K \geq 0$ and $\epsilon > 0$,*

$$s_K \leq \Upsilon \rho^K + O(\epsilon),$$

where, Υ is a finite constant defined in (3.106).

Proof. From the definition of $\mathbf{F}$ and Assumption 17 it follows that $\mathbf{F}$ is Lipschitz differentiable with constant $L := \max_{1 \leq i \leq n} L_{f_i}$ for all $\tilde{\mathbf{x}} \in \text{relint}(\mathcal{X})$, $\tilde{\mathbf{x}}_s \in \text{relint}(\mathbb{R}_{\geq 0}^m)$ and restricted strongly convex with respect to the optimal solution $\mathbf{x}^*$ on the set $\mathcal{X}$ with parameter $\sigma := \min_{1 \leq i \leq n} \sigma_i$. For sub-gradients $d\mathbf{F}(\mathbf{x}^{k+1}) \in \partial\mathbf{F}(\mathbf{x}^{k+1})$ and $d\mathbf{F}(\mathbf{x}^*) \in \partial\mathbf{F}(\mathbf{x}^*)$, using (3.73) and (3.78) under Assumption 17,

$$\begin{aligned} \sigma \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 &\leq [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [d\mathbf{F}(\mathbf{x}^{k+1}) - d\mathbf{F}(\mathbf{x}^*)] \\ &= [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [-\lambda^{k+1} - \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k) - \mathbf{A}^\top \mu^{k+1} + \lambda^* + \mathbf{A}^\top \mu^*] \\ &= [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\lambda^* - \lambda^{k+1}] - \gamma [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\mathbf{y}^{k+1} - \mathbf{y}^k] \\ &\quad + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] \end{aligned}$$

$$\begin{aligned}
&= [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\lambda^* - \lambda^{k+1}] + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] \\
&\quad + \gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{x}^k] + [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [2\lambda^k - \lambda^{k+1} - \lambda^{k-1}].
\end{aligned}$$

Using, ((3.83) and (3.84) we get,

$$\begin{aligned}
\sigma \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 &\leq [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top [\lambda^* - \lambda^{k+1}] + \gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{x}^k] \\
&\quad + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] + [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*]^\top [\lambda^{k+1} - \lambda^*] \\
&\quad - [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*]^\top \lambda^{k+1} + [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [2\lambda^k - \lambda^{k+1} - \lambda^{k-1}] \\
&= [\lambda^* - \lambda^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{y}^{k+1}] + \gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{x}^k] \\
&\quad + [\mathbf{x}^{k+1} - \mathbf{x}^*]^\top \mathbf{A}^\top [\mu^* - \mu^{k+1}] - [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*]^\top \lambda^{k+1} \\
&\quad + [\lambda^* - \lambda^{k+1}]^\top e^{k+1} + [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [2\lambda^k - \lambda^{k+1} - \lambda^{k-1}] \\
&= \frac{1}{\gamma} [\lambda^* - \lambda^{k+1}]^\top [\lambda^{k+1} - \lambda^k] + \gamma [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{x}^k] \\
&\quad + \frac{1}{\gamma} [\mu^* - \mu^{k+1}]^\top [\mu^{k+1} - \mu^k] - [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*]^\top \lambda^{k+1} \\
&\quad + [\lambda^* - \lambda^{k+1}]^\top e^{k+1} + [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [2\lambda^k - \lambda^{k+1} - \lambda^{k-1}],
\end{aligned}$$

where, in the second equality we used (3.71), and in the last equality we used (3.61) and (3.62). Let $t_k := 2\lambda^k - \lambda^{k+1} - \lambda^{k-1}$. Using (3.72) for $\frac{1}{\gamma}[\lambda^* - \lambda^{k+1}]^\top [\lambda^{k+1} - \lambda^k]$, $\frac{1}{\gamma}[\mu^* - \mu^{k+1}]^\top [\mu^{k+1} - \mu^k]$, and $\gamma[\mathbf{x}^* - \mathbf{x}^{k+1}]^\top [\mathbf{x}^{k+1} - \mathbf{x}^k]$,

$$\begin{aligned}
&\sigma \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 + \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2 + \frac{1}{2\gamma} \|\lambda^{k+1} - \lambda^k\|^2 \\
&\leq \left[\frac{\gamma}{2} \|\mathbf{x}^k - \mathbf{x}^*\|^2 + \frac{1}{2\gamma} \|\lambda^k - \lambda^*\|^2 + \frac{1}{2\gamma} \|\mu^k - \mu^*\|^2 \right] \\
&\quad - \left[\frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 + \frac{1}{2\gamma} \|\lambda^{k+1} - \lambda^*\|^2 + \frac{1}{2\gamma} \|\mu^{k+1} - \mu^*\|^2 \right] \\
&\quad - [\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*]^\top \lambda^{k+1} + [\lambda^* - \lambda^{k+1}]^\top e^{k+1} + [\mathbf{x}^* - \mathbf{x}^{k+1}]^\top t_k. \tag{3.99}
\end{aligned}$$

Observe from (3.73), (3.78), $\nabla \mathbf{F}(\mathbf{x}^{k+1}) - \nabla \mathbf{F}(\mathbf{x}^*) = -\lambda^{k+1} - \gamma(\mathbf{y}^{k+1} - \mathbf{y}^k) - \mathbf{A}^\top \mu^{k+1} + \lambda^* + \mathbf{A}^\top \mu^* = -\lambda^{k+1} + \lambda^* - \mathbf{A}^\top \mu^{k+1} + \mathbf{A}^\top \mu^* + \lambda^{k+1} + \lambda^{k-1} - 2\lambda^k - \gamma(\mathbf{x}^{k+1} - \mathbf{x}^k)$, where we used (3.61) to get a relation between $\mathbf{y}^{k+1}$ and $\mathbf{y}^k$. Therefore, $\gamma(\mathbf{x}^{k+1} - \mathbf{x}^k)$ is the summation of the terms $\mathbf{q}_1 := \nabla \mathbf{F}(\mathbf{x}^*) - \nabla \mathbf{F}(\mathbf{x}^{k+1})$ and $\mathbf{r}_1 := \lambda^* - \lambda^{k+1} + \mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} + \lambda^{k+1} + \lambda^{k-1} - 2\lambda^k$. Therefore, for any $\delta > 1$, we can apply the inequality

$$\|\mathbf{q}_1 + \mathbf{r}_1\|^2 + (\delta - 1)\|\mathbf{q}_1\|^2 \geq (1 - \frac{1}{\delta})\|\mathbf{r}_1\|^2,$$

$$\begin{aligned} & \gamma^2 \|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2 + (\delta - 1) \|\nabla \mathbf{F}(\mathbf{x}^{k+1}) - \nabla \mathbf{F}(\mathbf{x}^*)\|^2 \\ & \geq \left(1 - \frac{1}{\delta}\right) \|\lambda^* - \lambda^{k+1} + \mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\|^2 \\ & \geq \left(1 - \frac{1}{\delta}\right) \left[\|\lambda^{k+1} - \lambda^*\|^2 + \nu_{\min}(\mathbf{A}\mathbf{A}^\top) \|\mu^{k+1} - \mu^*\|^2 - 2\|t_k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1}\| \right. \\ & \quad \left. + \|t_k\|^2 - 2\|\lambda^{k+1} - \lambda^*\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\| \right], \end{aligned}$$

where, $\nu_{\min}(\mathbf{A}\mathbf{A}^\top) > 0$. By Lipschitz differentiability of $\mathbf{F}$,

$$\begin{aligned} & \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2 + \frac{(\delta - 1)L^2}{2\gamma} \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 \\ & \geq \frac{(1 - \frac{1}{\delta})}{2\gamma} \left[\|\lambda^{k+1} - \lambda^*\|^2 + \nu_{\min}(\mathbf{A}\mathbf{A}^\top) \|\mu^{k+1} - \mu^*\|^2 - 2\|t_k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1}\| \right. \\ & \quad \left. - 2\|\lambda^{k+1} - \lambda^*\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\| \right]. \end{aligned} \quad (3.100)$$

Similarly, $\lambda^{k+1} - \lambda^k = \gamma(\mathbf{x}^{k+1} - \mathbf{y}^{k+1}) = \gamma(\mathbf{x}^{k+1} - \mathbf{x}^*) + \gamma(\mathbf{y}^* - \mathbf{y}^{k+1})$. Therefore, $\lambda^{k+1} - \lambda^k$ is the summation of $\mathbf{q}_2 := \gamma(\mathbf{x}^{k+1} - \mathbf{x}^*)$ and $\mathbf{r}_2 := \gamma(\mathbf{y}^* - \mathbf{y}^{k+1})$. Therefore, for any $\delta > 1$, we can apply the inequality $\|\mathbf{q}_2 + \mathbf{r}_2\|^2 + (\delta - 1)\|\mathbf{q}_2\|^2 \geq (1 - \frac{1}{\delta})\|\mathbf{r}_2\|^2$, and obtain,

$$\begin{aligned} & \|\lambda^{k+1} - \lambda^k\|^2 + \gamma^2(\delta - 1)\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 \geq (1 - \frac{1}{\delta})\|\gamma(\mathbf{y}^{k+1} - \mathbf{y}^*)\|^2 \\ & = \left(1 - \frac{1}{\delta}\right) \|\lambda^{k+1} - \lambda^k - \gamma(\mathbf{x}^{k+1} - \mathbf{x}^*)\|^2 \\ & \geq (1 - \frac{1}{\delta}) \left[\|\lambda^{k+1} - \lambda^k\|^2 + \gamma^2\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 - 2\gamma\|\lambda^{k+1} - \lambda^k\| \|\mathbf{x}^{k+1} - \mathbf{x}^*\| \right]. \end{aligned} \quad (3.101)$$

Therefore, using (3.100) and (3.101) we get,

$$\begin{aligned} & \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2 + \frac{1}{2\gamma} \|\lambda^{k+1} - \lambda^k\|^2 + \frac{(\delta - 1)(\frac{L^2}{\gamma} + \gamma)}{2} \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 \\ & \geq \left(1 - \frac{1}{\delta}\right) \left[\frac{1}{2\gamma} \|\lambda^{k+1} - \lambda^k\|^2 + \frac{\gamma}{2} \|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 + \frac{\nu_{\min}(\mathbf{A}\mathbf{A}^\top)}{2\gamma} \|\mu^{k+1} - \mu^*\|^2 \right] \\ & \quad - \frac{(1 - 1/\delta)}{\gamma} \left[\|\lambda^{k+1} - \lambda^k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\| \right. \\ & \quad \left. + \|t_k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1}\| + \gamma\|\lambda^{k+1} - \lambda^k\| \|\mathbf{x}^{k+1} - \mathbf{x}^*\| \right]. \end{aligned}$$

Let $\Delta := (1 - \frac{1}{\delta}) \min\{1, \nu_{\min}(\mathbf{A}\mathbf{A}^\top)\}$ and $s_k := \frac{\gamma}{2}\|\mathbf{x}^k - \mathbf{x}^*\|^2 + \frac{1}{2\gamma}\|\lambda^k - \lambda^*\|^2 + \frac{1}{2\gamma}\|\mu^{k+1} - \mu^*\|^2$. Therefore, for $\delta \in (1, 1 + \frac{2\sigma\gamma}{L^2 + \gamma^2}]$, we have,

$$\begin{aligned} & \sigma\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2 + \frac{\gamma}{2}\|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2 + \frac{1}{2\gamma}\|\lambda^{k+1} - \lambda^k\|^2 \\ & \geq \Delta s_{k+1} - \frac{(1 - 1/\delta)}{\gamma} [\|\lambda^{k+1} - \lambda^*\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\| + \|t_k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1}\| \\ & \quad + \gamma\|\lambda^{k+1} - \lambda^k\| \|\mathbf{x}^{k+1} - \mathbf{x}^*\|]. \end{aligned} \quad (3.102)$$

Therefore, from (3.99) and (3.102), we have

$$\begin{aligned} s_{k+1} & \leq \left(\frac{1}{1 + \Delta}\right)s_k + \left(\frac{1}{1 + \Delta}\right) \left(\|\lambda^{k+1} - \lambda^*\| \|e^{k+1}\| \|\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*\| \|\lambda^{k+1}\| + \|\mathbf{x}^* - \mathbf{x}^{k+1}\| \|t_k\| \right. \\ & \quad + \frac{(1 - 1/\delta)}{\gamma} [\|\lambda^{k+1} - \lambda^*\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\| + \|t_k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1}\| \\ & \quad \left. + \gamma\|\lambda^{k+1} - \lambda^k\| \|\mathbf{x}^{k+1} - \mathbf{x}^*\| \right). \end{aligned} \quad (3.103)$$

Using (3.87),

$$\|\lambda^k - \lambda^*\| \leq \sqrt{2\gamma(s_0 + \frac{\mathbf{R}}{1 - \rho} + \frac{3\sqrt{n}\mathbf{Q}\rho}{(1 - \rho)^2})} := \Xi. \quad (3.104)$$

Let $S_1 := \|\lambda^{k+1} - \lambda^*\| \|e^{k+1}\|$, $S_2 := \|\bar{\mathbf{y}}^{k+1} - \mathbf{y}^*\| \|\lambda^{k+1}\|$, $S_3 := \|\mathbf{x}^* - \mathbf{x}^{k+1}\| \|t_k\|$, $S_4 := \frac{(1 - 1/\delta)}{\gamma} \|\lambda^{k+1} - \lambda^*\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1} - t_k\|$, $S_5 := \frac{(1 - 1/\delta)}{\gamma} \|t_k\| \|\mathbf{A}^\top \mu^* - \mathbf{A}^\top \mu^{k+1}\|$ and $S_6 := (1 - 1/\delta) \|\lambda^{k+1} - \lambda^k\| \|\mathbf{x}^{k+1} - \mathbf{x}^*\|$. Note $S_1 = \|\lambda^{k+1} - \lambda^*\| \|e^{k+1}\| \leq \sqrt{n}\Xi\rho^{k+1}$. Further, from Theorem 21, there exist $\mathbf{C} < \infty$ and $\mathbf{S}^k$ with $\lim_{k \rightarrow \infty} \mathbf{S}^k = 0$ such that $S_2 + S_3 + S_4 + S_5 + S_6 \leq \mathbf{C}\mathbf{S}^k$. Therefore, from (3.103) and using the fact $\rho^{k+1} \leq \rho^k, \forall k$

$$s_{k+1} \leq \left(\frac{1}{1 + \Delta}\right)s_k + \sqrt{n}\Xi \left(\frac{1}{1 + \Delta}\right)\rho^k + \mathbf{C} \left(\frac{1}{1 + \Delta}\right)\mathbf{S}^k. \quad (3.105)$$

By repeated substitution for $k = 0, 1, \dots, K$ we get,

$$\begin{aligned}
s_K &\leq \left(\frac{1}{1+\Delta}\right)^K s_0 + \sqrt{n}\Xi \sum_{t=0}^K \left(\frac{1}{1+\Delta}\right)^t \rho^{K-t} + \mathbf{C} \sum_{t=0}^K \left(\frac{1}{1+\Delta}\right)^{K-t+1} \mathbf{S}^t \\
&= \left(\frac{1}{1+\Delta}\right)^K s_0 + \rho^K \sqrt{n}\Xi \sum_{t=1}^K \left(\frac{1}{1+\Delta}\right)^t \rho^{-t} + \mathbf{C} \sum_{t=0}^K \left(\frac{1}{1+\Delta}\right)^{K-t+1} \mathbf{S}^t \\
&\leq \rho^K \left(s_0 + \sqrt{n}\Xi \sum_{t=1}^K \left(\frac{1}{1+\Delta}\right)^t \rho^{-t} \right) + \mathbf{C} \sum_{t=0}^K \left(\frac{1}{1+\Delta}\right)^{K-t+1} \mathbf{S}^t.
\end{aligned}$$

Given, $\epsilon > 0$, let K' be such that $\mathbf{S}^k < \epsilon$ for all $k \geq K'$ (Theorem 21 guarantees existence of such K'). Let $\bar{\mathbf{S}} := \max_{0 \leq t \leq K'} \mathbf{S}^t$, therefore,

$$\begin{aligned}
s_K &\leq \rho^K \left(s_0 + \sqrt{n}\Xi \sum_{t=1}^K \left(\frac{1}{1+\Delta}\right)^t \rho^{-t} + \mathbf{C}\bar{\mathbf{S}} \sum_{t=0}^{K'} \rho^{1-t} \right) + \epsilon \left(\mathbf{C} \sum_{t=K'}^K \rho^{K-t+1} \right) \\
&\leq \rho^K \left(s_0 + \frac{\sqrt{n}\Xi \rho(1+\Delta)}{\rho(1+\Delta) - 1} + \frac{\mathbf{C}\bar{\mathbf{S}}(\rho^{-K'+1} - \rho^2)}{1 - \rho} \right) + \epsilon \left(\frac{\mathbf{C}}{1 - \rho} \right) \leq \Upsilon \rho^K + O(\epsilon),
\end{aligned}$$

where,

$$\Upsilon := s_0 + \frac{\mathbf{S}_2 \rho(1+\Delta)}{\rho(1+\Delta) - 1}. \quad (3.106)$$

This completes the proof. $\square$

Remark 23. Although, Theorem 23 provides a tight bound on the requirement of ρ , in practice the tolerance sequence parameter ρ can be chosen from the interval $[\frac{2}{3}, 1)$ which does not require the knowledge of the restricted strong convexity parameter σ_i of the individual functions and the optimal solution x^* . We explain this below:

From the definition of Δ we have, $\Delta \leq (1 - 1/\delta)$. Using the condition on δ in Theorem 23

we get,

$$\begin{aligned}
1 < \delta &\leq \frac{L^2 + \gamma^2 + 2\sigma\gamma}{L^2 + \gamma^2} \\
1 > \frac{1}{\delta} &\geq \frac{L^2 + \gamma^2}{L^2 + \gamma^2 + 2\sigma\gamma} \\
-1 < -\frac{1}{\delta} &\leq -\frac{L^2 + \gamma^2}{L^2 + \gamma^2 + 2\sigma\gamma} \\
0 < 1 - \frac{1}{\delta} &\leq \frac{2\sigma\gamma}{L^2 + \gamma^2 + 2\sigma\gamma} = \frac{1}{\frac{L^2 + \gamma^2}{2\sigma\gamma} + 1} \leq \frac{1}{2},
\end{aligned}$$

where, the last inequality follows from the AM-GM inequality and the fact that $L \geq \sigma$. Thus, for $\frac{1}{\Delta+1} = (1 + (1 - \frac{1}{\delta}))^{-1}$

$$\frac{2}{3} \leq \frac{1}{\Delta+1} < 1.$$

Note that the $\rho \in [\frac{1}{1+\Delta}, 1)$. Thus, $\rho \in [\frac{2}{3}, 1)$. The simulation results demonstrate the theory's applicability and show good performance for the **DC-DistADMM** algorithm with $\rho = 0.75 \in [\frac{2}{3}, 1)$.

Remark 24. By Lemma 18 the total number of communication iterations performed until iteration k of the **DC-DistADMM** algorithm in Theorem 23, is upper bounded by $\mathcal{K} := \sum_{s=1}^k \bar{t}_s := O(k^2)$. Thus, compared to Theorem 22 the improved rate of convergence leads to an increase in the number of communication iterations (Remark 21). Moreover, compared to algorithms utilizing multiple communication steps in the literature [116], [175]–[178], [189]–[191] the **DC-DistADMM** algorithm has the same communication complexity. In particular, methods in [176], [178], [189] have communication complexity, $O(k^2)$, in getting a geometric rate. Schemes proposed in [116], [175], [177], [190], [191] have the same, $O(k \log k)$, communication complexity in getting a $O(1/k)$ rate of convergence.

Remark 25. The geometric rate of convergence for **DC-DistADMM** algorithm does not follow from the existing centralized results (see [192], [193]) as problem (3.8) does not satisfy the assumptions used in these works. In particular, the function $\mathcal{I}_{C_0}(\mathbf{y})$ is not differentiable and does not meet the requirements of being global Lipschitz differentiable, twice continuously differentiable, and strongly convex used in [192], [193]. Further, unlike [192], [193], the $\mathbf{y}$ variable update in the ADMM scheme of **DC-DistADMM** algorithm

in an inexact manner is solved via the ε -consensus protocol that adds additional complexities in the analysis. These reasons have motivated us to present the analysis of the DC-DistADMM given in Theorem 23.

Numerical Simulations:

In this section, three simulation studies are presented for the proposed **DC-DistADMM** algorithm. First, a performance comparison of the **DC-DistADMM** algorithm with two existing algorithms in the literature for solving constrained distributed optimization problems, [158] and [152], in solving a distributed ℓ_1 regularized logistic regression with a local linear equality and set (norm-ball) constraints is presented.

Second, a performance comparison of the **DC-DistADMM** algorithm is provided for solving an unconstrained ℓ_1 regularized Huber loss minimization problem with the existing state-of-the-art unconstrained distributed optimization algorithms on directed graphs. The algorithms used for comparison with the proposed **DC-DistADMM** algorithm are the following: (i) EXTRAPush [139], (ii) PushPull [140], (iii) PushDIGing [194], and (v) the subgradientPush [120] algorithm.

Third, a comparison between the **DC-DistADMM** algorithm and two existing algorithms utilizing multiple communication steps, [178] and [175] on solving the unconstrained distributed least squares problem is presented.

A network of 100 agents connected via: (i) an undirected graph generated using an Erdos-Renyi model [121] with a connectivity probability of 0.3 in simulation studies one and three and (ii) a directed graph generated using an Erdos-Renyi model with a connectivity probability of 0.2 in the simulation study two is considered. The weight matrices for the various algorithms are chosen using the equal neighbor model [51]. To provide a comparison, solution residual plots against the total (computation + communication) iteration counts for all the algorithms unless stated otherwise are presented. Further, a comparison based on the CPU time (the amount of time required by a computer (processor) to execute the instructions) between **DC-DistADMM** and the other algorithms is provided.

All the numerical examples in this section are implemented in MATLAB, and run on a desktop computer with 16 GB RAM and an Intel Core i7 processor running at 1.90 GHz. The parameters used in the simulations for all the algorithms are reported in Table 3.2. In choosing the step-sizes of the algorithms in Table 3.2 we followed the approach of hand-optimizing the hyper-parameters that produce good performance for

Table 3.2: Algorithm Parameters of **DC-DistADMM**

ALGORITHM	Parameter
DC-DistADMM	tolerance $\eta_k = 0.01, 1/k^{2.1}, 0.75^k, \gamma = 10$
CDA1 [158]	step-size $\alpha = 0.01$
CDA2 [152]	step-size $\alpha(k) = 1/k^{1.2}$
EXTRAPush [139]	step-size $\alpha = 0.0009$
PushPull [140]	step-size $\alpha = 0.05$
PushDIGing [194]	step-size $\alpha = 0.001$
subgradientPush [120]	step-size $\alpha = 0.005$
FDGD [175]	step-size $\alpha = 0.01, \beta(k) = k/(k+3)$
nearDGD [178]	step-size $\alpha = 0.01$

these algorithms while maintaining the stability of the algorithmic estimates for the class of problems under consideration.

A. Performance on constrained optimization problem:

Consider a ℓ_1 regularized distributed logistic regression with linear equality constraint and local inequality constraints:

$$\begin{aligned}
& \text{minimize } \sum_{i=1}^n \sum_{j=1}^{n_i} \log(1 + \exp(-y_{ij}(a_{ij}^\top x))) + \theta \|x\|_1 \\
& \text{subject to } H_i x = h_i, x^\top x \leq r_i, \text{ for all } i \in \mathcal{V},
\end{aligned} \tag{3.107}$$

where, each agent $i \in \mathcal{V}$ has n_i data samples $\{(a_{ij}, y_{ij})\}_{j=1}^{n_i}$, with, $a_{ij} \in \mathbb{R}^{50}$ being the feature vector and $y_{ij} \in \{-1, +1\}$ is the binary outcome (or class label) of the feature vector a_{ij} . The objective is to learn the weight vector $x \in \mathbb{R}^{50}$ based on the available data $\{(a_{ij}, y_{ij})\}_{j=1}^{n_i}, i \in \mathcal{V}$ such that x is sparse. The parameter θ enforces the sparsity in x . Further, the solution satisfy the linear equality constraints $H_i x = h_i$, where, $H_i \in \mathbb{R}^{500 \times 50}$. Denote $\mathbf{H} := \mathbf{I} \otimes H_i, \mathbf{h} = [h_1^\top, \dots, h_n^\top]^\top$. Each agent $i \in \mathcal{V}$ also has a local set constraint $x^\top x \leq r_i$. For our simulation we generate an instance of problem (3.107) where each agent has $n_i = 10^6$ data samples. Each feature vector $a_{ij} \in \mathbb{R}^{50}$ and the ‘true’ weight vector $x_{\text{true}} \in \mathbb{R}^{50}$ are generated to have approximately 40% zero entries. The non-zero entries of a_{ij} and x_{true} are sampled independently from the standard normal distribution. The class labels y_{ij} are generated using the equation: $y_{ij} = \text{sign}(a_{ij}^\top x_{\text{true}} + \delta_{ij})$, where δ_{ij} is a normal random variable with zero mean and variance 0.1. The parameter θ is set to $0.1\theta_{\max}$, where $\theta_{\max}$ is the critical value above

which the solution $x^* = 0$ (see [141] section 11.2 for the calculation of $\theta_{\max}$). The value of r_i in the constraints is chosen as follows: the unconstrained version of (3.107) with a centralized solver and denote the solution as x_c^* . The constraint r_i is set as $r_i = (1 + \xi_i)x_c^*$, where ξ_i is drawn from a uniform distribution on $[0, 1]$. A value of $\gamma = 10$ is used for the Augmented Lagrangian. The entries of each H_i are sampled independently from the standard normal distribution. The vector $h_i, i \in \mathcal{V}$ is calculated as $h_i = H_i x_{\text{true}}$. To solve (3.107) using the **DC-DistADMM** algorithm, the fast iterative shrinkage thresholding algorithm (FISTA) [195] is used for the updates (3.58) at each agent i . A proximal residue (difference between the output of the proximal minimization step and the base-point of the proximal term) [195] is utilized as a stopping criterion for FISTA. The FISTA iterations are terminated when the proximal residue becomes less than 10^{-4} . A fixed accuracy for FISTA is used at all iterations of **DC-DistADMM** algorithm. For the updates (3.59) the ε -consensus protocol with the tolerance ε set equal to a desired level of inaccuracy η in the solution of (3.59) is employed. The results for three choices of the sequence $\{\eta_k\}_{k \geq 1}$: (i) $\eta_k = 0.01$, (ii) $\eta_k = 1/k^{2.1}$ and (iii) $\eta_k = 0.75^k$, where, k is the iteration counter are presented for Algorithm 6. Note, that sequences (ii) and (iii) satisfy condition (3.82) and are utilized to derive explicit rate of convergence for the **DC-DistADMM** algorithm in theorems 22, and 23. Two existing algorithms [158] and [152] (termed as Constrained Distributed Algorithm 1 (CDA1) and CDA2 respectively for reference) for solving constrained distributed optimization problems are compared with the **DC-DistADMM** algorithm. Figure 3.6 shows the plots of consensus constraint residual ($\|\hat{\mathbf{x}}^k - \hat{\mathbf{y}}^k\|$ for **DC-DistADMM** algorithm and $\|\mathbf{x}^k - \sum_{i=1}^n x_i^k/n\|$ for CDA1 and CDA2) and the equality constraint residual for the three algorithms with respect to the total iterations. The **DC-DistADMM** performs well as seen in Figure 3.6(a). The consensus residual under **DC-DistADMM** decreases to a value less than 10^{-4} within 200 iterations for the choice $\eta_k = 0.75^k$. Note, that the constant η_k sequence also have good performance with a value less than 10^{-2} within 200 iterations. The plots in Figure 3.6(b) demonstrate the decrease in the equality constraint residual $\|\mathbf{H}\hat{\mathbf{x}}^k - \mathbf{h}\|$. Figure 3.7 presents the trajectory of solution residuals $\frac{\|\mathbf{x}^k - \mathbf{x}^*\|^2}{\|\mathbf{x}^0 - \mathbf{x}^*\|^2}$ with respect to the total iterations and the CPU time (secs) for each algorithm. The proposed **DC-DistADMM** algorithm outperforms the other two algorithms. The solution residual for **DC-DistADMM** is lesser than CDA1 and CDA2 for all the three choices of the tolerance sequence η_k with significant improvement with the choice $\eta_k = 0.75^k$. Note that both CDA1 and CDA2 cannot handle directed communication topologies and need centralized synthesis for problem (3.107) whereas **DC-DistADMM** does not.

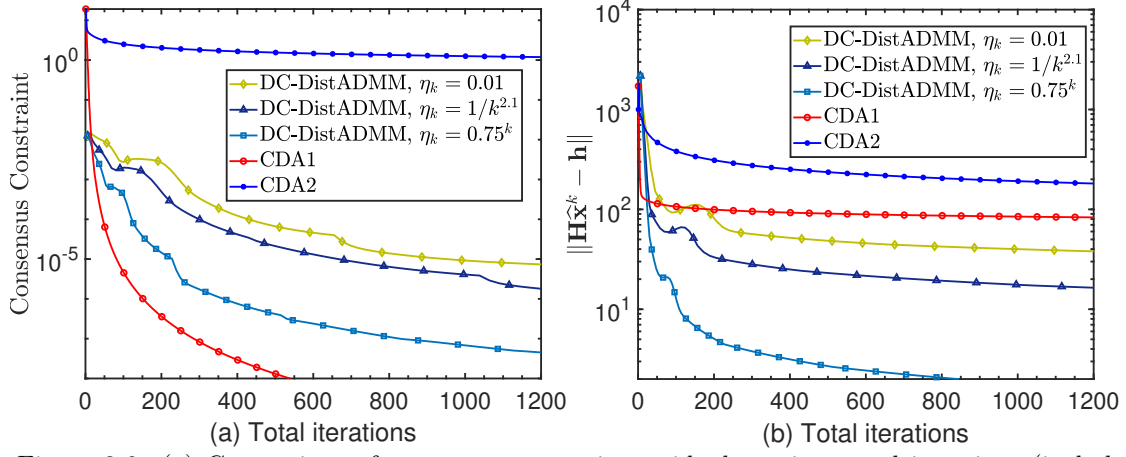


Figure 3.6: (a) Comparison of consensus constraint residuals against total iterations (including communication steps) between **DC-DistADMM** and constrained optimization algorithms (b) Comparison of equality constraint residuals against total iterations between **DC-DistADMM** and constrained optimization algorithms. $\eta^k = 0.75^k$ gives the best performance for the **DC-DistADMM** algorithm in terms of the equality constraint residual. The **DC-DistADMM** algorithm is second best in terms of the consensus constraints.

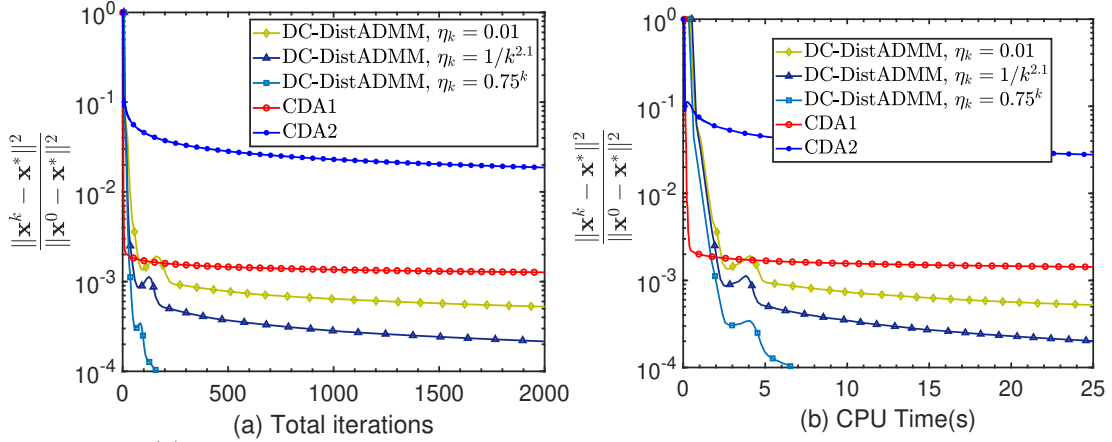


Figure 3.7: (a) Comparison of solution residuals against total iterations between **DC-DistADMM** and constrained optimization algorithms (b) Comparison of solution residuals against CPU time between **DC-DistADMM** and constrained optimization algorithms. **DC-DistADMM** algorithm with $\eta^k = 0.75^k$ has the best decrease in the solution residual.

B. Performance on unconstrained optimization problem:

In the second part of the numerical simulation study a comparison of the proposed **DC-DistADMM** algorithm is presented with some state-of-the-art distributed optimization algorithms for solving unconstrained optimization problems. Consider the following ℓ_1

regularized Huber loss minimization problem:

$$\text{minimize } \sum_{i=1}^n \Phi_i(x) + \theta \|x\|_1, \quad (3.108)$$

where, $\Phi_i(x)$ for all i is the standard Huber loss given by:

$$\Phi_i(x) = \begin{cases} \frac{1}{2} \|D_i x - d_i\|^2, & \text{if } \|D_i x - d_i\| \leq 1, \\ \|D_i x - d_i\| - \frac{1}{2}, & \text{if } \|D_i x - d_i\| > 1. \end{cases}$$

Each agent $i \in \mathcal{V}$ has a measured data vector $d_i \in \mathbb{R}^{100}$ and a scaling matrix $D_i \in \mathbb{R}^{100 \times 25}$. The objective is to estimate the unknown signal $x \in \mathbb{R}^{25}$. The entries of the matrices $D_i, i \in \mathcal{V}$ and the observed data d_i are sampled independently from the standard normal distribution. The true solution vector x^* has 70% non-zero entries which are sampled from a standard normal distribution. A value of $\theta = 3$ is chosen for the regularization parameter. Here, FISTA is employed to solve the sub-problem (3.58) with the same stopping criterion (proximal residue $< 10^{-4}$) as in the logistic regression problem. The update (3.59) is solved using ε -consensus protocol with the tolerance $\eta_k = 1/k^{2.1}$. The progression of the solution residual against the total iterations and the CPU time is presented in Figure 3.8.

In Figure 3.8(a) the progression of the solution residual against the algorithm iteration k (first curve) for the **DC-DistADMM** algorithm is also provided. The rationale for this is to provide a comparison between **DC-DistADMM** and other algorithms considering both the algorithm iterations and the total iterations. It can be seen that the proposed **DC-DistADMM** algorithm has a significantly better performance compared to other algorithms against the algorithm iterations. In particular, solution residual decreases to a value of 10^{-4} in less than 50 algorithm iterations of **DC-DistADMM** algorithm whereas the second best method PushDIGing takes around 100 iterations to reach the same solution residual. In terms of total iterations, the **DC-DistADMM** algorithm has acceptable performance. The ε -consensus protocol utilized at each iteration of the **DC-DistADMM** algorithm incurs additional consensus-related steps, however, the information mixing steps (3.64)-(3.65) have a low computational footprint and do not significantly increase the overall run time of the **DC-DistADMM** algorithm. This is illustrated by the comparison between the algorithms based on the required CPU time shown in Figure 3.8(b). The residual plots illustrate that the **DC-DistADMM** algorithm performs better in terms

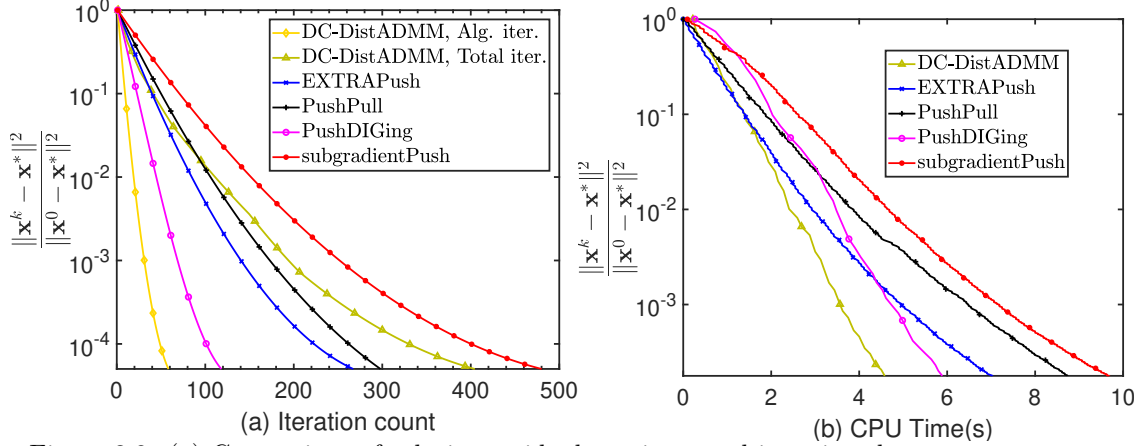


Figure 3.8: (a) Comparison of solution residuals against total iterations between **DC-DistADMM** and unconstrained optimization algorithms (b) Comparison of solution residuals against CPU time between **DC-DistADMM** and unconstrained optimization algorithms. **DC-DistADMM** algorithm performs the best in terms of the CPU time while having acceptable performance with respect to total iterations. **DC-DistADMM** has the best decrease in solution residual with respect to algorithm iterations.

of the CPU time requirement compared to the other methods to reach the same level of residual value.

C. Comparison with algorithms utilizing multiple communication steps:

Here, a comparison with two unconstrained distributed optimization algorithms that utilize multiple communication steps, nearDGD [178] and the algorithm in [175] (referred here as Fast-DGD (FDGD)) is provided; the reader is directed to [175], [178] for the motivations for multiple consensus steps. We consider an unconstrained distributed least squares problem,

$$\text{minimize } \frac{1}{2} \sum_{i=1}^n \|D_i x - d_i\|^2. \quad (3.109)$$

Here, each agent $i \in \mathcal{V}$ has a measured data vector $d_i \in \mathbb{R}^{100}$ and a scaling matrix $D_i \in \mathbb{R}^{100 \times 25}$. The objective is to estimate the unknown signal $x \in \mathbb{R}^{25}$. The entries of the matrices $D_i, i \in \mathcal{V}$ and the observed data d_i are sampled independently from a standard normal distribution $\mathcal{N}(0, 1)$. The true signal x^* also have entries that are sampled from an i.i.d standard normal distribution. In this case, the sub-problem (3.58) takes a closed form solution which can be readily computed by the first order optimality condition of

the unconstrained problem. The solution residuals obtained while solving (3.109) using

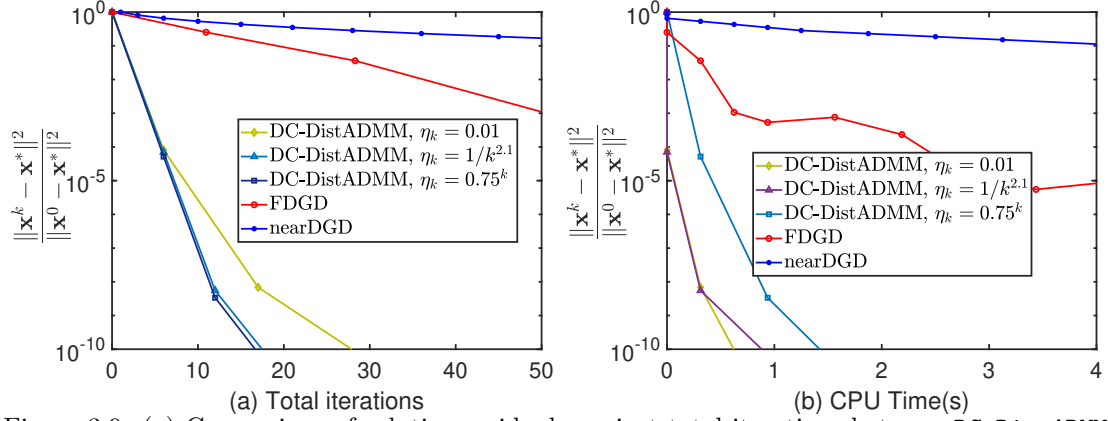


Figure 3.9: (a) Comparison of solution residuals against total iterations between **DC-DistADMM** and algorithms utilizing multiple communication steps (b) Comparison of solution residuals against CPU time between **DC-DistADMM** and algorithms utilizing multiple communication steps. $\eta_k = 0.75^k$ has the best decrease in solution residual with respect to total iterations. **DC-DistADMM** with a constant tolerance performs the best with respect to CPU time.

the three algorithms are presented with respect to the total iterations and CPU time in Figure 3.9. It can be seen that the **DC-DistADMM** algorithm has the fastest decrease in the solution residual. Figure 3.10 presents the number of communication iterations utilized

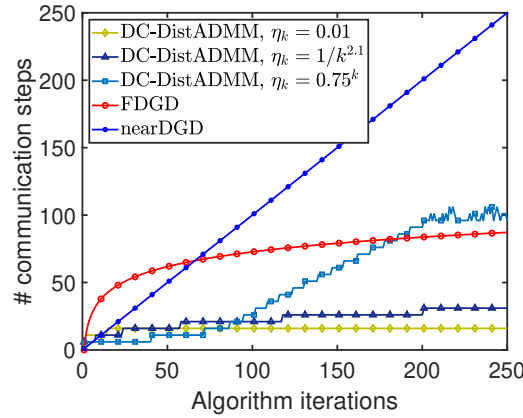


Figure 3.10: Comparison of the number of communication iterations performed by **DC-DistADMM** algorithm for the three choices of η_k and the FDGD and nearDGD algorithms. **DC-DistADMM** with $\eta_k = 0.01$ and $\eta_k = 1/k^{2.1}$ has lesser communication cost compared to the FDGD and nearDGD algorithms while the FDGD has better communication complexity than **DC-DistADMM** with $\eta_k = 0.75^k$ as the number of iterations increases.

by the **DC-DistADMM** algorithm while solving (3.109) for the three error sequences and the

FDGD and nearDGD algorithms. The trend suggested by Lemma 18 can be seen in the plots in Figure 3.10, where the number of communication iterations for the **DC-DistADMM** algorithm increases as we tighten the accuracy. Note, the sequence $\eta_k = 1/k^{2.1}$ with convergence rate guarantees provided by Lemma 22 results in a logarithmic increase in the number of communication iterations and is a suitable choice for getting good performance. The **DC-DistADMM** algorithm with tolerance $\eta_k = 0.01$ and $\eta_k = 1/k^{2.1}$ has lesser communication cost compared to the FDGD and nearDGD algorithms while the FDGD has better communication complexity than the **DC-DistADMM** algorithm with the tolerance $\eta_k = 0.75^k$ as the number of iterations increases.

Summarizing, the simulation results indicate that with respect to the state-of-the-art *constrained* optimization problems **DC-DistADMM** provides better performance while encompassing a large class of distributed optimization scenarios with linear equality, inequality, and set constraints. Even with respect to the state-of-the-art *unconstrained* optimization frameworks, the **DC-DistADMM** algorithm provides better performance with respect to computations (CPU time) and remains acceptable with respect to total communication steps.

Chapter 4

Methods for Optimal and Resilient Operation of Power and Energy Systems

4.1 Gist of the Chapter

This chapter addresses three critical challenges in microgrid (MG) and distributed energy resource (DER) management. First, it introduces a novel distributed secondary control scheme for reactive power sharing and voltage regulation among grid-forming inverter-based resources (GFM IBRs). Unlike existing approaches, the proposed scheme ensures plug-and-play operation, protects private information, and reduces communication requirements, enhancing scalability and privacy. Second, it tackles the resilience issue in DER coordination by developing a distributed intruder detection mechanism. This scheme enables honest DERs to identify and isolate malicious agents, ensuring reliable and secure operation. Lastly, the chapter presents the Horizon of Viability (HoV) engine, an optimization framework for scheduling local generation resources and deferrable loads during grid outages. The HoV engine maximizes the viability of critical infrastructure (CI) units while maintaining service quality for other loads. These contributions are validated through controller hardware-in-the-loop (CHIL) simulations, demonstrating practical and robust solutions to improve MG and DER control. The research work presented in this chapter is published in the articles [196]–[198].

4.2 State-of-the-art on Power and Energy System Operation and Control

AC microgrids (MGs) consist of loads and inverter-based resources (IBRs) interfacing with renewable energy resources and energy storage via power electronics. Based on the nature and the expected functionality, these IBRs usually operate in a grid-following (GFL) or grid-forming (GFM) mode. IBRs operating in the GFM mode should generate and maintain stable voltages and frequency of the MG when disengaged from the main grid. MGs typically follow a hierarchical control structure by adapting ANSI/ISA-95-based international standard [199]. The hierarchical control structure consists of three levels: primary, secondary, and tertiary control. The concept of droop control is utilized extensively at primary control for active and reactive power sharing [200]. However, for MG operation with GFM IBRs with only primary droop control, the voltage of the system deviates from its nominal values, and reactive power demand is not shared as desired. One of the main challenges at the secondary control level is the coordination of the GFM IBRs for restoring system voltages to the nominal value and sharing reactive power appropriately.

Traditional centralized secondary control systems [199], [201]–[203] require global information aggregation and dissemination and don't scale well. Researchers have explored distributed secondary control schemes to reduce communication bandwidth requirements. Most distributed secondary control schemes include modification of the droop control with an additive corrective term (see [173], [204]–[214] and the references therein). The distributed voltage controllers proposed in [204]–[206] require each GFM IBR to collect measurements of every GFM IBR in the network to produce the appropriate local control signal; leading to a huge communication bandwidth for every GFM IBR. In articles [207], [208] communication networks with a directed spanning tree are considered. Further, to generate voltage control set-points, a leader-follower synchronization scheme is utilized that is susceptible to a single point of failure. Articles [173], [209] developed distributed proportional and integral (PI) controllers for frequency and voltage regulation via averaging dynamics based on a balanced positive definite Laplacian matrix. Articles [210], [211] propose a centrally synthesized non-linear non-smooth distributed control signal based on a leader-follower synchronization and suffers from the same limitations as [207], [209]. The work in article [212] considers discrete-time updates with non-uniform communication intervals for a virtual

leader-follower distributed PI controller. Article [213] utilizes an event-triggered communication approach to reduce the utilization of the communication channels. Article [214] presents an optimization problem for the design of a sparsity-promoting Laplacian matrix utilized in the centrally synthesized controllers schemes in [173], [209]. Manipulating the virtual impedance (VI) in the control loop provides another approach for controlling reactive power sharing among GFM IBRs (see [215]–[217] and references therein). In [215], the VI parameters are tuned via a genetic algorithm, introduced to minimize the system’s global reactive power sharing error. Article [216] utilizes a VI proportional to the reactive power. The VI design in [215], [216] is performed offline in a centralized MG configuration stage. The scheme in article [217] considers an adaptive VI designed via a balanced Laplacian matrix.

With the proliferation of power electronic converter technologies, renewable energy resources, and battery storage systems, the modern microgrid network is transitioning towards integrating a large number of smaller distributed energy resources (DERs) scattered throughout the network. Further, with the presence of the DERs the dispatch methodologies for deploying ancillary services for reliable operation of the modern grid are changing significantly (for example, the Electric Reliability Council of Texas’s (ERCOT’s) responsive reserve services (RRS)), thereby demonstrating the viability of widescale adoption of DERs for such applications [218], [219]. To achieve grid ancillary services a large number of DERs must be coordinated on a fast timescale. The coordination of multiple DERs presents significant challenges. Several articles (see [220], [221] and references therein) in the literature have introduced centralized approaches wherein a centralized controller at the distribution level sends dispatch commands to local actuators and requires information from all of the DERs. Such centralized approaches lack flexibility and scalability, and they require expensive high-performance computing and high-speed communication networks to meet the ancillary service requirements satisfactorily. To mitigate these challenges, distributed control approaches are proposed with DERs as a multi-agent system [222]–[224]. Advantages of distributed approaches include coordination using only local computations and plug-and-play capability. Article [225] presents a gather-broadcast method, a distributed average integral method was developed in [226]. Article [227] develops a distributed approach but does not provide any theoretical guarantees and relies on heuristics to achieve dispatch requests. A distributed consensus-based power apportioning algorithm overcoming the limitations of the above methods is developed in [228], [229]. We focus on the distributed power apportioning algorithms proposed in articles [228], [229].

A key component of the distributed apportioning schemes in [228], [229] is achieving consensus between the decision variables of different DERs. The commonly employed consensus algorithms in the literature [228]–[231] assume an underlying trust and credibility of the participating agents. When factoring in the distributed coordination for ancillary service dispatch, any DER’s lack of honesty can skew the resulting decision, which can adversely affect the power network. Thus, it is imperative to employ resilient measures to counter the presence of malicious agents in the network. Several works in the literature have studied the problem of securing the consensus updates [232]–[236]. A trust-based model for multi-agent systems is considered in [232]. Article [233] devises an intrusion detection system based on network traffic analysis. In contrast, authors in [234], [235] assert designing the interaction topology between the agents to achieve an agreement. Article [236] develops a scheme based on the monotonic properties of global maximum and minimum states of the agents under the consensus algorithm to detect intrusions.

The rapid recovery of power flow, possibly after a blackout, is a crucial need arising in scenarios that are increasingly frequent [237]–[239]. The disruption of power to critical infrastructure (CI) units—such as hospitals, medical centers, data centers, etc.—often results in a debilitating impact on physical and economic security, public health and safety. Solutions for the sustained viability of power to CI units in a microgrid, while the grid is being restored, are urgently needed to keep CI units online. Increasingly, the *sustenance* of reliable power requires assistive services to the CI units for long periods of time. Accredited standards like IEEE Std 2030.7 recommend a “dispatch rule”-based optimal energy management system due to its simplicity and low cost in implementation in a microgrid [240]; however, it is primarily concerned with the grid-tie mode of microgrid operation for energy management. Even though the need for energy management during islanded mode is urgent, there has been only sparse effort toward a strategy for ensuring the sustained viability of power to the CI units in a microgrid in the aftermath of the increasingly frequent catastrophic events.

There are some works on the “rule-based” optimal dispatch control of energy resources for microgrids [241]–[245]. In [241], [242], an energy management control scheme with the energy storage system used for power smoothing and state-of-charge (SOC) controlling is proposed for a microgrid with a fixed load; however, critical and deferrable load management of the microgrid, which is crucial when loads are prioritized based on criticality in the system, is not included in the energy management scheme. The works [243]–[245] use a conventional battery cycle-charging strategy which has

the disadvantage of shortening battery lifetime through frequent and deep discharges. Moreover, these schemes are primarily focused on generation resource allocation against a fixed and non-vulnerable load in a microgrid. A variety of works in the literature have dealt with the scheduling of deferrable loads in microgrids [246]–[252]. The works [246]–[249] model the deferrable loads as tasks having fixed start and end times with a given energy requirement. This type of deferrable load modeling is not best suited in a constrained scenario when the total generation is less than the total critical and deferrable load demands. In particular, due to the strict start and end times of the deferrable loads, there is less flexibility to switch off certain loads to maintain the supply and demand power balance at all times of operation. This may lead to no load and generation dispatch, reducing the overall quality of service to the critical and deferrable loads in the CI unit. The works [250]–[252] proposed coordinated cost optimal dispatch strategies for deferrable loads. [250] solves a binary programming problem to find a deferrable load action to take for every given frequency deviation with hourly updates. [252] uses dynamic programming approach to find the deferrable load switch-on-off actions for fast frequency response to support the grid. The articles [246]–[252] do not consider optimal scheduling of the available generation sources. The reference [253] compares eight optimization-based coordination frameworks present in the literature that deals only with load scheduling. The reference [254] developed distributed approach for coordinating load aggregators that integrates with automatic generation control for secondary frequency response. The authors in [255] proposed a centralized solution for load control with pre-coordinated day-ahead estimated parameters. The article [256] demonstrated similar combined load control capabilities wherein resources grid tied resources run with static droop characteristics and a direct load frequency control approach is utilized to generate load actions. The authors in [257] proposed a scheme for the dispatch of distributed energy resources, including renewable generation and storage reserves; however, similar to [246]–[249], [257] considers the deferrable loads as tasks. Unlike [246]–[252], [254]–[257], where the presence of a grid is required and the power available from the grid is used to dispatch loads under normal operating conditions.

4.3 Research Questions

Research Focus 1:

Although existing secondary control schemes in the literature can be implemented in a distributed manner, they often rely on centralized synthesis based on global network information, rendering them unsuitable for plug-and-play operation. In this chapter, we propose a novel secondary control scheme designed for reactive power sharing and voltage regulation among heterogeneous grid-forming inverter-based resources (GFM IBRs). Our developed scheme ensures that the private information of each GFM IBR remains confidential and is not exposed to the entire network.

This chapter addresses several key questions:

1. How can reactive power sharing and voltage regulation be achieved without relying on centralized synthesis?
2. What strategies can ensure plug-and-play capability while maintaining system stability and performance?
3. How can private information of individual GFM IBRs be protected while enabling effective distributed coordination?

By exploring these questions, our scheme enhances system scalability and flexibility, supports dynamic network configurations without requiring re-synthesis, and provides a practical foundation for future advancements in distributed control strategies for GFM IBRs.

Research Focus 2:

Existing distributed schemes for aggregating distributed energy resources (DERs) to provide ancillary services lack mechanisms for resilient operation in the presence of malicious DERs. These methods often exhibit sensitivity to gain coefficients, leading to slow convergence toward dispatch outputs. Additionally, they require repeated interactions between agents and the collection of large amounts of information from across the network. This complexity makes it challenging to generalize these schemes, particularly when formulating notions of cooperation or rejecting unreliable information. Critically, existing methods do not effectively isolate malicious agents.

To address these challenges, we propose a novel distributed intruder detection scheme. This approach empowers honest DERs to detect and isolate maliciously behaving DERs

within the distributed apportioning algorithm. By enhancing robustness and efficiency, our scheme provides a foundation for secure and reliable DER aggregation in distributed networks.

Research Focus 3:

We focus on a grid interruption event when a large portion of grid power is disrupted and the microgrid containing the CI unit is working in an islanded mode while being supported by its own available backup sources and surrounding community buildings (CBs). The objective of the microgrid is to sustain the CLs in the microgrid and Deferrable Loads (DLs) in the CI unit by utilizing *only* the available backup generation-such as the stand-alone diesel generators, battery inverter systems, and the locally available uncertain renewable power generation-until the grid support comes back. A trivial solution for the microgrid is to shut down all the DLs of the CB units and utilize the available generation to only support the CI unit and the CLs of the CB units; however, this might lead to a reduced quality of service in the microgrid due to *no* DLs in the CB units being served, leading to discomfort for the consumers in the microgrid. Hence, there is a need to devise an optimal scheduling policy for the net-load resources in the microgrid. Toward this objective, in this chapter, we focus on developing an optimization algorithm, called the HoV engine, which determines a portfolio of generation and load to guarantee the viability of all the CLs in the microgrid while maximizing the level of service of the DLs in the microgrid over a given time horizon. The developed scheme emphasizes keeping the CI unit online with the objective of maximizing the time horizon of power viability by optimally scheduling the locally available generation resources and deferrable loads at and surrounding the CI unit in the event of a disaster.

4.4 Contributions of the Chapter

The current chapter makes the following contributions toward the three research focuses outlined above:

4.4.1 Contributions Towards Research Focus 1:

1. A distributed secondary control strategy for voltage regulation and reactive power sharing is proposed based on a distributed optimization framework. Compared

to the existing works in the literature the proposed scheme allows for a fully distributed synthesis and a plug-and-play operation.

2. The proposed distributed secondary control scheme provides reactive power sharing and voltage regulation among heterogeneous distributed GFM IBRs connected through varying line impedances.
3. We utilize an online algorithm using only local measurements to lower the communication bandwidth requirements. Additionally, only non-linear estimates are shared between the GFM IBRs during updates, safeguarding private information such as real-time power measurement, maximum capacity of generations, and local load demands, which is beneficial to enhance the protection of information privacy against malicious cyber-attacks.

We evaluate the proposed distributed secondary control scheme using a controller hardware-in-the-loop (CHIL)-based real-time simulation. The laboratory test results establish the efficacy of the proposed scheme for secondary control in MGs.

4.4.2 Contributions Towards Research Focus 2:

1. We develop a distributed intruder detection scheme for distributed power apportioning with DERs in microgrids.
2. The developed scheme allows the DERs to locally detect and isolate the communication links of the maliciously behaving DERs in their neighborhood.
3. Compared to the existing works in the literature the proposed scheme leads to distributed detection and isolation of malicious DERs and is applicable for general interaction topologies.
4. The developed scheme has a low computation and communication footprint unlike existing schemes in the literature [234], [235] that require a polynomial number of computations and communication in the number of DERs. The proposed algorithm works only in the neighborhood of a DER and thus, scales well due to local interactions.

4.4.3 Contributions Towards Research Focus 3:

The contributions of this article are as follows:

1. We develop a net-load (generation and load) dispatch scheme, termed horizon of viability (HoV) engine, to optimally schedule the locally available generation sources and the deferrable loads to provide guaranteed viability of power to the CI unit over a time horizon in the event of a catastrophe leading to the disruption of power from the main grid.
2. We evaluate the HoV engine using controller-hardware-in-the-loop (CHIL) based real-time simulation. These laboratory-based testing results establish the efficacy of the proposed HoV engine in guaranteeing the power viability to CI units in an islanded scenario.
3. We compare the performance of the proposed HoV engine with an existing net-load management [252] system to demonstrate the advantage of the proposed scheme over the existing algorithms in the literature.

4.5 Developed Solutions

4.5.1 Distributed Secondary Control for Microgrids with Grid-Forming Inverters

Brief Description of Secondary Control:

In this section, secondary control is discussed from the scope of reactive power sharing control among multiple GFM IBRs in an islanded MG. Figure 4.1(a) shows a generic MG consisting of N number of GFM IBRs connected to a common bus, point-of-common-coupling (PCC), via various lines. A complete description of the control typologies of the GFM IBRs is out of scope; a brief description is provided. Each GFM IBR is controlled via voltage controller-based *inner control loop* where the reference voltage signal is generated using the $P \sim f / Q \sim V$ droop control-based *primary control*. For inductive lines, the *primary controller* without the *secondary control* specify the i^{th} GFM IBR's frequency, ω_i , and voltage magnitude, V_i by:

$$\omega_i = \omega^* - n_i P_i, \quad V_i = V^* - m_i Q_i, \quad (4.1)$$

where ω^* , V^* are the nominal frequency (in rad/s) and voltage set-point (in volt) of the MG respectively, and the gains n_i , m_i are the droop coefficients. P_i and Q_i are the measured average active and reactive power of the i^{th} GFM IBR, respectively. It is

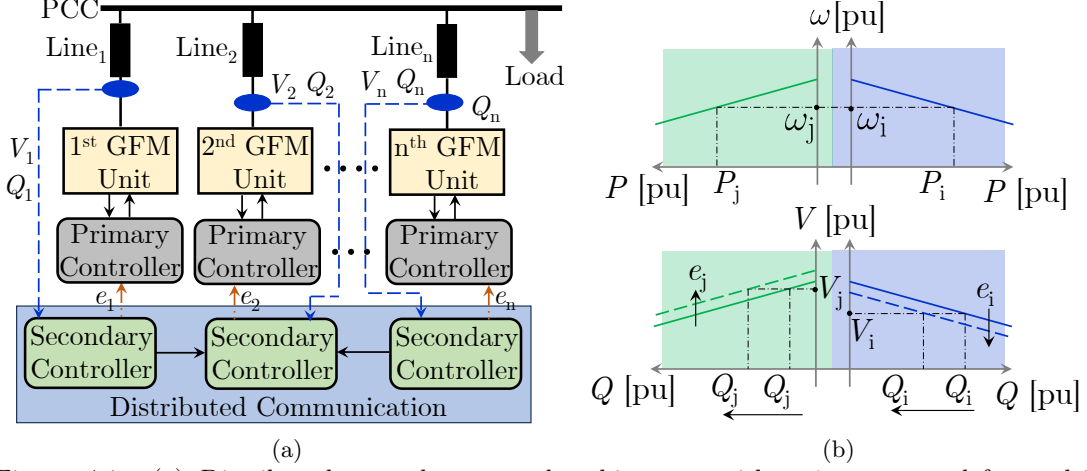


Figure 4.1: (a) Distributed *secondary control* architecture with *primary control* for multiple GFM IBRs connected in parallel, (b) $P \sim f$, $Q \sim V$ droop-based *primary control* with *secondary control* for parallel GFM IBRs i and j with equal ratings, connected to PCC via reactive lines with $X_j > X_i$.

generated by processing the instantaneous active power, p_i , and reactive power, q_i , via a low-pass filters with the time constant, $\tau_{S,i} \in \mathbb{R}_{>0}$. As a result, $P_i := [1/(\tau_{S,i}s + 1)]p_i$ and $Q_i := [1/(\tau_{S,i}s + 1)]q_i$. The large-signal stability analysis of the MG system yields the steady-state network frequency as,

$$\omega_{ss} = \omega^* - \left(\frac{1}{\sum_{i=1}^n (1/n_i)} \right) P_L, \quad (4.2)$$

where P_L is the total active power load in the MG. The steady-state frequency, ω_{ss} , being a global quantity of the MG system, is different from the nominal ω^* , but equal in all the GFM IBRs. Thus, using (4.1) and (4.2), the following can be stated as,

$$n_1 P_1 = n_2 P_2 = \dots = n_i P_i = \dots = n_n P_n. \quad (4.3)$$

Therefore, the total active power demand, P_L , is shared according to the selection of the $P \sim f$ droop coefficients. In other words, the per-unitized (pu) active power with each GFM IBR's base rating, is equal. This phenomenon can be understood using the top figure of Figure 4.1(b). However, the same analysis for the $Q \sim V$ droop controller is not as straightforward. It is well studied that due to the unequal line impedances, the $Q \sim V$ droop controller is unable to share reactive power demand among even identical

GFM IBRs operating in parallel and as a result,

$$m_1 Q_1 \neq m_2 Q_2 \neq \dots \neq m_i Q_i \neq \dots \neq m_n Q_n. \quad (4.4)$$

For simplicity of exposition, a case study of two GFM IBRs with identical ratings (i^{th} and j^{th} GFM unit), but operating through reactive lines with $X_j > X_i$ in parallel is considered. The bottom figure of Figure 4.1(b) depicts the $Q \sim V$ droop law before the *secondary control* action (solid lines of bottom figure of Figure 4.1(b)). The GFM IBRs operate at voltages V_i and V_j with reactive power injections Q_i and Q_j . Since, $X_j > X_i$, the terminal voltages of the GFM IBRs follows $V_i < V_j$ and as a result, reactive power flow follows $Q_i > Q_j$. This is the limitation of employing only *primary control* without the *secondary control* compensation in parallel operation of multiple GFM IBRs. Moreover due to non-zero values of Q_i 's and voltage drop across the line impedances, the voltage at PCC becomes less than nominal, V^* . Therefore, to ensure reactive power sharing among multiple GFM IBRs connected to the PCC via asymmetrical line impedances and desired voltage regulation at PCC, a dynamic compensation of $Q \sim V$ droop law is required as discussed in the next section.

Distributed Reactive Power Sharing Scheme:

A. Control/Update Rule Design:

Consider a MG with N GFM IBRs. Let the voltage and the reactive power at the i^{th} GFM IBR at any time be denoted as $V_i(t)$ and $Q_i(t)$ respectively. A droop law is employed for updating the voltage at the i^{th} GFM IBR any time $t + 1$,

$$V_i(t + 1) = V^* - m_i Q_i(t) + v_i(t), \quad (4.5)$$

where V^* is the voltage reference, m_i is the droop coefficient, and $v_i(t)$ is the adjustment given at time t . The adjustment $v_i(t)$ should either maintain equal-rated reactive power sharing between the GFM IBRs or tight regulation of the voltage at the GFM IBRs depending on the system requirement. We follow an optimization-oriented approach to design the adjustment $v_i(t)$. Let the directed graph $\mathcal{G}$ represent the communication network of the GFM IBRs ($\mathcal{G}$ isn't necessarily the same as the power network connections). Let $\mathcal{G}(\{1, 2, \dots, N\}, \mathcal{E})$ denote a graph with the set of edges $\mathcal{E} \subset \{1, 2, \dots, N\} \times \{1, 2, \dots, N\}$. Let $\mathcal{N}_i^- := \{j \mid (i, j) \in \mathcal{E}\}$ and $\mathcal{N}_i^+ := \{j \mid (j, i) \in \mathcal{E}\}$ respectively denote the set of in-neighbors and out-neighbors of node i in $\mathcal{G}$. Consider

the optimization problem at time t ,

$$\underset{x_1, x_2, \dots, x_N}{\text{minimize}} \sum_{i=1}^N \frac{1}{2} (x_i - \alpha_i(t))^2 + \frac{\gamma}{2} x_i^2 \quad (4.6)$$

$$\text{subject to } \alpha_i(t) := a_{V_i}(V^* - V_i(t)) + a_{Q_i} m_i Q_i(t) \text{ for all } i \in \{1, \dots, N\}, \quad (4.7)$$

$$x_i = x_j \text{ for all } i, j \in \{1, 2, \dots, N\},$$

where a_{V_i} and a_{Q_i} are free parameters leading to different objective functions and $\gamma > 0$ is a regularization parameter. Let $x_t^* = x_1^* = \dots = x_N^*$ denote the solution to the optimization problem (4.6). Then the adjustment $v_i(t)$ at time t , is given by the following control law, for all $i \in \{1, 2, \dots, N\}$,

$$v_i(t) := x_t^* + \beta_{Q_i} m_i Q_i(t) - \beta_{V_i}(V^* - V_i(t)), \quad (4.8)$$

where β_{Q_i} and β_{V_i} are the free parameters that lead to different steady-state characteristics of the system as discussed later.

Lemma 20. *Let graph $\mathcal{G}$ be strongly connected (i.e. $\mathcal{G}$ has a directed path between every pair of distinct nodes). Then, the solution x_t^* for problem (4.6) is $x_t^* := \frac{1}{N(1+\gamma)} \sum_{i=1}^N \alpha_i(t)$.*

Proof. Let $x_i = x_j = x, \forall i, j$. The separable form of the objective leads to an equivalent problem, with $\alpha_i(t)$ as in (4.7),

$$\underset{x}{\text{minimize}} \sum_{i=1}^N \frac{1}{2} (x - \alpha_i(t))^2 + \frac{\gamma}{2} x^2. \quad (4.9)$$

As problem (4.9) is unconstrained we can differentiate the objective and set it to zero to determine the optimal solution. In particular, $\sum_{i=1}^N [(1 + \gamma)x_t^* - \alpha_i(t)] = 0$ which implies. $x_t^* = \frac{1}{N(1+\gamma)} \sum_{i=1}^N \alpha_i(t)$. Therefore, $x_i = x_j = x_t^*$. $\square$

Remark 26. *Note that in problem (4.6) and the update (4.8) any GFM IBR i only requires local measurements. Further, the GFM IBRs can choose the free parameters based on their local requirements without centralized constraints.*

Delaying the discussion on how problem (4.6) is solved, we first present the steady state analysis for the voltage and reactive power dynamics of the GFM IBRs. Assuming the overall system consisting of the loads and the IBRs has a steady state, let t_{ss} denote

the time after which the system achieves a steady state. Therefore, for any time $t > t_{ss}$, $V_i(t) = V_i(t_{ss})$, $Q_i(t) = Q_i(t_{ss})$. Then from (4.5) and (4.8),

$$\begin{aligned} V_i(t_{ss}) &= V^* - m_i Q_i(t_{ss}) + v_i(t_{ss}) \\ &= V^* - m_i Q_i(t_{ss}) + x_{t_{ss}}^* + \beta_{Q_i} m_i Q_i(t_{ss}) - \beta_{V_i} (V^* - V_i(t_{ss})) \\ &= (1 - \beta_{V_i}) V^* - (1 - \beta_{Q_i}) m_i Q_i(t_{ss}) + x_{t_{ss}}^* + \beta_{V_i} V_i(t_{ss}). \end{aligned}$$

Thus,

$$\begin{aligned} &(1 - \beta_{V_i}) V_i(t_{ss}) + (1 - \beta_{Q_i}) m_i Q_i(t_{ss}) \\ &= (1 - \beta_{V_i}) V^* + x_{t_{ss}}^* \\ &= (1 - \beta_{V_i}) V^* + \frac{1}{N(1 + \gamma)} \left[\sum_{l=1}^N a_{V_l} (V^* - V_l(t_{ss})) + \sum_{l=1}^N a_{Q_l} m_l Q_l(t_{ss}) \right]. \end{aligned} \quad (4.10)$$

Note that different choices of the free design parameters achieve different objectives in the steady state. In particular,

- (i) let $\beta_{V_i} = \beta_V = 1, \beta_{Q_i} = \beta_Q = 0$, for all $i \in \{1, 2, \dots, N\}$. Then from (4.10), $m_i Q_i(t_{ss}) = m_j Q_j(t_{ss})$ for all $i, j \in \{1, 2, \dots, N\}$. Thus, equal-rated reactive power sharing is achieved.
- (ii) let $\beta_{V_i} = \beta_V = 0, \beta_{Q_i} = \beta_Q = 1, a_{V_i} = a_{Q_i} = 0$, for all $i \in \{1, 2, \dots, N\}$. Then from (4.10), $V_i(t_{ss}) = V^*$ for all $i \in \{1, 2, \dots, N\}$. Thus, voltage regulation to V^* is achieved.
- (iii) a combination of the above cases can also be achieved, i.e. let $\beta_{V_i} = \beta_V = 1, \beta_{Q_i} = \beta_Q = 0$ for certain buses towards the objective of the equal-rated power reactive sharing. In contrast, other buses have $\beta_{V_i} = \beta_V = 0, \beta_{Q_i} = \beta_Q = 1, a_{V_i} = a_{Q_i} = 0$ to maintain voltage regulation.

Remark 27. Problem (4.6) and update (4.8) provide a versatile framework to allow the individual GFM IBRs to strike diverse trade-offs wherein, for example, a set of GFM IBRs might want to emphasize voltage regulation while others emphasize how reactive power is shared. This is the first control scheme with such a capability. We demonstrate this powerful ability of our framework via CHIL experiments in Section 4.5.1.

Next, we discuss how the solution to problem (4.6) is obtained. We first review the

GradConsensus method developed in [258] and then utilize it to present the algorithm to generate the update law (4.8) at any time t .

GradConsensus Method:

The *GradConsensus* method developed in [258] solves a static distributed optimization problem of the form

$$\begin{aligned} & \underset{x_1, \dots, x_N}{\text{minimize}} \quad \sum_{i=1}^N f_i(x_i) \\ & \text{subject to} \quad x_i = x_j, \text{ for all } i, j \in \{1, 2, \dots, N\}, \end{aligned} \quad (4.11)$$

where x_i is an estimate maintained locally at agent i . The local estimates of node i at any iteration k are updated as,

$$x_i^+ := x_i(k-1) - \rho \nabla f_i(x_i(k-1)), \quad (4.12)$$

$$x_i(k) = \text{Consensus}_{\varepsilon(k-1)}(x_i^+), \quad (4.13)$$

where $\nabla f_i(x_i(k-1))$ is the gradient of the function f_i evaluated at $x_i(k-1)$ and $\text{Consensus}_{\varepsilon}(y^0)$ is a distributed approximate consensus protocol initialized with the value $y^0 \in \mathbb{R}^N$ and a precision tolerance ε . Next, we describe the $\text{Consensus}_{\varepsilon}$ protocol. Under the $\text{Consensus}_{\varepsilon}$ protocol every node i maintains estimates y_i, z_i, r_i, M_i, m_i . The estimates y_i, z_i, r_i have the following local distributed updates

$$y_i(k+1) = \frac{1}{\mathcal{N}_i^+ + 1} y_i(k) + \sum_{j \in \mathcal{N}_i^-} \hat{y}_j(k), \quad (4.14)$$

$$z_i(k+1) = \frac{1}{\mathcal{N}_i^+ + 1} z_i(k) + \sum_{j \in \mathcal{N}_i^-} \hat{z}_j(k), \quad (4.15)$$

$$r_i(k+1) = \frac{1}{z_i(k+1)} y_i(k+1), \quad (4.16)$$

with $y_i(0) = y_i^0, z_i(0) = 1$ for all $i \in \mathcal{E}$. Here, $\hat{y}_j(k), \hat{z}_j(k)$ are the estimates sent by neighboring node j to agent i with $\hat{y}_j(k) = \frac{1}{\mathcal{N}_j^+ + 1} y_j(k), \hat{z}_j(k) = \frac{1}{\mathcal{N}_j^+ + 1} z_j(k)$. The estimates

M_i, m_i are updated as: Let $\mathcal{N}_i^u = \mathcal{N}_i^- \cup \{i\}$,

$$M_i(k+1) = \max_{j \in \mathcal{N}_i^u} M_j(k), \quad m_i(k+1) = \min_{j \in \mathcal{N}_i^u} m_j(k), \quad (4.17)$$

with $M_i(0) = m_i(0) = r_i(0)$, for all $i \in \mathcal{E}$. Given a tolerance ε , iterations (4.14)-(4.16) are terminated after k_ε such that $|M_i(k) - m_i(k)| \leq \varepsilon$, for $k > k_\varepsilon$ [258]. It is established in [258] that at any iteration k_{end} when the *GradConsensus* algorithm is terminated the estimates $r_i(k_{\text{end}})$ and $r_j(k_{\text{end}})$ of any two agents i and j are such that $|r_i(k_{\text{end}}) - r_j(k_{\text{end}})| \leq \varepsilon$ and $|r_i(k_{\text{end}}) - \frac{1}{N} \sum_{j=1}^N y_j^0| \leq \varepsilon$ for all $i, j \in \{1, 2, \dots, N\}$.

Remark 28. *The iterations (4.14)-(4.17) are amenable to distributed synthesis ([231]) as a GFM IBR only requires the communicated estimates $\hat{y}_j, \hat{z}_j$ from its neighbors for updating its states, no additional coordination is required. Further, as any GFM IBR j joins or leaves the network, GFM IBR i can update its neighborhood sets $\mathcal{N}_i^-, \mathcal{N}_i^+$ locally not requiring a centralized re-design. Hence, the GFM IBRs under the proposed algorithm operate in a plug-and-play manner.*

Solution to Problem (4.6):

Problem (4.6) is instantiated at every time t . The constraint $x_i = x_j$ for all $i, j \in \{1, 2, \dots, N\}$ couples the local estimates of all the GFM IBRs. Thus, a *distributed* strategy that allows the GFM IBRs to obtain an estimate of the solution to problem (4.6) while satisfying the coupling constraint (within a prescribed tolerance) is needed. We adopt the iterations (4.12) and (4.13) to provide such a distributed solution. Let $x_i(t)$ denote the estimate maintained at GFM IBR i at time t and $\varepsilon(t) > 0$ be any specified precision tolerance. Then the estimate $x_i(t)$ is updated as

$$x_i^+ := x_i(t-1) + \rho((1+\gamma)x_i(t-1) - \alpha_i(t-1)), \quad (4.18)$$

$$x_i(t) = \text{Consensus}_{\varepsilon(t-1)}(x_i^+), \quad (4.19)$$

At any time t , $x_i(t)$ is the estimate of the optimal solution x_t^* with GFM IBR i . Thus, having $x_i(t)$ at hand any GFM IBR i can determine the adjustment $v_i(t)$ at time t as,

$$v_i(t) := x_i(t) + \beta_{Q_i} m_i Q_i(t) - \beta_{V_i} (V^* - V_i(t)). \quad (4.20)$$

Next, we present a convergence result for updates (4.18)-(4.20).

Theorem 24. *Let graph $\mathcal{G}$ be strongly connected (i.e. $\mathcal{G}$ has a directed path between every pair of distinct nodes). Assume, that the iterations (4.18)-(4.20) achieve a steady state after time t_{ss} . Given, $\varepsilon > 0$, $|x_i(t) - x_t^*| \leq \varepsilon$, for $t \geq t_{ss}$, for all i .*

Proof. From (4.18) and (4.19), at any time t and any i ,

$$x_i(t+1) = \frac{(1 + \rho(1 + \gamma))}{N} \sum_{j=1}^N x_j(t) - \frac{\rho}{N} \sum_{i=1}^N \alpha_i(t) + \xi_i(t), \quad (4.21)$$

where $|\xi_i(t)| \leq \varepsilon(t)$. Taking the average over node index i ,

$$\frac{1}{N} \sum_{i=1}^N x_i(t+1) = \frac{(1 + \rho(1 + \gamma))}{N} \sum_{j=1}^N x_j(t) - \frac{\rho}{N} \sum_{i=1}^N \alpha_i(t) + \frac{1}{N} \sum_{i=1}^N \xi_i(t).$$

For $t > t_{ss}$, $\frac{1}{N} \sum_{i=1}^N x_i(t_{ss}) = \frac{(1 + \rho(1 + \gamma))}{N} \sum_{j=1}^N x_j(t_{ss}) - \frac{\rho}{N} \sum_{i=1}^N \alpha_i(t) + \frac{1}{N} \sum_{i=1}^N \xi_i(t)$. Therefore, from (4.21), $\frac{\rho(1 + \gamma)}{N} \sum_{j=1}^N x_j(t_{ss}) - \frac{\rho}{N} \sum_{i=1}^N \alpha_i(t) = x_i(t_{ss}) - \frac{1}{N} \sum_{j=1}^N x_j(t_{ss}) - \xi_t(t_{ss}) = -\frac{1}{N} \sum_{i=1}^N \xi_i(t)$. Thus,

$$\begin{aligned} x_i(t_{ss}) &= \frac{1}{N} \sum_{j=1}^N x_j(t_{ss}) + \xi_i(t) - \frac{1}{N} \sum_{i=1}^N \xi_i(t) \\ &= \frac{\rho(\sum_{i=1}^N \alpha_i(t))}{N(\rho(1 + \gamma))} - \frac{\sum_{i=1}^N \xi_i(t)}{N(\rho(1 + \gamma))} + \xi_i(t) - \frac{\sum_{i=1}^N \xi_i(t)}{N} \end{aligned}$$

Let $\varepsilon(t) < \varepsilon/2$ for $t > t_{ss}$, using the result of Lemma 20,

$$\begin{aligned} |x_i(t_{ss}) - x_t^*| &= \left| x_i(t_{ss}) - \frac{\sum_{i=1}^N \alpha_i(t)}{N(1 + \gamma)} \right| \\ &\leq \left| \frac{1 - \rho(1 + \gamma)}{N\rho(1 + \gamma)} \sum_{i=1}^N \xi_i(t) + \xi_i(t) \right| \leq \frac{1}{N} \sum_{i=1}^N |\xi_i(t)| + |\xi_i(t)| \\ &\leq \frac{1}{N} \sum_{i=1}^N \varepsilon(t) + \varepsilon(t) \leq \varepsilon. \end{aligned}$$

□

Remark 29. *During updates (4.18)-(4.20) the GFM IBRs share a non-linear estimate of their measurements, obscuring the private information and hence preventing malicious cyber-attacks. Moreover, the iterations (4.14)-(4.17) can be easily modified to work under*

various practical scenarios incorporating time-delays, dynamic GFM IBR topology [230], presence of communication noise [259]. Further, the detection of the malicious behaving GFM IBRs can also be achieved [236].

CHIL Demonstration and Results:

In this section, we will evaluate the performance of our approach using a CHIL-based demonstration.

A. Experimental Configuration:

We emulate the residential-scale sub-network of the North American low-voltage distribution feeder (single phase, 240V, 60Hz, 1.25MVA) from CIGRE Task Force [260] (see Figure 4.2). It has 10 GFM IBRs connected at various buses in the network, resulting in an asymmetrical connection. Figure 4.2 provides the location of the GFM IBRs. In this demonstration, we assume that each GFM IBR has equal ratings. The control logic for the inner-voltage loop and droop-based primary layer are adopted from [261], [262]. GFM IBRs are modeled with single-phase H-bridge topology with LC filters (L_f , R_f , and C_f as inverter side filter inductors, parasitic resistances and filter capacitor respectively) with electrical parameters, tabulated in Table 4.1. The droop control parameters of the GFM IBRs are tabulated in Table 4.1.

Table 4.1: 1-Phase GFM IBR Under Study

Inverter	Value
Ratings (1- ϕ)	240 V, 60 Hz, 200 kVA
Inverter Parameters	$V_{dc} = 500$ V, $f_{sw} = 20$ kHz
Filter Parameters	$L_f = 0.02$ mH, $R_f = 4$ m Ω , $C_f = 20$ μ F
Control	Value
Droop Parameters	$n_i = 0.32$ Hz/kW, $m_i = 40$ V/kVAr

On top of the power layer, there is a communication layer that enables the transmission of various measurements, from the North American low-voltage distribution feeder to the optimization engine, and various control signals, from the optimization engine to the North American low-voltage distribution feeder. There are three important components in the demonstration: i) the North American low-voltage distribution feeder and the GFM IBRs, emulated using the eMEGASIM platform with a simulation step

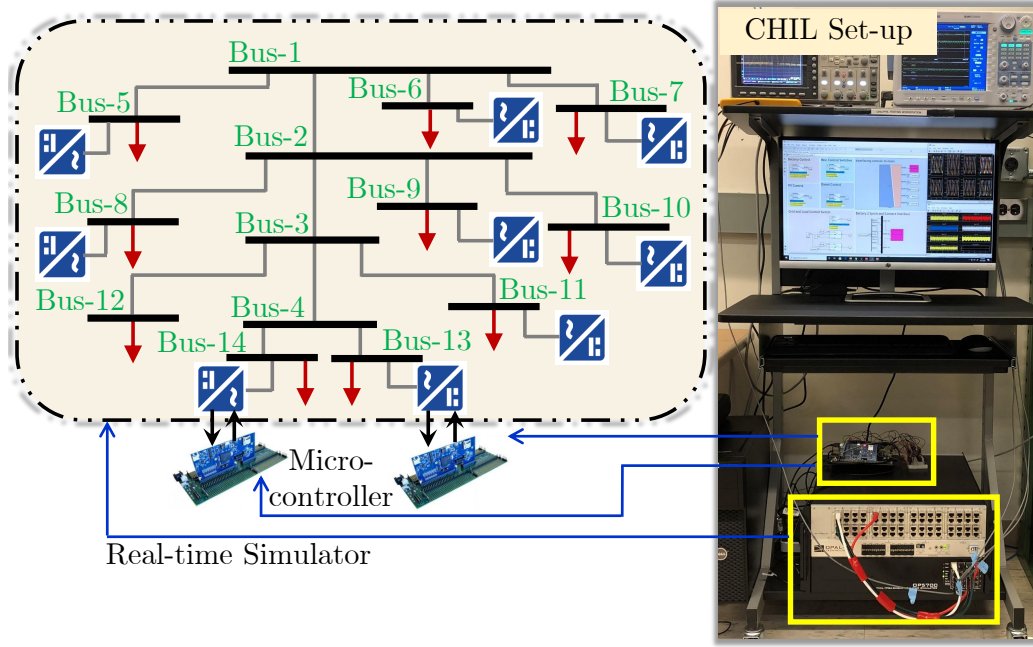


Figure 4.2: The laboratory-based system-in-the-loop (SIL) and controller hardware-in-the-loop (CHIL) experimental hardware setup.

size of $50\mu\text{s}$ inside the OP5700 RT-simulator (RTS), which is manufactured by OPAL-RT and interfaced with two low-cost Texas Instruments TMS320F28379D, 16/12-bit floating-point 200-MHz Delfino microcontroller boards for controlling two of the GFM IBRs; ii) the optimization engine of the IBRs running 10 independent instances via 10 parallel processes running the optimization iterations in Python 3.7.1 on a laptop with 16 GB RAM and an Intel Core i7 processor running at 1.90 GHz. The GFM IBRs are connected via a directed communication network having a diameter of 5 nodes; and iii) a Device Control Gateway (DCG), realized using standard User Datagram Protocol (UDP) [263], that interfaces the emulated system and the optimization engine and forms the communication layer that continuously listens to the measurements sent from the power network and adjustment commands from the optimization engine.

B. Results and Discussions:

Three test cases are demonstrated by emulating a sequence of events as follows:

- **CASE-1:** GFM IBRs operated without secondary control compensation until about 4.5 s. Then, at 4.5 s, distributed secondary control began, evenly sharing reactive power among GFM IBRs 1 to 10. At 13.5 s, system demand rose from 1000 kW to 1250 kW

for active power and from 750 kVAr to 1125 kVAr for reactive power. At 24 s, demand decreased back to 1000 kW for active power and 750 kVAr for reactive power.

- **CASE-2:** Until approximately 23 seconds, all GFM IBRs operated without the proposed secondary control compensation. Then, at 23 seconds, distributed secondary control for voltage regulation among GFM IBRs 1 to 10 began. By 34 seconds, the total system demand increased to 1250 kW for active power and 1125 kVAr for reactive power from 1000 kW and 750 kVAr respectively. Subsequently, at 47 seconds, the total system demand decreased back to 1000 kW for active power and 750 kVAr for reactive power.
- **CASE-3:** Until around 6 seconds, all GFM IBRs operated without the proposed secondary control compensation. Then, at 6 seconds, distributed secondary control began, with GFM IBRs 1 to 7 equally sharing reactive power, while GFM IBRs 8 to 10 regulated voltage. By 11.5 seconds, the total system demand increased to 1250 kW for active power and 1125 kVAr for reactive power from 1000 kW and 750 kVAr respectively. Subsequently, at 17 seconds, the total system demand decreased back to 1000 kW for active power and 750 kVAr for reactive power.

Figures 4.3(i)-(iv) provide the results of **CASE-1**. Here, the GFM IBRs share the total reactive power demand by providing an equal amount of 75 kVAr (0.375 pu) during 4.5 to 13.5 seconds during the reactive power sharing objective. The developed scheme tracks both the increase and the decrease in load with all the GFM IBRs sharing reactive power equally as shown in Figure 4.3(ii). The voltages at the GFM IBRs however are not tightly regulated as seen in Figures 4.3(iii). Figure 4.3(i) and Figure 4.3(iv) show that the proposed secondary control is not influencing the active power sharing functionality and as result the frequency of the system is also kept almost unchanged with minimal transients.

Figures 4.3(v)-(viii) provide the results of **CASE-2**. It can be seen from Figure 4.3(vii) that the voltages of the GFM IBRs are tightly regulated around the reference voltage under **CASE-2**. The total reactive power is not shared equally among the GFM IBRs as seen in Figure 4.3(vi). Similarly here also, as shown in Figure 4.3(v) and Figure 4.3(viii), the proposed secondary control is not influencing the active power sharing functionality and the system frequency has minimal transients.

Figures 4.3(ix)-(xii) present the system state under **CASE-3**. Here, three GFM IBRs (GFMs 8 – 10) have unequal sharing of reactive power while all the remaining seven GFM IBRs (GFMs 1 – 7) have their voltages tightly regulated. This shows the versatility of our proposed scheme in achieving both voltage regulation and equal-rated reactive

power-sharing objectives. Similarly, the control doesn't not adversely affect the active power sharing of the GFM IBRs and the frequency of the system is maintained around the nominal value of 60 Hz as seen in Figures 4.3(ix) and Figures 4.3(xii).

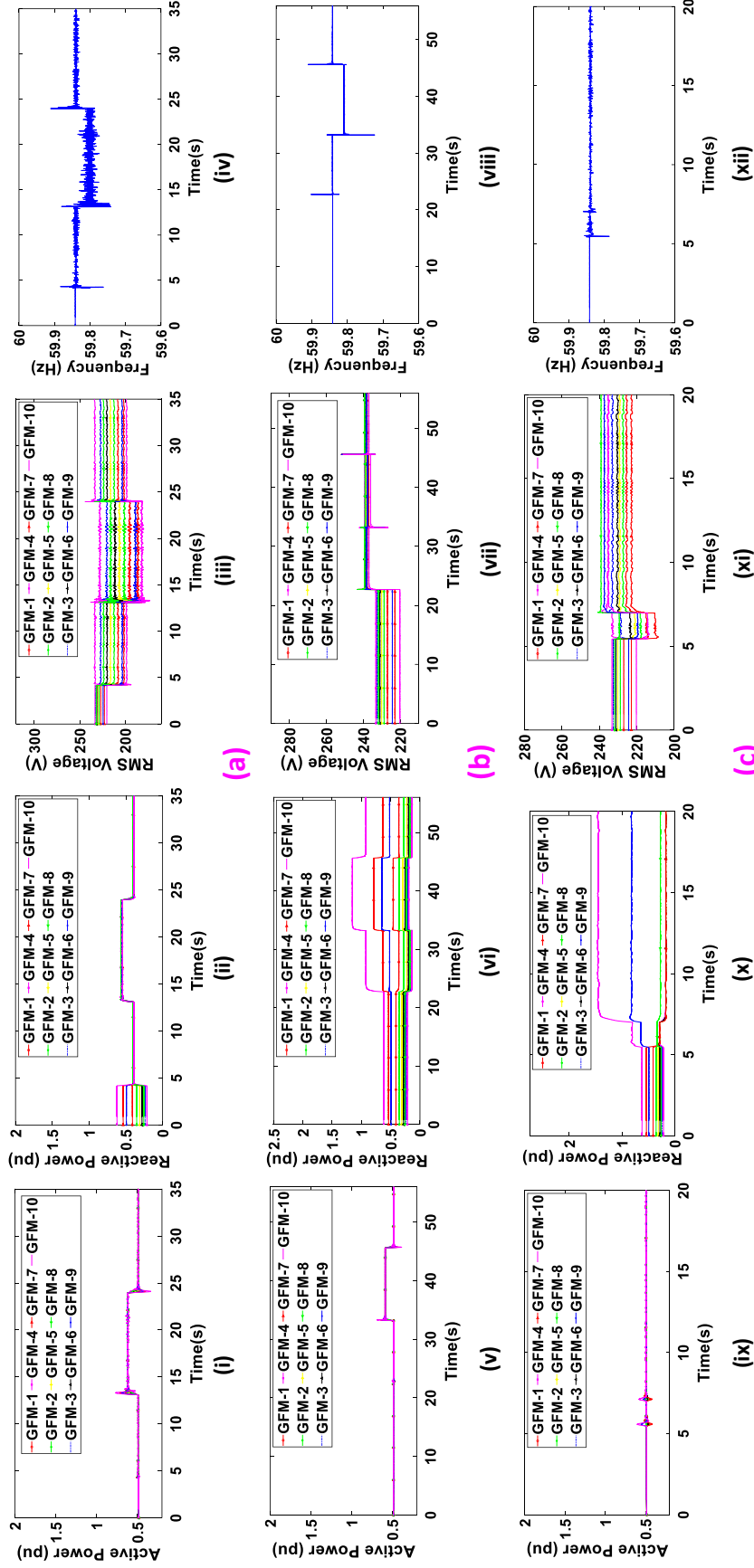


Figure 4.3: Performance results of the developed distributed secondary controller. (a) Under CASE-1: equal-rated reactive power sharing, (b) Under CASE-2: tight voltage regulation, (c) Under CASE-3: mixed objective DERs 1-7 equal-rated reactive power sharing and DERs 8-10 tight voltage regulation.

4.5.2 Resilient Power Apportioning in Microgrids with Online Malicious Behavior Detection

System Description:

A. Description of the Microgrid:

The electrical network is developed based on a power network at the University of Minnesota campus, called the “Central Core”, and the suburb Minneapolis distribution network. The schematic diagram of the electrical network is shown in Figure 4.4(a), which consists of commercial-scale critical infrastructure units (csCIu) on Bus 3, central core units (CCus) Bus 4-9, mobile auxiliary generation units on Bus 8, and a cold load station located at Bus 5. Battery grid-forming (GFM) inverters [264] (denoted as B,B₁,B₂ in Figure 4.4(a)) are designed based on P-f and Q-V droop law. Photovoltaic (PV) grid-following (GFL) inverter system [264] is designed based on outer P-Q controller. Diesel generation sets (denoted as G₁,G₂ in Figure 4.4(a)) are designed based on load-frequency-based governor control and Q-V-based automatic voltage regulator. The ratings of the central core are summarized in Table 4.2. The central core demands, $\rho_d(t_s)$, and since it contains the critical infrastructure, the external power network needs to support the central core and generate active power output that is equal to $\rho_d(t_s)$. The external power network is connected to the central core at the feeder near Bus-5. The power network is a 3- ϕ , 0.48 kV, 60 Hz, 55-bus distribution network with 10 residential apartments, called highly-scaled distributed units (hsDu), and 5 residential-scale central core units (rsCCu), each of which contains 8 hsDu. The external network has 50 hsDus with controllable DERs. *In the later, we will use the terms hsDu and DER interchangeably.* The ratings of the external power network are summarized in Table 4.3. To meet the requested power demand of the central core the external power network with 50 DERs is employed with a power apportioning engine, running Algorithm 1 of [228]. The central core, at a given time t_s , commands a total power demand set point

Table 4.2: Ratings of Various Sources and Loads in Central Core of the Microgrid

Types	Central Core of the Microgrid							
	csCIu	CCu3	CCu4	CCu5	CCu6	CCu7	CCu8	Mobile Unit
Battery GFM Inverter (kVA)	200	25	100	80	200	100	50	100
Diesel Gen – Set (kVA)	100	-	-	-	-	-	-	-
PV GFL Inverter (kVA)	50	50	-	100	100	50	100	50
Uncontrollable Load (kVA)	250	30	35	30	40	25	30	-
Controllable Load (kVA)	500	300	315	300	360	225	270	-

Table 4.3: Ratings of Various Sources and Loads in External Network of the Microgrid

Types (kVA)	hsDu in External Microgrid										hsDu Groups				
	No. 1	No. 2	No. 3	No. 4	No. 5	No. 6	No.7	No. 8	No.9	No.10	No. 1	No. 2	No. 3	No. 4	No. 5
PV	3	5	-	5	5	5	2	5	2	2	40	24	64	32	40
Battery	3	2	8	-	5	-	7	7	2	5	16	32	32	24	24
Load	3	2	2	4	6	1	3	7	3	3	24	20	24	20	28

based on its current situation. The DERs' decisions are aggregated via a communication network. Figures 4.4(b)(i) and (b)(ii) show two representative communication topologies, one with sparse connectivity and the other with dense connectivity respectively. Each node in the communication network is a hsDu with a communication interface, together referred to as an agent responsible for generating the active power set point for the corresponding DER based on the results obtained from the power apportioning engine. The power apportioning engine is described in detail next.

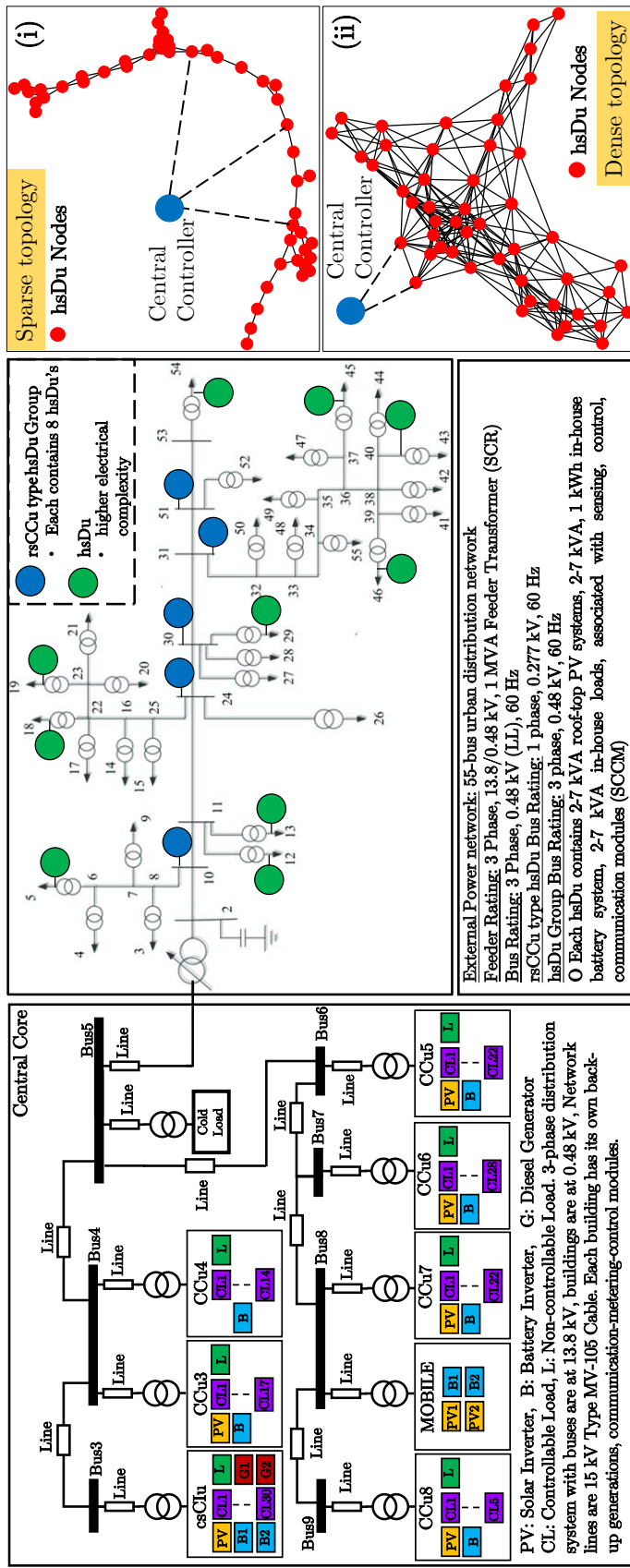


Figure 4.4: Figure of (a) power network of the microgrid under study; (b) communication topology used in the power apportioning.

B. Distributed Power Apportioning:

The central controller is an entity interfacing with the central core on one end and the hsDu network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, with $|\mathcal{V}| = 50$, on the other. The central controller, similar to a power aggregator in the Transmission System, aggregates the DERs available in the external network to provide ancillary power to the central core. Considering the geographical span of the external power network it may not be feasible for all DERs to directly communicate with the central controller. Hence, only some DERs given by the set $\mathcal{N}_d \subseteq \mathcal{V}$, with $l = |\mathcal{N}_d|$ can directly communicate with the central controller. The central controller aggregates the central core's power demand $\rho_d(t_s)$, and relays it to DER $i, i \in \mathcal{N}_d$ at time instant $t_s \geq 0$. The DERs need to collectively meet the demand $\rho_d(t_s)$ while communicating only to their neighboring DERs in a connected communication graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ and respecting individual resource constraints. In particular, the DERs with power reference commands $\pi_i^*(t_s)$, ensure $\sum_{i \in \mathcal{V}} \pi_i^*(t_s) = \rho_d(t_s)$, and the capacity constraints of the DERs are not violated, that is $\pi_i^{\min}(t_s) \leq \pi_i^*(t_s) \leq \pi_i^{\max}(t_s)$, where $\pi_i^{\min}(t_s)$ and $\pi_i^{\max}(t_s)$ denote the maximum and minimum generation capacities of the DER i at time instant t_s . State of the art on solving distributed apportioning involves an iterative finite-time average consensus protocol [228], [229]:

At each iteration k of the protocol, every DER sends three internal states, $x_i(k), y_i(k), z_i(k)$ to its out-neighbors $\mathcal{N}_j^+ := \{i | (i, j) \in \mathcal{E}, i \neq j\}$, and receives these estimates from its in-neighbors $\mathcal{N}_j^- := \{j | (j, i) \in \mathcal{E}, i \neq j\}$. The states are then updated as follows:

$$x_i(k+1) = p_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i^-} p_{ij}x_j(k), \quad (4.22)$$

$$y_i(k+1) = p_{ii}y_i(k) + \sum_{j \in \mathcal{N}_i^-} p_{ij}y_j(k), \quad (4.23)$$

$$z_i(k+1) = \frac{1}{y_i(k+1)}x_i(k+1) \quad (4.24)$$

where $p_{ij} > 0, i, j \in \mathcal{V}$ denotes predefined weights and $x_i(0) = (\rho_d(t_s)/l) - \pi_i^{\min}(t_s)$, if $i \in \mathcal{N}_d$, $x_i(0) = -\pi_i^{\min}(t_s)$ if $i \in \mathcal{V} \setminus \mathcal{N}_d$, and $y_i(0) = \pi_i^{\max}(t_s) - \pi_i^{\min}(t_s)$. The updates (4.22)-(4.24) are accompanied by additional maximum and minimum consensus protocols for terminating the iterations and producing a faster decision in finite time [228], [229]. In particular, when the difference between the global maximum and minimum goes below

a specified tolerance ε then algorithm iterations are terminated with the state value

$$z_i^* = \frac{\rho_d(t_s) - \sum_{i \in \mathcal{V}} \pi_i^{\min}(t_s)}{\sum_{i \in \mathcal{V}} (\pi_i^{\max}(t_s) - \pi_i^{\min}(t_s))}. \quad (4.25)$$

The DERs then dispatch the following power reference set-points,

$$\pi_i^*(t_s) := \pi_i^{\min}(t_s) + z_i^*(\pi_i^{\max}(t_s) - \pi_i^{\min}(t_s)).$$

Note that

$$\sum_{i \in \mathcal{V}} \pi_i^*(t_s) = \rho_d(t_s).$$

Articles [228], [229] provide more details about the distributed power apportioning protocol.

Malicious Behavior of DERs:

We considered the distributed apportioning network in the previous section and its ability to provide ancillary services to the central core with aggregation of hsDus. For the updates (4.22)-(4.23) of any DER utilize the information shared by its neighbors under the assumption of trustworthy cooperation towards the objective of meeting the ancillary service goal for the central core. This leaves the DER network vulnerable to malicious agents trying to manipulate the power-apportioning protocol towards selfish goals or something even more severe by disrupting the convergence of the apportioning protocol to make the central core system unstable. We model the malicious behavior via an intruder model. Let $\mathcal{N}_m$ denote the set of maliciously behaving DERs. Let k_m denote the iteration of the start of the malicious behavior for any malicious DER m . Any DER $m \in \mathcal{N}_m$ applies a linear drift $\theta_m(k)$ to its true state $x_m(k)$ and sends

$$\hat{x}_m(k) := x_m(k) + \theta_m(k), \quad (4.26)$$

for all $k \geq k_m$, to its neighbors. The update of the neighboring DER i of the malicious DER m in the power apportioning protocol becomes,

$$x_i(k+1) = p_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i^- \setminus m} p_{ij}x_j(k) + p_{im}(x_m(k) + \theta_m(k)). \quad (4.27)$$

Different choices of the perturbation θ_m allow the malicious DER m to achieve different objectives, in particular

- a) if $\theta_m(k)$ is a divergent (bounded or unbounded) sequence then the updates (4.22)-(4.23) for the DERs do not converge
- b) if $\theta_m(k)$ is convergent series with $\sum_k \theta_m(k) = \alpha$. This leads to the DER solution z_i^* to converge to $\frac{\rho_d(t_s) - \sum_{i \in \mathcal{V}} \pi_i^{\min}(t_s) + \alpha}{\sum_{i \in \mathcal{V}} (\pi_i^{\max}(t_s) - \pi_i^{\min}(t_s))}$ (this can be shown using [229], Theorem III.1), leading to $\sum_{i \in \mathcal{V}} \pi_i^*(t_s) = \rho_d(t_s) + \alpha$, which can be made arbitrarily smaller/larger than $\rho_d(t_s)$.

Remark 30. *Although we consider drift in the states x_i by the malicious agents, a similar deviation in the y_i updates can be considered. However, the malicious DER can cause any desired result by only manipulating x_i states and doesn't have additional benefits by manipulation of the y_i states. Further, the intruder model with states $\hat{x}_m$ and the update (4.27) allows for modeling DERs that are not necessarily malicious but are misbehaving (and send imperfect values to neighboring DERs) due to errors in computation or communication or any other internal failures.*

A. Resilient Distributed Power Apportioning:

In light of the malicious DERs, we strive to equip each DER with the ability to detect deviant behavior and isolate the communication links of the malicious DERs. We make the following assumption:

Assumption 18. *At the beginning of the power apportioning, i.e. $k = 0$, the information sent by the DERs is truthful.*

Next, we describe this strategy: Given $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, each DER maintains additional states $x_j^i(k)$ for all its neighbors $j \in \mathcal{N}_i^-$ and a detection_flag d_i utilized by DER i to indicate the presence of a malicious DER in its neighborhood. The distributed apportioning protocol is modified to allow each DER i to utilize these additional states x_j^i . The augmented distributed apportioning protocol has the following key steps:

- At the start of the apportioning protocol, i.e. $k = 0$,
 - ◊ each DER i , in addition to $x_j(0), y_j(0)$, requires all its in-neighbors $j \in \mathcal{N}_i^-$ to send their weights p_{jj} .
- At any subsequent iteration $k \geq 1$,
 - ◊ every DER j sends $x_j(k), y_j(k)$ and an additional state $s_j(k-1) = \sum_{l \in \mathcal{N}_j^-} p_{jl} x_l(k-1)$ to its out-neighbors $i \in \mathcal{N}_j^+$.

◇ every DER i then updates its states to $x_i(k+1), y_i(k+1), z_i(k+1)$ via the updates (4.22)-(4.24), and the states x_j^i as $x_j^i(k) = p_{jj}x_j(k-1) + s_j(k-1)$.

◇ if there is a mismatch between the estimated state $x_j^i(k)$ and the actual received state $x_j(k)$ then the agent i detects that the communication links between its neighbor j and itself are compromised. In particular, with small $\varepsilon > 0$, if $|x_j^i(k) - x_j(k)| > \varepsilon$ then agent i detects node j is compromised. Further, DER i sets its detection_flag d_i to 1 and adds the communication address of DER j to the list of compromised communication links, $\mathcal{C}_i^m$, in its neighborhood.

◇ Any DER i that detects a malicious DER in its neighborhood, propagates its detection_flag in the network. This is achieved via a 1-bit consensus algorithm. Each DER initializes $d_i(k)$ at every $k = l\mathcal{D}$, where $\mathcal{D}$ is an upper bound on the diameter of the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, iteration for $l \in \{0, 1, 2, \dots\}$ with 1 or 0 depending on whether the DER has detected a malicious DER in its neighborhood or not, and updates its detection_flag value every iteration using,

$$d_i(k+1) = \bigcup_{j \in \mathcal{N}_j^- \cup \{i\}} d_j(k), \quad (4.28)$$

where $\bigcup$ denotes the “OR” operation and $d_j(0) = 1$ if node j has detected a malicious DER at initialization instant 0 and $d_j(0) = 0$ otherwise. Clearly, if $d_j(0) = 1$ for any DER $j \in \mathcal{V}$, then $d_i(\mathcal{D}) = 1$ for all $i \in \mathcal{V}$ where $\mathcal{D}$ is the diameter. Thus, each DER can use $d_i(\mathcal{D})$ as a criterion to ascertain the presence of a malicious DER in the network.

The modified power apportioning algorithm is presented in Figure 4.5. We present the following result for the detection scheme.

Theorem 25. *Let Assumption 18 hold. Then the detection scheme detailed above and in Figure 4.5 leads to detecting any malicious DER obscuring its true states via (4.26).*

Proof. We present an argument by considering a particular DER index i , the argument holds symmetrically for all DERs $i \in \mathcal{V}$. Following the procedure detailed in Figure 4.5 at the starting iteration $k = 0$ node i receives $x_j(0), y_j(0), p_{jj}$ from all its in-neighboring DERs. Note that under Assumption 18 $x_j(0), y_j(0)$ and p_{jj} are the true values of these quantities. Following this, DER i updates its states following (4.22)-(4.23), $x_i(1) = p_{ii}x_i(0) + \sum_{j \in \mathcal{N}_i^-} p_{ij}x_j(0)$, $y_i(1) = p_{ii}y_i(0) + \sum_{j \in \mathcal{N}_i^-} p_{ij}y_j(0)$, $z_i(1) = \frac{x_i(1)}{y_i(1)}$. At $k = 1$, the DER i sends $x_i(1), y_i(1)$ and $s_i(0) = \sum_{l \in \mathcal{N}_i^-} p_{il}x_l(0)$ to its out-neighboring DERs and receives the $x_j(1), y_j(1)$ and $s_j(0) = \sum_{l \in \mathcal{N}_j^-} p_{jl}x_l(0)$ from its in-neighboring DERs

j . Following this DER i determines the estimates $x_j^i(1) = p_{jj}x_j(0) + s_j(0)$. Without loss of generality assume DER j behaves maliciously at iteration $k = 1$, and adds a linear drift $\theta_j(1)$ to the true state $x_j(1)$ and sends $\hat{x}_j(1) = x_j(1) + \theta_j(1)$ to its out-neighbors including DER i . Thus at DER i , given a small detection threshold ε , $|\hat{x}_j(1) - x_j^i(1)| = |p_{jj}x_j(0) + s_j(0) - x_j(1) - \theta_j(1)| = |p_{jj}x_j(0) + s_j(0) - p_{jj}x_j(0) - s_j(0) - \theta_j(1)| = |\theta_j(1)| > \varepsilon$ and j gets detected as a maliciously behaving DER by DER i . Similarly, all other DERs in the network compute the difference between the states sent by their in-neighbors and the estimates maintained locally to detect any malicious DER in their neighborhood. Thus, any linear drift θ_j aimed to shift the DER estimates significantly away from the true state values can be detected. This completes the proof. $\square$

After all the DERs ascertain the presence of malicious DERs via the `detection_flag`, a new instance of the distributed apportioning algorithm is started among the trustworthy DERs by removing the compromised communication links $C_i^m \in \mathcal{E}$ from their neighborhoods.

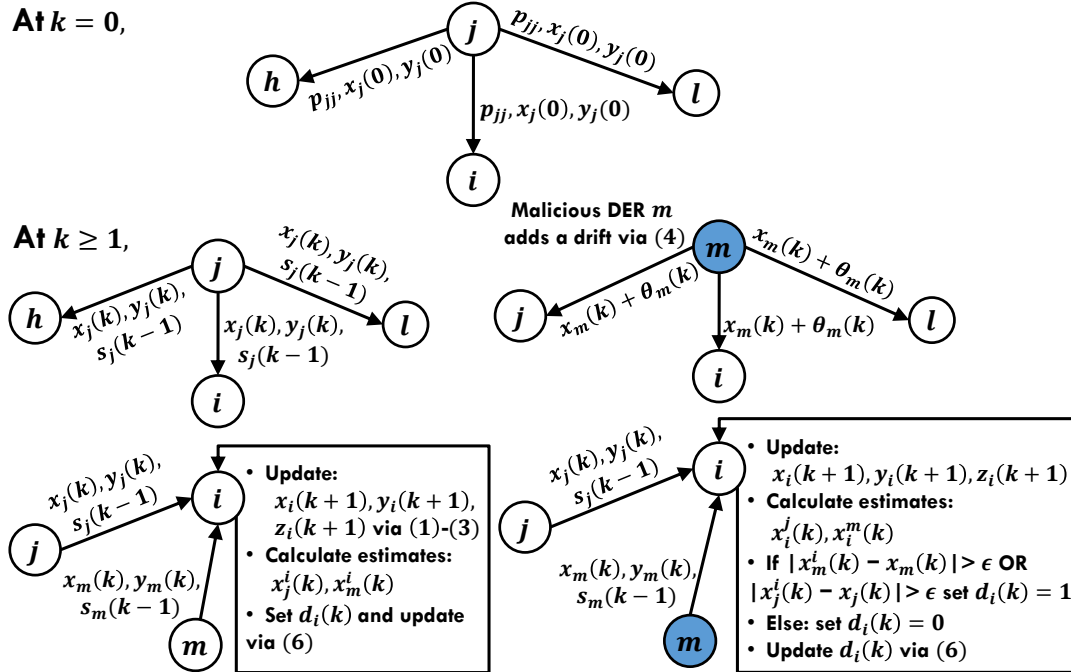


Figure 4.5: Resilient distributed power apportioning scheme with malicious DER m . Honest DER i detects the presence of m .

CHIL Demonstration and Results:

A. Experimental Configuration:

There are three important components in the experimental setup:

- 1.** The microgrid power network under study consisting of the Central Core and the external power network as shown in Figure 4.4(a) is emulated using the multi-timescale real-time simulation method, with time-step of $T_s = 50\mu s$ and $T_{ss} = 5ms$ via eMEGASIM and ePHASORSIM platform respectively inside the OP5700 RT-simulator (RTS) manufactured by OPAL-RT.
- 2.** The central core is equipped with a central dispatch control system that provides high-speed, cost-optimal coordination of net loads (a combination of generation and controllable loads) [265], [266]. We refer to this central dispatch controller as the NLM engine. The NLM runs on a desktop computer with 16 GB RAM and an Intel Core i7 processor running at 1.90 GHz, utilizing Python 3.7.1. The devices in the central core of the emulated microgrid communicate with the NLM engine via using standard User Datagram Protocol (UDP) [263].
- 3.** The resilient distributed power apportioning algorithm with 50 nodes connected via pre-defined topology information, embedded in a JavaScript Object Notation (JSON) file, running in a Raspberry-based cluster device. The NLM engine transmits the total power demand command ρ_d to the power apportioning engine via UDP and the power set point for all the 50 DERs, situated in the external electrical network of the emulated microgrid, are transmitted back to the RT simulator via UDP after the apportioning algorithm has converged.

Realistic load profiles (obtained through the metering system by the energy management group of the University of Minnesota), and the real-world solar irradiance profiles (obtained from the National Renewable Energy Laboratory's Solar Measurement and Instrumentation Data Center and the National Solar Radiation Database [267]) are fed into the real-time simulator with 1 sec resolution. Figure 4.6 shows the entire CHIL experimental setup.

We consider two attack scenarios where hsDu 4 is malicious and has two different objectives. For both attack scenarios, the total power demand requested by the central core is 300 kW. Further, we validate the performance of the proposed resilient power apportioning scheme with two extreme cases of communication topology, as shown in Figure 4.4(b)(i) and Figure 4.4(b)(ii), between the hsDus: one sparse (here hsDu 14, 17 and 46 are the neighbors of hsDu 4) and another dense network (here hsDu 4 has 15

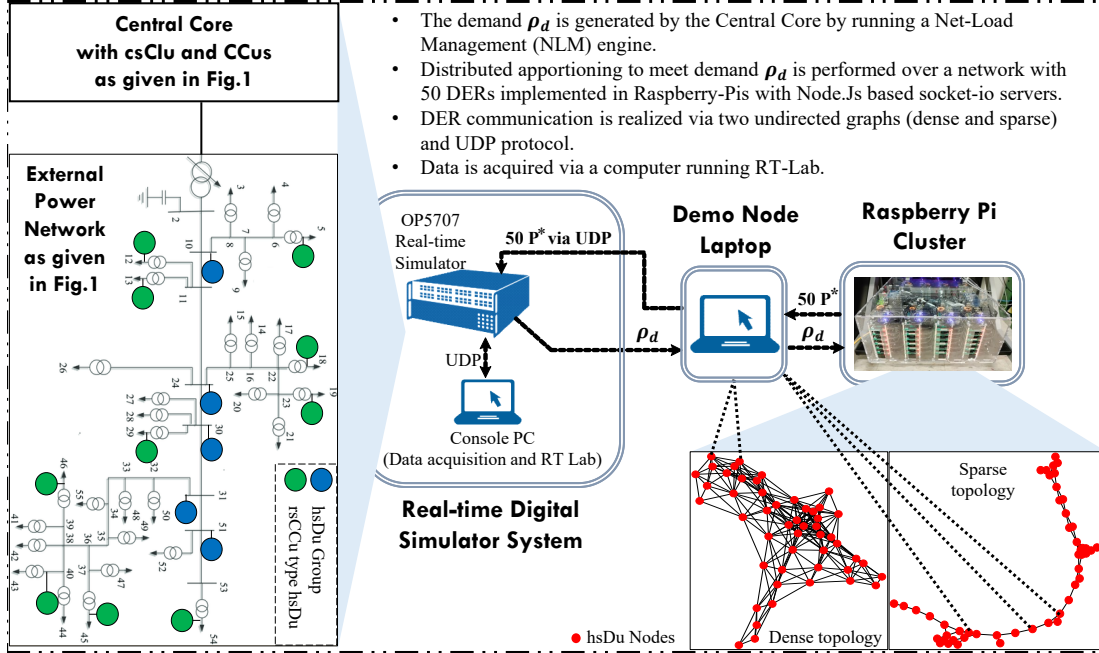


Figure 4.6: Laboratory CHIL setup.

neighboring hsDus). The convergence of updates (4.22) and (4.24) to the solution (4.25) depends on the connectivity of the underlying graph $\mathcal{G}$. Therefore, the sparse and dense topologies with very different connectivities provide a comparison of the performance of the developed resilient distributed apportioning algorithm under practical conditions. The two attack scenarios are as follows:

- **ATTACK-1 Operation Disruption Attack:** The malicious hsDu 4 injects false state estimates in the power apportioning scheme such that the algorithm does not converge and no power ancillary support to the central core can be provided. To achieve this hsDu 4 chooses a divergent perturbation sequence $\theta_4(k) = \delta, k : \text{even}$ and $\theta_4(k) = -\delta, k : \text{odd}$.
- **ATTACK-2 Operation Manipulation Attack:** The malicious hsDu 4 injects false state estimates to alter the steady-state power dispatch set-points of the honest hsDus to provide only 50%, 150 kW, of the requested ancillary demand.

B. Results and Discussions:

During the CHIL experiments for both attack scenarios, we perform **two** runs of the distributed power apportioning engine: (i) without and (ii) with the distributed detection scheme. We report the results without the detection scheme first to show the

impact of the attack on the power network and then present the performance of the apportioning algorithm with the detection scheme. Figures 4.7 and 4.8 provide the results of the CHIL experiments under the attack scenarios ATTACKs-1 and-2. In all the Figs. we present the total power output of the hsDu network in the top half and the power output of individual hsDus in the bottom half.

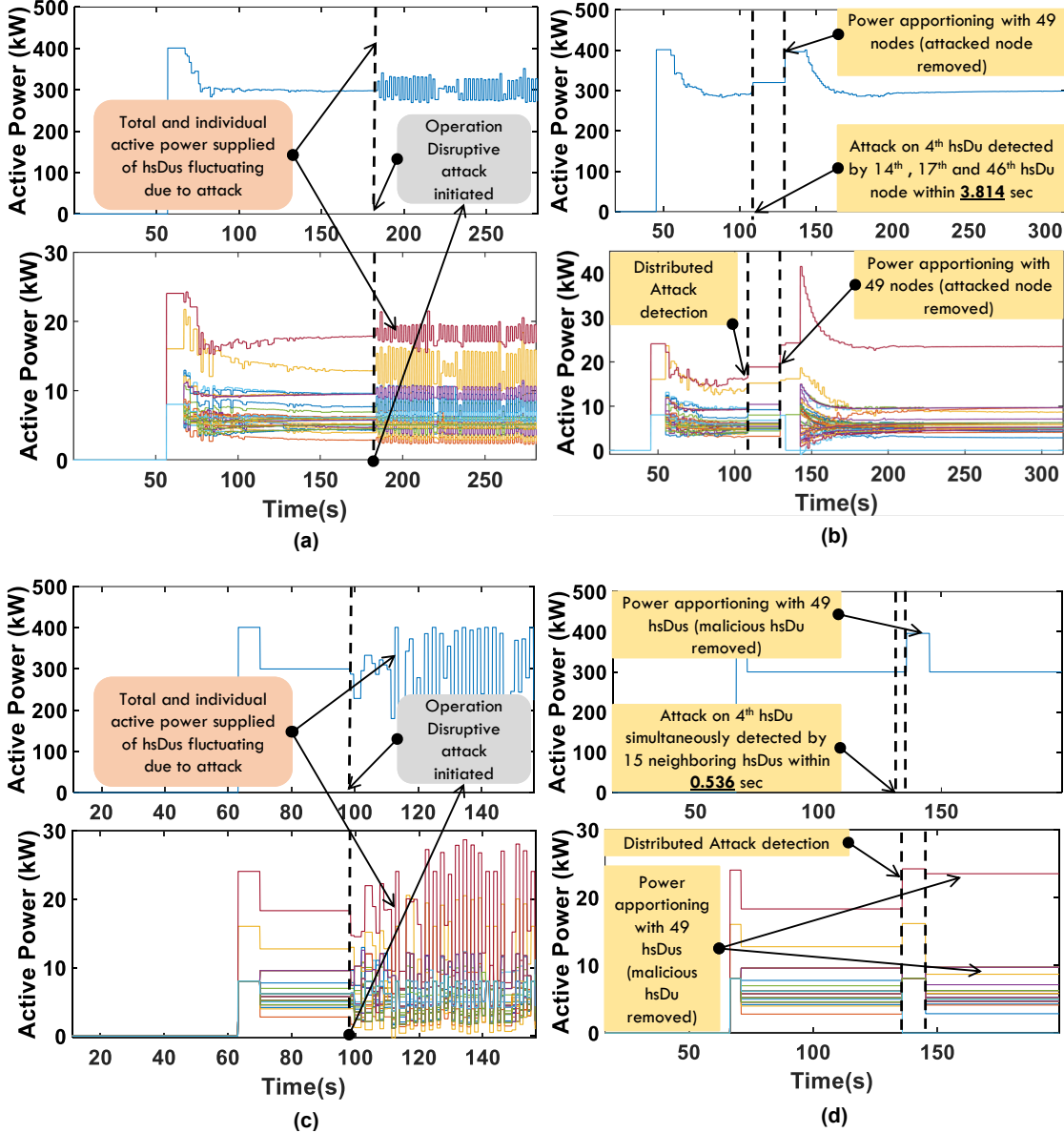


Figure 4.7: Performance of the distributed power apportioning scheme with malicious hsDu detection under ATTACK-1.

Figures 4.7(a)-(b) and Figures 4.7(c)-(d) present the results under **ATTACK-1** for the sparse and the dense communication topology respectively. It can be seen that the power set-points of the hsDus do not reach a steady state and fluctuate a lot (Figures 4.7(a),(c)) and potentially can make the system unstable if no countermeasures are taken by the honest hsDu towards detection of the malicious hsDu 4. With the proposed detection scheme the hsDus in the neighborhood of hsDu 4 detects the operation disruption attack within 3.814 secs and 0.536 secs in the sparse and dense communication topology respectively (Figures 4.7(b),(d)). Following the detection of the malicious hsDu 4, the remaining 49 hsDus restart power apportioning to provide the requested demand of 300 kW to the central core.

Figures 4.8(e)-(f) and Figures 4.8(g)-(h) present the results under **ATTACK-2** for the sparse and the dense communication topology respectively. Although the honest hsDus can meet the desired demand of 300 kW (Table 4.3). The hsDu 4 is able to manipulate the convergence of the power apportioning algorithm reducing the ancillary power support to the central core to 50% (150 kW), Figures 4.8(e),(g). Thus, a mechanism to detect the malicious actions of the hsDus is necessitated. With the proposed detection scheme the hsDus in the neighborhood of hsDu 4 detects the operation manipulation attack within 5.016 secs and 0.838 secs in the sparse and dense communication topology respectively (Figures 4.8(f),(h)). Following the detection of the malicious hsDu 4, the remaining 49 hsDus restart the power apportioning to provide the requested power demand of 300 kW to the central core. We remark that the time of detection of the attack is smaller in the dense communication topology as in the dense graph there are more simultaneous detections due to a higher degree of the hsDus.

Thus, the augmentation of the proposed malicious node detection with the power apportioning leads to a resilient operation of microgrids allowing the hsDus to aggregate their resources in a robust manner to meet the ancillary power demand of the central core.

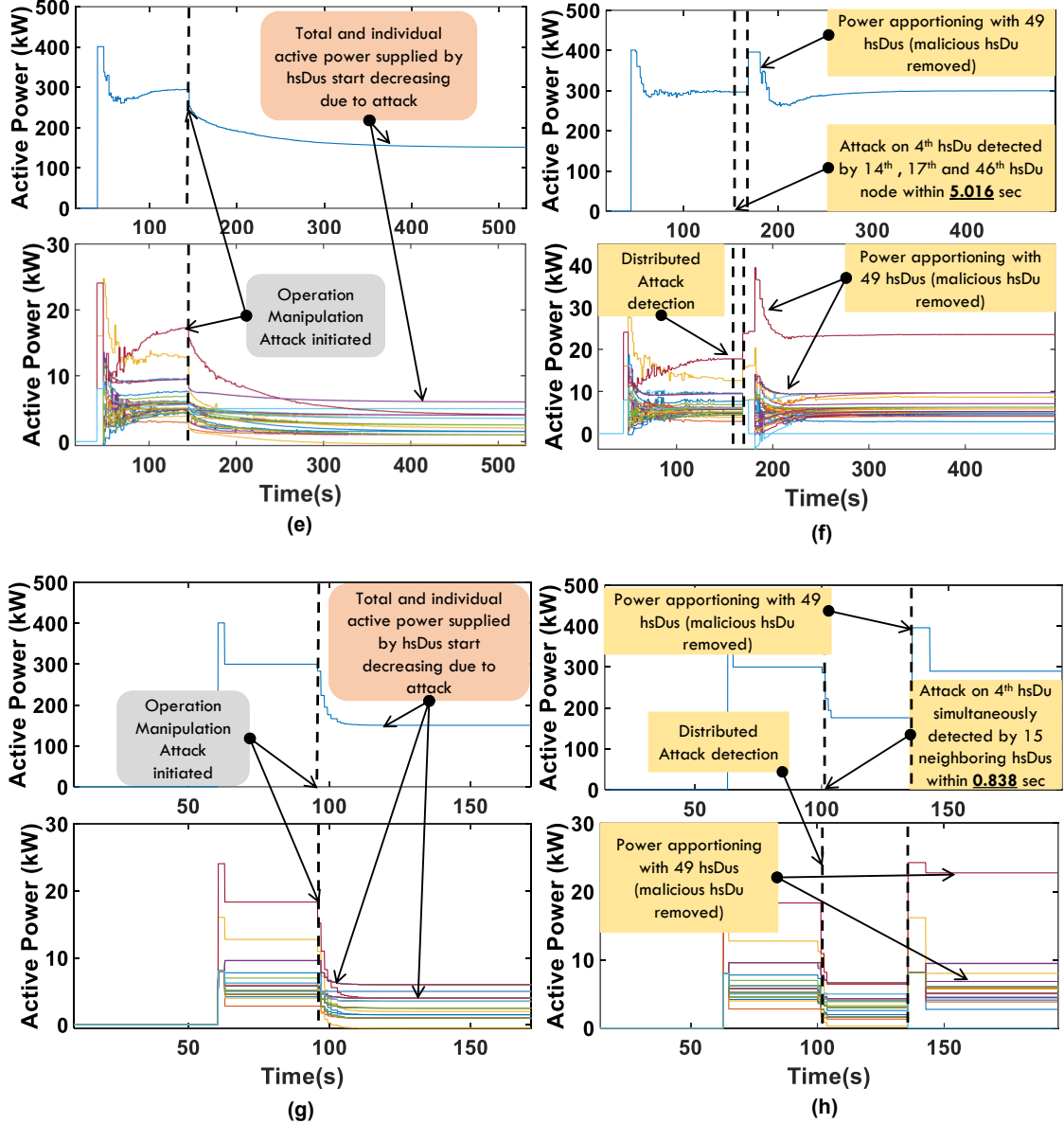


Figure 4.8: Performance of the distributed power apportioning scheme with malicious hsDu detection under and-ATTACK-2 (e)-(h).

4.5.3 Horizon of Viability Optimization Engine for Sustained Power to Critical Infrastructure

Problem Formulation:

Consider a microgrid with a CI unit electrically connected via power lines with surrounding community building (CB) units. The microgrid, as a whole, contains a collection of net-load resources: n_d number of DLs, n_c number of CLs, n_g number of diesel generators, n_p number of renewable generation sources including solar PV inverters and/or wind generators, etc., n_b number of grid-following (GFL) battery inverter systems; and a slack bus (combining all the grid-forming inverter systems in the microgrid) to maintain stable voltage and frequency in the microgrid. We assume that the renewable generation exhibits significant variability. Let $P_{r_j}^t$ denote the renewable generation power of the j^{th} renewable energy resource at time t . The power delivered by the j^{th} diesel generator and the j^{th} GFL battery inverter at time t are denoted by $P_{g_j}^t$ and $P_{b_j}^t$, respectively. The power delivered by the slack bus at time t is denoted by P_s^t . Unlike the diesel generator, the GFL battery inverters are allowed both, to discharge with a positive power output and/or charge with a negative power set point, depending on the net-load and the respective SOC. Here, each battery inverter, j , is characterized by the maximum available energy capacity, E_{b_j} , and the maximum discharging, W_{b_j} , and charging, $-W_{b_j}$, power transfer rates. The maximum capacity of the j^{th} diesel generator and the slack bus are denoted by W_{g_j} and W_s , respectively. The maximum available energy in the diesel generator, j , at the start of the time, t_0 , of the horizon is denoted by E_{g_j} .

HoV Engine Optimization Problem:

Here, a convex optimization problem is solved at each discrete time instant, t_0 , to determine the net-load scheduling decisions at some future time instant, $t_0 + t$, where $t \in \{1, 2, \dots, T\}$, with T being the horizon length. The dispatch decisions are based on renewable generation forecasts, predicted profiles of the DLs, CLs in the microgrid, and information about the latest SOC and SOE of the GFL battery inverters and diesel generators, respectively. This process is repeated at the subsequent time instants with updated sets of measurements. Denote by $x^t := [x_1^t, x_2^t, \dots, x_{n_d}^t] \in \{0, 1\}^{n_d}$, $P_g^t = [P_{g_1}^t, P_{g_2}^t, \dots, P_{g_{n_g}}^t] \in \mathbb{R}^{n_g}$, $P_b^t = [P_{b_1}^t, P_{b_2}^t, \dots, P_{b_{n_b}}^t] \in \mathbb{R}^{n_b}$, and $P_s^t \in \mathbb{R}$, $\mathbb{T}_{t_0} := \{t_0, \dots, T + t_0 - 1\}$. The HoV engine optimization problem at any time, t_0 , has the following components:

A. Optimization Variables:

- a) $\mathbf{x} = [x^t, t \in \mathbb{T}_{t_0}] \in \{0, 1\}^{n_d \times T}$, where, x^t denotes the ON/OFF decisions of the DLs at time t
- b) $\mathbf{P}_b = [P_b^t, t \in \mathbb{T}_{t_0}] \in \mathbb{R}^{n_b \times T}$, where P_b^t is the power set points of the GFL battery inverters at time t
- c) $\mathbf{P}_g = [P_g^t, t \in \mathbb{T}_{t_0}] \in \mathbb{R}^{n_g \times T}$, where P_g^t is the power set points of the diesel generators at time t
- d) $\mathbf{P}_s = [P_s^t, t \in \mathbb{T}_{t_0}] \in \mathbb{R}^T$, where P_s^t is the slack bus power required to maintain the load and generation power balance at time t .

Let $\pi := [\mathbf{x}, \mathbf{P}_b, \mathbf{P}_g, \mathbf{P}_s]$ denote a policy of net-load dispatch decisions over the horizon $[t_0, t_0 + T]$.

B. Real Time Measurements:

- a) $s_b^{t_0} = [s_{b_j}^{t_0}, j = 1, \dots, n_b] \in [0, 1]^{n_b}$, where, $s_{b_j}^{t_0}$ denotes the SOC of the GFL battery j at time t_0
- b) $s_g^{t_0} = [s_{g_j}^{t_0}, j = 1, \dots, n_g] \in [0, 1]^{n_g}$, where, $s_{g_j}^{t_0}$ denotes the SOE of the diesel generator j at time t_0 .

C. Predicted Variables:

- a) $\hat{P}_L^t = [\hat{P}_{L_j}^t, j = 1, \dots, n_d, t \in \mathbb{T}_{t_0}] \in \mathbb{R}^{n_d \times T}$, where $\hat{P}_{L_j}^t$ is the predicted load profile of the DL j at time t
- b) $\hat{P}_c^t = [\hat{P}_{c_j}^t, j = 1, \dots, n_c, t \in \mathbb{T}_{t_0}] \in \mathbb{R}^{n_c \times T}$, where $\hat{P}_{c_j}^t$ is the predicted load profile of the CL j at time t
- c) $\hat{P}_r^t = [\hat{P}_{r_j}^t, j = 1, \dots, n_p, t \in \mathbb{T}_{t_0}] \in \mathbb{R}^{n_p \times T}$, where $\hat{P}_{r_j}^t$ is the predicted power output of the j^{th} renewable energy source at time t .

D. Cost Function:

$$\mathcal{C}(\mathbf{x}, \mathbf{P}_g, \mathbf{P}_b, s_b, s_g) = \sum_{t=t_0}^{T+t_0-1} \left[\sum_{j=1}^{n_b} c_{b_j}(s_{b_j}^{t_0}, P_{b_j}^t) - \sum_{j=1}^{n_g} c_{g_j}(\eta_{g_j}(P_{g_j}^t)) + \sum_{j=1}^{n_d} c_{\ell_j}(1 - x_j^t) \hat{P}_{L_j}^t \right].$$

The first term captures the cost of dispatching the GFL battery inverters. $c_{b_j}(\cdot)$ is a convex function and ensures a cost-optimal charging/discharging dispatch of the GFL battery inverters over the horizon $[t_0, t_0 + T]$ based on the SOC of the GFL batteries. The second term captures the cost of dispatch for the diesel generators. The function $c_{g_j}(\cdot)$ is a concave function of the arguments $\mathbf{P}_g^{t_0}$ and $s_g^{t_0}$. The function c_{g_j} enforces that the diesel generators are dispatched over the horizon $[t_0, t_0 + T]$ at a higher efficiency set-point $\eta_{g_j}(P_{g_j}^t)$ for maximum utilization of the available stand-alone diesel generator storage. The last term generates an optimal portfolio of loads to be switched on (a DL j is *on* at time t if and only if $x_j^t = 1$) during the horizon $[t_0, t_0 + T]$ based on the load switched-off costs c_{ℓ_j} . The costs, c_{ℓ_j} , are decided based on the relative priorities of the DLs and the effect of switching the particular load off on the overall discomfort for the users in the microgrid. Together, the three terms in the cost function ensure that the CI units remain viable over the horizon. Further, the discomfort due to switching off a lot of DLs in the microgrid is also minimized.

E. Optimization Problem:

Let $\hat{s}_{b_j}^{t_0} := -(1 - s_{b_j}^{t_0})$,

$$\begin{aligned} \Lambda := \underset{\pi}{\text{minimize}} \quad & \mathcal{C}(\mathbf{x}, \mathbf{P}_g, \mathbf{P}_b, s_b^{t_0}, s_g^{t_0}), \quad \text{subject to :} \\ 0 \leq \sum_{t=t_0}^{T+t_0-1} P_{g_j}^t & \leq s_{g_j}^{t_0} E_{g_j}, \quad \text{for } j = 1, \dots, n_g, \end{aligned} \quad (4.29)$$

$$\hat{s}_{b_j}^{t_0} E_{b_j} \leq \sum_{t=t_0}^{T+t_0-1} P_{b_j}^t \leq s_{b_j}^{t_0} E_{b_j}, \quad \text{for } j = 1, \dots, n_b, \quad (4.30)$$

$$0 \leq P_{g_j}^t \leq W_{g_j}, \quad \text{for } j = 1, \dots, n_g, t \in \mathbb{T}_{t_0}, \quad (4.31)$$

$$-W_{b_j} \leq P_{b_j}^t \leq W_{b_j}, \quad \text{for } j = 1, \dots, n_b, t \in \mathbb{T}_{t_0} \quad (4.32)$$

$$0 \leq P_s^t \leq W_s, \quad \text{for } t \in \mathbb{T}_{t_0}, \quad (4.33)$$

$$\sum_{j=1}^{n_d} x_j^t \hat{P}_{L_j}^t + \sum_{j=1}^{n_c} \hat{P}_{c_j}^t = \sum_{j=1}^{n_g} P_{g_j}^t + \sum_{j=1}^{n_b} P_{b_j}^t + \sum_{j=1}^{n_p} \hat{P}_{r_j}^t + P_s^t, \quad \text{for } t \in \mathbb{T}_{t_0} \quad (4.34)$$

$$x_j^t \in \{0, 1\}, \quad \text{for } j = 1, \dots, n_d, \quad \text{for } t \in \mathbb{T}_{t_0}. \quad (4.35)$$

F. Constraints:

- i) (Available Energy Over the Horizon) Constraints (4.29) and (4.30) limit the maximum and minimum amount of total energy served to the loads during the horizon $[t_0, t_0 + T]$ based on the available energy in the stand-alone diesel generators and the GFL inverters at the start of the horizon.
- ii) (Generation) Constraints (4.31)-(4.33) enforce the maximum and minimum capacity limits on the power dispatch of the diesel generators, GFL inverters, and the slack bus.
- iii) (Power Balance) Constraint (4.34) ensures that the sum of all the power allocations to the critical loads and deferrable loads is equal to the available total generation.
- iv) (Load Selection) Constraint (4.35) is the binary constraint to decide the deferrable load switch-on-off status.

The solution, π^* , to the optimization problem, Λ , is utilized as the net-load dispatch decision for the microgrid containing the CI unit, with P_g^* and P_b^* being the diesel generator and the battery set points, respectively; P_{slk}^* being the slack battery power to be utilized; and x^* being the load switch-on-off decisions.

G. Heuristics to Improve the Run time of the HoV Optimization Problem:

Recall that the HoV optimization problem, Λ , due to the binary constraint (4.35), used for the load selection, is a mixed integer programming problem, which can be very hard to solve in general [268]. The solution time for the problem, Λ , can be improved by using the following heuristics:

1. The equality constraint (4.34) can be relaxed by specifying an operating band based on the power mismatch that can be safely tolerated by the slack bus. In particular, the constraint (4.34) can be equivalently written as a set of two inequality constraints: for $t \in \mathbb{T}_{t_0}$,

$$\begin{aligned} \underline{\delta} \left[\sum_{j=1}^{n_g} P_{g_j}^t + \sum_{j=1}^{n_b} P_{b_j}^t + \sum_{j=1}^{n_p} \hat{P}_{r_j}^t + P_s^t \right] &\leq \sum_{j=1}^{n_d} x_j^t \hat{P}_{L_j}^t + \sum_{j=1}^{n_c} \hat{P}_{c_j}^t \\ &\leq \bar{\delta} \left[\sum_{j=1}^{n_g} P_{g_j}^t + \sum_{j=1}^{n_b} P_{b_j}^t + \sum_{j=1}^{n_p} \hat{P}_{r_j}^t + P_s^t \right], \end{aligned}$$

where, $\bar{\delta} \in (0, 1]$, $\underline{\delta} \in (0, \bar{\delta}]$. The exact values of $\bar{\delta}$ and $\underline{\delta}$ can be chosen based on the application. Note that $\bar{\delta} = \underline{\delta} = 1$ recovers the original constraint (4.34).

2. Providing a good initial guess of the optimal solution to the optimization solver also improves the run time. A good initial solution for the problem, Λ , can be obtained by finding the optimal set of deferrable loads to be switched on at the starting time instant t_0 and assuming that the same set of deferrable loads remains operational throughout the horizon $[t_0, t_0 + T]$ based on the instantaneous power generation measurements $\{P_{g_j}^{t_0}, P_{b_j}^{t_0} \forall j\}$ from the microgrid. In particular, consider the simpler problem,

$$\begin{aligned} & \underset{x}{\text{minimize}} \quad \sum_{j=1}^{n_d} c_{\ell_j} (1 - x_j) \hat{P}_{L_j}^{t_0}, \\ & \text{subject to: } 0 \leq \sum_{j=1}^{n_d} x_j \hat{P}_{L_j}^{t_0} + \sum_{j=1}^{n_c} \hat{P}_{c_j}^{t_0} - \sum_{j=1}^{n_g} P_{g_j}^{t_0} - \sum_{j=1}^{n_b} P_{b_j}^{t_0} - \sum_{j=1}^{n_p} \hat{P}_{r_j}^{t_0} \leq W_s, \\ & \quad x_j \in \{0, 1\}, \text{ for } j = 1, \dots, n_d. \end{aligned}$$

The solution $x^t = x$ for $t \in \mathbb{T}_{t_0}$ can be taken as an initial solution to the variable $\mathbf{x}^{t_0}$ the in HoV optimization problem, Λ .

CHIL Demonstration and Results:

Figure 4.9 shows the three-phase microgrid test system under study that contains 1 CI unit and the surrounding 3 CB units. The ratings of various sources, loads, and line

Table 4.4: Ratings of Various Generation Sources, Loads, and Line Parameters of the Microgrid Under Study

Generation and Load (kW)	Microgrid Test System			
	CI Unit	CB Unit 1	CB Unit 2	CB Unit 3
PV	50	100	100	150
GFL Bat-1	100	-	-	-
GFL Bat-2	100	-	-	-
GFM Bat	-	200	250	200
Diesel Gen-1	50	-	-	-
Diesel Gen-2	50	-	-	-
Critical Load	230	100	225	150
Deferrable Load	350	300	275	350
Microgrid Network		Values		
Transformers		3- ϕ , Y-Y, 1 MVA, 13.8/0.48 kV		
Line Parameters		$R_{\text{line}} = 5 \text{ m}\Omega$, $L_{\text{line}} = 30 \text{ }\mu\text{H}$		

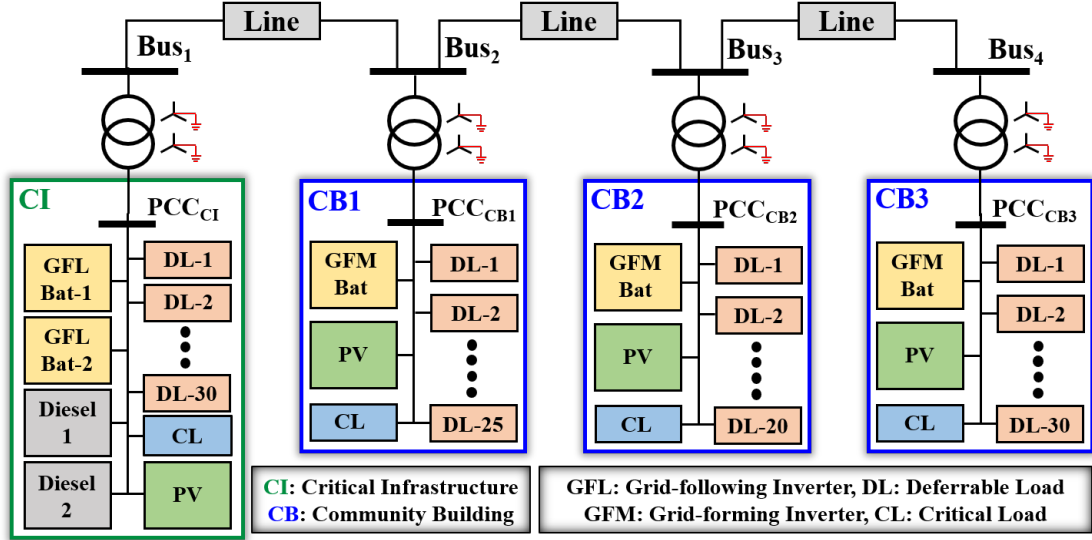


Figure 4.9: The microgrid test system contains 1 critical infrastructure (CI) unit and 3 community building (CB) units. The CI unit contains 2 GFL battery inverters, 2 diesel generators, 1 PV inverter system, 1 critical load (CL), and 30 deferrable loads (DLs). Similarly, CB-1, CB-2, and CB-3 contain 1 GFM battery inverter each, 1 PV inverter system each, 1 CL each, and 25, 20, and 30 DLs, respectively.

parameters are tabulated in Table 4.4. On top of the power layer, there is a communication layer that enables the transmission of various measurements, from the system to the HoV engine, and various control signals, from the HoV engine to the microgrid. The extensive list of measurements and control signals is shown in the schematic of the CHIL-based real time implementation of the HoV engine with the microgrid in Figure 4.10. There are three important components in the demonstration: (i) the microgrid, emulated using the eMEGASIM platform with a simulation step size of $50\mu s$ inside the OP5700 RT-simulator (RTS), which is manufactured by OPAL-RT and interfaced with two low-cost Texas Instruments TMS320F28379D, 16/12-bit floating-point 200-MHz Delfino microcontroller boards for controlling two GFM inverters in the CB-1 and CB-2 units; (ii) the HoV optimization engine running on a server workstation, developed using Python 3.7.1 with commercial optimization solver, MOSEK [269] to solve the mixed integer convex programming problem, Λ of Section 4.5.3; and (iii) a Device Control Gateway (DCG), realized using standard User Datagram Protocol (UDP) [263], that interfaces the microgrid and the HoV engine and emulates the communication layer that continuously listens to the measurements sent from the microgrid and

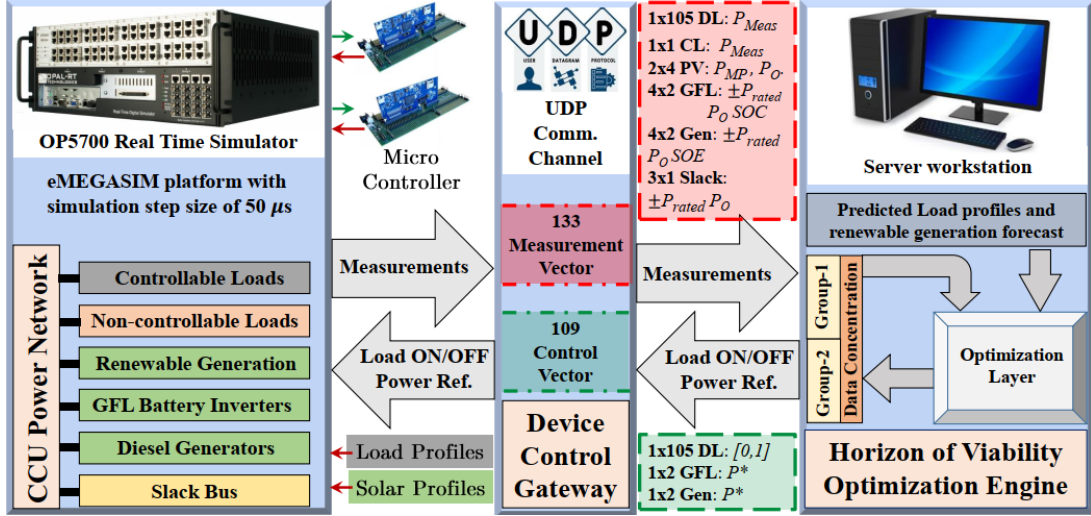


Figure 4.10: Schematic of the CHIL-based demonstration of the HoV optimization engine.

dispatch commands from the HoV engine. Realistic load profiles and the solar irradiance profile are fed into the real-time simulator with 1-s resolution. Most of the load profiles are obtained through the metering system by the energy management group of the University of Minnesota, and, the rest are synthetically generated. Real-world solar irradiance data for the study are obtained from the National Renewable Energy Laboratory's solar Measurement and Instrumentation Data Center and the National Solar Radiation Database [267]. A 2-hour experiment is conducted to validate the proposed HoV engine. Moreover, to show the efficacy, a comparative study is conducted with the net-load management (NLM) scheme existing in the literature [252].

To evaluate the performance of the developed HoV optimization engine, we define the following metrics:

1) *Total Remaining Energy (TRE)*: The TRE (in kWh) in the local generation sources in the microgrid (i.e, GFL battery inverters and diesel generators in the CI units of the microgrid) at the end of the time horizon $[0, \mathcal{T}]$ is defined as:

$$\text{TRE}(\mathcal{T}) := \sum_{j=1}^{n_g} s_{g_j}^{\mathcal{T}} E_{g_j} + \sum_{j=1}^{n_b} s_{b_j}^{\mathcal{T}} E_{b_j}. \quad (4.36)$$

2) *Cost-Weighted Energy (CWE)*: The total CWE delivered to all critical and deferrable

loads in the microgrid over a time horizon $[0, \mathcal{T}]$ is defined as:

$$\text{CWE}(\mathcal{T}) := \int_0^{\mathcal{T}} \sum_{i=1}^{n_d+n_c} c_{\ell_i} p_{L_i}^t dt, \quad (4.37)$$

where, $p_{L_i}^t$ is the per-unit power of load i .

The metrics in (4.36) and (4.37) capture the trade-off between the guaranteed viability of power for the CI units (higher remaining energy implies the availability of more energy in the generation resources of the CI units) and the level of service of the required CLs and DLs (higher CWE implies that more higher-priority loads are running). For comparison with the existing NLM scheme, we also define the ratios of the CWE and TRE metrics for the proposed HoV engine and the existing NLM engine. Specifically, we define the following:

$$\rho_{\text{CWE}}(t) := \frac{\text{CWE}_{\text{HoV}}(t)}{\text{CWE}_{\text{NLM}}(t)}, \quad (4.38)$$

$$\rho_{\text{TRE}}(t) := \frac{\text{TRE}_{\text{HoV}}(t)}{\text{TRE}_{\text{NLM}}(t)}. \quad (4.39)$$

Note that $n_g = 2$, $n_b = 2$, $n_d = 105$, $n_c = 1$, and $\mathcal{T} = 2$ hours (7200 s) for the CHIL experiment with the microgrid test system under study, as shown in Figure 4.9.

Figures 4.11-4.14 demonstrate the performance of the HoV engine compared to the NLM engine [252] with respect to the TRE and CWE metrics. Figure 4.11 provides a comparison based on the total remaining energy. The HoV optimization engine optimally schedules the available generation sources that result in a higher total remaining

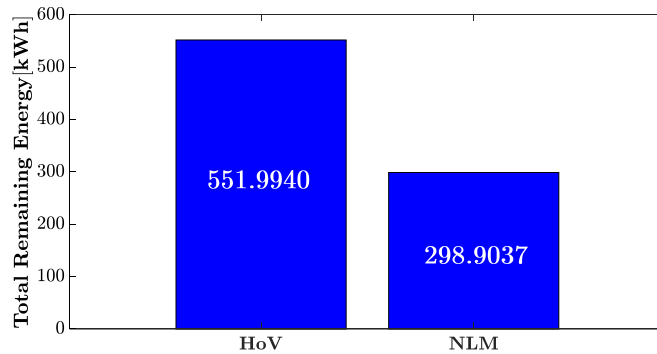


Figure 4.11: Total remaining energy (in kWh) after 2 hours of the CHIL experiment of the proposed HoV engine and existing NLM engine on the test system in Figure 4.9.

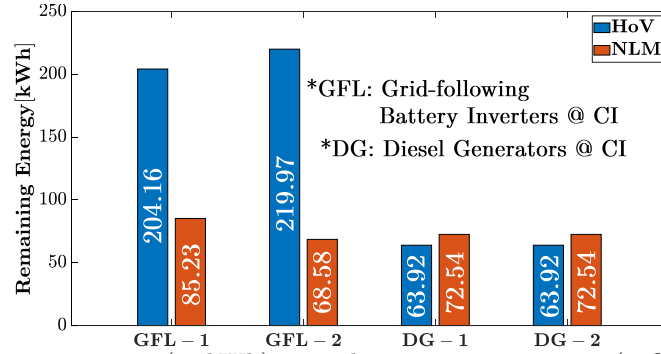


Figure 4.12: Remaining energy (in kWh) in each generating resource (2 GFL batteries and 2 diesel generators) in the CI unit after 2 hours of the CHIL experiment of the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.

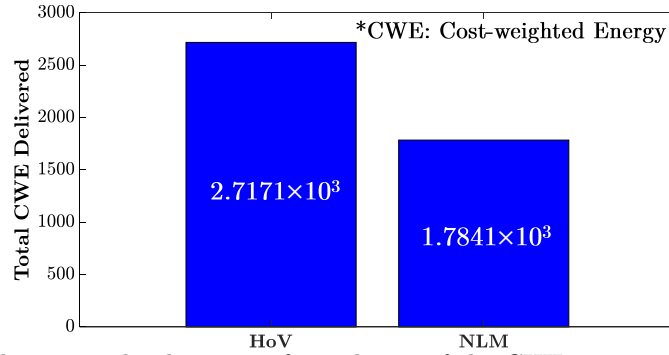


Figure 4.13: Total cost-weighted energy after 2 hours of the CHIL experiment of the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.

energy available at the end of 2 hours that can be utilized to serve the critical loads in the CI and CB units at a later time instant. The breakdown of the total energy remaining (among the 2 GFL batteries and 2 diesel generators) is presented in Figure 4.12. This shows the advantage of the proposed HoV engine over the existing NLM engine in the sense of extending the horizon of the viable operation of the CI units in a micro-grid. Figure 4.13 demonstrates that the HoV engine provides more total CWE than the NLM engine over the 2 hour horizon; therefore, the HoV engine provides power to more high-priority deferrable loads during the time duration of the experiment. Figure 4.14 shows that the total energy (kWh) delivered to the deferrable loads is more in the NLM dispatch; however the HoV engine better respects the load priorities than the NLM engine, as demonstrated by the benefit in terms of the CWE in Figure 4.13. This shows the benefit of the HoV engine in supplying power to the higher-prioritized loads in a resource-constrained scenario.

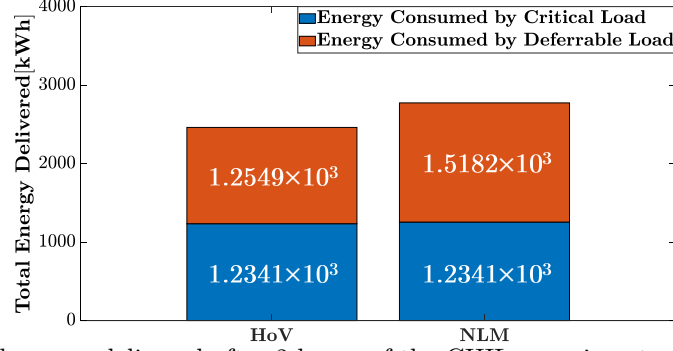


Figure 4.14: Total energy delivered after 2 hours of the CHIL experiment of the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.

The behavior of the HoV engine during the experiment is presented in Figures 4.15 and 4.16. Although the diesel generators are always operated at a high-efficiency set-point near the rated capacity, the HoV engine intelligently controls the GFL battery inverters based on the current SOC of the battery sources. It can be observed here that initially both the HoV and the NLM engine are inclined toward running more load in the microgrid, and, as a result, the power set points are at the rated peak for both GFL batteries. Unlike the NLM engine, however, the HoV engine moves to the GFL battery-saving mode by generating charging commands for both GFL batteries after an instance when the SOC of the GFL batteries fall below 50%. Therefore, initially, the HoV engine is aggressive in terms of maintaining a good quality of service to the load, but it adjusts itself to conserve generation for future times of operation. The NLM engine, on the other hand, does not adjust itself to address future contingencies

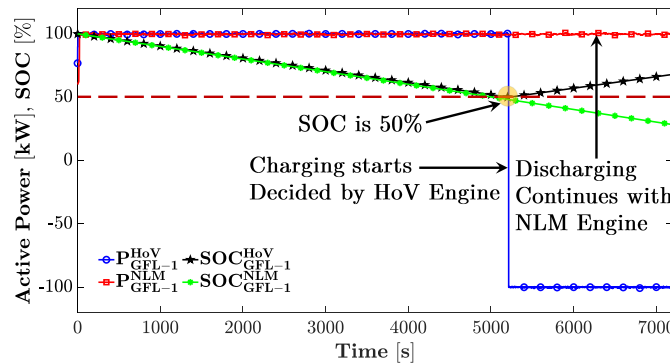


Figure 4.15: Active power output and SOC of the GFL battery-1 in the CI unit during 2 hours of the CHIL experiment with the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.

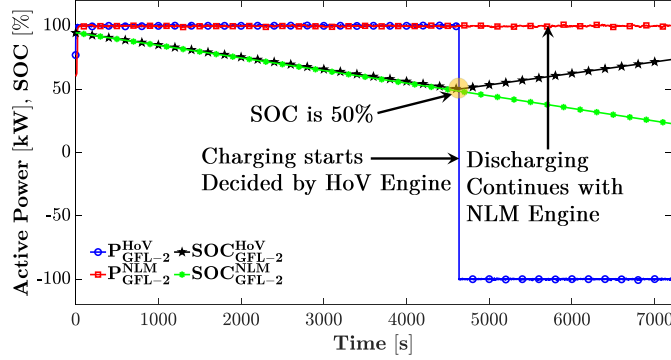


Figure 4.16: Active power output and SOC of the GFL battery-2 in the CI unit during 2 hours of the CHIL experiment with the proposed HoV engine and the existing NLM engine on the test system in Figure 4.9.

of power delivery to the CI units, which is demonstrated by a lower SOC in the GFL batteries (Figures 4.15 and 4.16) and consequently lower total energy remaining in the generation sources (Figure 4.11 and Figure 4.12).

This behavior of the proposed HoV engine and the NLM engine can be further corroborated by referring to Figure 4.17 and Figure 4.18. These two figures show the run time ratios of the CWE and TRE metrics (ρ_{CWE} and ρ_{TRE}), defined in (4.38) and (4.39). It can be observed that the proposed HoV engine is equally conservative in restoring the energy of the GFL batteries in the CI unit but more aggressive in terms of supplying a higher number of prioritized loads in the microgrid compared to the NLM engine (indicated by $\rho_{\text{CWE}} > 1$ and $\rho_{\text{TRE}} \approx 1$). Yet, toward the end of the horizon, the HoV engine is more aggressive in restoring the energy of the GFL batteries in the CI unit but equally aggressive in terms of supplying prioritized loads in the microgrid compared to the NLM engine (indicated by $\rho_{\text{CWE}} \approx 1$ and $\rho_{\text{TRE}} > 1$).

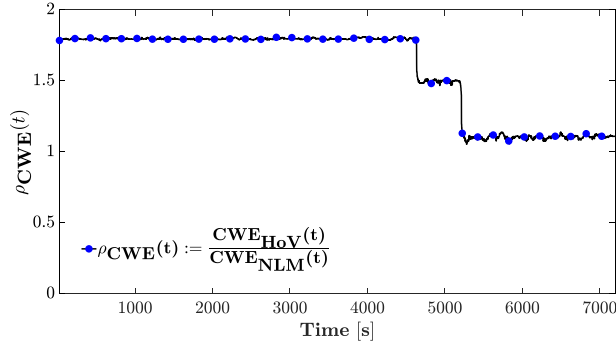


Figure 4.17: Ratio of the CWE metric value resulting from HoV engine and NLM engine during 2 hours of CHIL experiment on the test system of Figure 4.9.

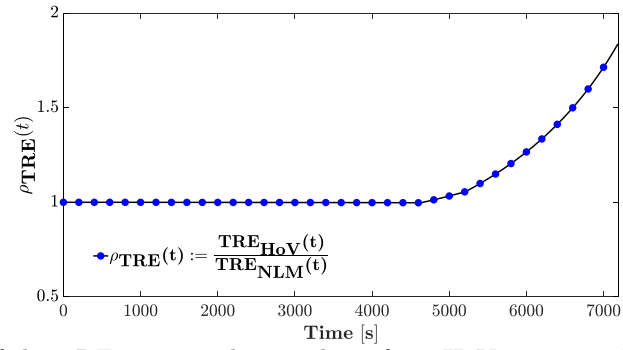


Figure 4.18: Ratio of the TRE metric value resulting from HoV engine and NLM engine during 2 hours of CHIL experiment on the test system of Figure 4.9.

Chapter 5

Conclusion And Future Work

5.1 Concluding Remarks

This thesis makes significant contributions to advancing robust and efficient distributed decision-making in networked dynamical systems (NDS). By addressing challenges in scalability, resilience, and real-time adaptability, it introduces innovative solutions with the potential to transform distributed methods under practical constraints. The work bridges theoretical advancements with practical implementations, ensuring broad applicability across diverse fields, including energy systems. Notably, it supports the integration of renewable energy sources and enhances the reliability of modern power networks, contributing to a sustainable and resilient energy future.

In Chapter 2, we introduced the monotonic property of the convex hull of network states in high-dimensional consensus algorithms. This property was utilized to develop a finite-time stopping criterion and design a distributed algorithm. Additionally, we proposed an algorithm to approximate minimum-norm balls enclosing all network states at a given iteration, demonstrating that their radii converge to zero. These radii were then employed to formulate a finite-time stopping criterion, significantly reducing communication overhead compared to existing methods. The effectiveness of the proposed approach was validated through simulations of a vector ratio consensus algorithm with states in $\mathbb{R}^{10}$ on a network graph comprising 25 nodes.

Furthermore, in Chapter 2, we presented a protocol for distributed finite-time termination of consensus algorithms in networks with dynamic interconnection topology. A key contribution was the introduction of the novel concept of a “time path,” which

enables analysis of how an agent influences others in the network. We proved the existence of finite-length time paths and established the strict monotonicity of the global maximum and minimum of the state values for each node after every n' iterations, where n' is an upper bound on the number of nodes. Building on this, we proposed a new maximum-minimum protocol tailored for dynamic topologies and used it to design the distributed finite-time termination algorithm. We also highlighted the limitations of existing finite-time termination algorithms, which rely on static topologies, in real-time applications such as *ad hoc* cognitive radio networks and autonomous agent control. The effectiveness of our algorithm in overcoming these challenges was demonstrated through experimental implementation on a dynamically changing network topology using a Node.js framework.

In addition, Chapter 2, tackles the challenge of designing an average consensus algorithm for directed graphs with communication links subjected to noise. We demonstrated that the widely used PushSum algorithm, commonly applied in distributed decision-making over directed graphs, performs poorly under noisy communication conditions. To overcome this limitation, we introduced the **NR-PushSum** algorithm. This approach allows each agent to update its estimates using its noise-free, weighted initial state value while appropriately weighting information received from neighbors to account for noisy communication links. These adaptive weights enable agents to mitigate the impact of noisy estimates from neighbors and achieve consensus. We proved that the **NR-PushSum** algorithm achieves a geometric rate of convergence under noiseless communication and ensures *almost sure* convergence to a consensus value under additive noise. Numerical case studies under various noise scenarios and algorithm parameters validated the effectiveness of the **NR-PushSum** algorithm. Simulation results showcased its superior performance compared to existing state-of-the-art methods, emphasizing its robustness and reliability in noisy environments.

In Chapter 3, we address the problem of distributedly minimizing the sum of n convex functions over directed multi-agent networks. We introduce the *optimize then agree* framework, which decouples the optimization and consensus steps to enhance convergence and feasibility by reducing the mismatch between agents' solution estimates. Building on this framework, we developed the novel **GradConsensus** algorithm, where each agent performs a gradient-descent update during the optimization step and employs a finite-time ε -consensus protocol to achieve practical agreement among estimates at every iteration. We provide explicit convergence rate estimates for the **GradConsensus**

algorithm under two distinct sets of assumptions on the aggregate objective functions. Notably, we demonstrated that the iterates generated by the **GradConsensus** algorithm converge to the optimal solution (or its neighborhood) at a linear rate under these conditions. To validate the algorithm, we applied **GradConsensus** to solve a distributed logistic regression problem, demonstrating its effectiveness in handling distributed optimization tasks. Furthermore, the *optimize then agree* framework is versatile, offering guidelines for enhancing existing algorithms and inspiring the development of new methods to solve distributed optimization problems efficiently.

In the second part of Chapter 3, we developed the **DC-DistADMM** algorithm to address constrained multi-agent optimization problems with local linear equality, inequality, and set constraints over directed graphs. This algorithm represents the first ADMM-based solution capable of handling both constrained and unconstrained distributed optimization problems in such settings. Unlike traditional ADMM approaches, the **DC-DistADMM** algorithm replaces the computationally demanding primal variable optimization step with a more efficient ε -consensus protocol, making it particularly suited for distributed multi-agent systems. The **DC-DistADMM** algorithm is designed for distributed implementation and synthesis, combining techniques from Lagrangian dual-based optimization with methods from average consensus literature. Each agent solves a decentralized constrained optimization problem and updates its estimate of the optimal solution using the finite-time ε -consensus algorithm. The **DC-DistADMM** algorithm offers strong convergence guarantees, including (i) a rate of $O(1/k)$ for convergence when the individual functions are convex but not necessarily differentiable, (ii) geometric convergence to a small neighborhood of the optimal solution when the objective functions are smooth and restricted strongly convex at the optimal solution, (iii) numerical simulations validated the **DC-DistADMM** algorithm's performance against state-of-the-art methods for constrained and unconstrained distributed optimization problems. Additionally, we compared its efficiency with existing algorithms that rely on multiple consensus steps, highlighting the **DC-DistADMM** algorithm's computational advantages and robust convergence properties.

Chapter 4 extends the algorithmic foundations laid in Chapters 2 and 3 to address decision-making challenges in modern power systems. Specifically, it focuses on the secondary control problem in microgrids with heterogeneous grid-forming inverter-based

resources (GFM IBRs) connected via varying line impedances. Departing from traditional approaches, we proposed a fully distributed controller that supports both distributed implementation and synthesis using only locally available information. Key features of the developed scheme include (i) Plug-and-play Operation: The controller accommodates dynamic changes in the network without requiring centralized updates, (ii) Versatility: It achieves diverse control objectives, making it adaptable to various operational scenarios, (iii) Privacy-Preserving Updates: Information exchange is limited to essential data, safeguarding system privacy. We conducted a convergence analysis, demonstrating the controller’s ability to ensure voltage regulation and reactive power sharing under steady-state conditions in GFM networks. To validate the proposed method, Controller-Hardware-in-the-Loop (CHIL) experimental studies were performed. These experiments confirmed the effectiveness and reliability of the distributed controller for secondary control in microgrids.

In another distributed decision-making scenario for microgrids, Chapter 4 introduces a resilient framework for the aggregation of Distributed Energy Resources (DERs) to provide ancillary service support to the external power network. A key component of this framework is a distributed malicious behavior detection algorithm, enabling DERs to identify and isolate compromised communication links within the microgrid. We demonstrated that the proposed scheme effectively detects maliciously behaving DERs, ensuring the integrity of the aggregation process. The algorithm’s performance was rigorously tested under various attack scenarios using Controller-Hardware-in-the-Loop (CHIL) experiments. Results showed that DERs successfully identified malicious behavior within seconds in all tested attack scenarios, highlighting the scheme’s robustness and effectiveness in maintaining secure and reliable microgrid operations.

In response to the growing need for reliable and sustained power delivery to critical infrastructure (CI) units in microgrids during catastrophic events, Chapter 4 introduces an intelligent optimization scheme termed the Horizon of Viability (HoV) engine. This innovative framework ensures sustained power supply to CI units over a defined time horizon following a power grid failure. The HoV engine employs a mixed-integer convex programming approach to generate a cost-optimal portfolio of locally available generation sources and loads, ensuring operational efficiency and reliability. Its control performance was rigorously evaluated through Controller-Hardware-in-the-Loop (CHIL) experiments. The results demonstrated the HoV engine’s superior ability to maintain the viability of CI units post-grid interruption, outperforming existing net-load management schemes in the literature.

5.2 Future Directions of Research

In the previous section, we outlined this thesis’s contributions to establishing a framework for distributed decision-making and devising methods for the aggregation and control of DERs in power systems to enable a clean and efficient modern power grid. In the following, we present a summary of open research topics that merit further exploration.

- A promising avenue for future research lies in extending the **NR-PushSum** algorithm to design distributed optimization algorithms that operate effectively on directed graphs under communication noise. Current state-of-the-art methods often fall short in addressing practical challenges such as noise, delays, and the inherent asymmetry of directed communication. Incorporating the robustness of the **NR-PushSum** algorithm into optimization frameworks offers the potential for more resilient and efficient distributed decision-making in NDS. Further theoretical investigation into the convergence properties and performance bounds of such algorithms, particularly under varying levels of noise and delay, would provide deeper insights into their scalability and applicability. Adapting the **NR-PushSum** algorithm to handle dynamic graph topologies would also be a valuable contribution, with practical implications for applications like mobile sensor networks and autonomous vehicle fleets. Finally, large-scale simulations and real-world implementations are essential to validate the algorithm’s efficacy and bridge the gap between theoretical development and practical deployment, making the **NR-PushSum** algorithm a versatile tool for advancing distributed decision-making in complex systems.
- Another significant extension of the research presented in this thesis is adapting the *optimize-then-agree* framework and the proposed **GradConsensus** and **DC-DistADMM** algorithms to operate under time-varying directed topologies with communication delays. Addressing these complexities would broaden the applicability of the framework to dynamic, real-world systems such as sensor networks, autonomous vehicle coordination, and smart grids. A critical research direction involves establishing theoretical guarantees of convergence and stability for the algorithms under such conditions, ensuring their reliability and robustness in dynamic environments. Additionally, devising efficient strategies to counteract the adverse effects of delays and topology variations while maintaining computational

and communication efficiency is essential for practical viability. Validating the extended framework through large-scale simulations and real-world deployments would further underscore its robustness and effectiveness, paving the way for its application in increasingly complex distributed decision-making scenarios.

- I am currently developing distributed algorithms for optimization problems where objective functions are expressed as the difference of two convex functions (DC programming). These problems frequently arise in nonconvex decision-making scenarios, particularly in fields such as statistics, machine learning, and signal processing. An early version of this work was recently accepted for publication at the Annual Allerton Conference [270], marking a key milestone in this research. The proposed algorithms are designed to address the challenges of scalability and efficiency in solving DC problems in distributed settings. Future directions include enhancing their robustness against communication constraints such as noise and delays, as well as extending their applicability to dynamic and heterogeneous networks. I also plan to investigate their integration into practical applications like federated learning and distributed resource allocation to validate their real-world effectiveness. Moreover, I am actively exploring collaborative opportunities to improve the algorithms' convergence guarantees and computational efficiency, ensuring they remain at the forefront of distributed DC optimization research.
- The distributed controller presented in Chapter 4 can be further enhanced to address practical challenges such as communication delays, dynamic interconnections among GFM DERs, and variations in line impedances. A promising research direction involves performing a closed-loop stability analysis of the GFM and GFL DER network using a sampled-data controller, grounded in discrete-time distributed optimization algorithms. This line of investigation is currently being pursued by the author. Resilience enhancements for the distributed apportioning algorithm are another focus, particularly to counteract sophisticated cyber-attack scenarios, including data falsification, denial-of-service, and coordinated multi-point attacks. Incorporating adaptive control mechanisms to mitigate dynamic changes in system parameters and network topology variations is also a critical avenue for development. Additional research involves evaluating the controller's performance under stochastic disturbances and uncertainties in system modeling, which is crucial for its robustness. Experimental validation in real-world micro-grid setups could provide practical insights, highlighting deployment challenges

and solutions. Ongoing collaborations aim to develop scalable implementations of the controller tailored for large-scale distributed energy resource networks. Finally, extending the theoretical framework to hybrid AC/DC microgrids could significantly broaden the scope and applicability of the proposed methodology, addressing the evolving needs of modern power systems.

In conclusion, distributed decision-making in networked dynamical systems poses numerous challenging questions that demand innovative approaches. This thesis tackles several critical aspects of these challenges, offering foundational insights and methodologies to push the boundaries of the field. The author aspires that the research presented not only addresses some of these pressing questions but also establishes a robust foundation for diverse future investigations.

Promising directions for further research include improving the scalability of distributed algorithms to handle large, complex networks and developing methods to account for uncertainties and stochastic behaviors inherent in real-world systems. Enhancing communication efficiency and reliability through advanced protocols in dynamic and resource-constrained environments presents another compelling avenue. Additionally, integrating machine learning techniques into adaptive decision-making processes and optimizing coordination across multi-layered networks could pave the way for significant advancements.

The author envisions this thesis as a stepping stone toward the development of robust, efficient, and resilient distributed systems, capable of meeting the demands of increasingly interconnected and dynamic environments.

Bibliography

- [1] R. Diestel. *Graph Theory*. Springer-Verlag, Berlin, Germany, 2006.
- [2] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.
- [3] Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.
- [4] Xuyang Wu, Zhihai Qu, and Jie Lu. A second-order proximal algorithm for consensus optimization. *IEEE Transactions on Automatic Control*, 66(4):1864–1871, 2020.
- [5] Council on Foreign Relations. What is the u.s. federal reserve? <https://www.cfr.org/backgroundunder/what-us-federal-reserve>. [Online; accessed 21-Jan-2025].
- [6] Stanford University. The grid: Electricity transmission, industry, and markets. <https://understand-energy.stanford.edu/energy-currencies/electricity-grid>. [Online; accessed 21-Jan-2025].
- [7] Harvard Business Review. The triple-a supply chain. <https://hbr.org/2004/10/the-triple-a-supply-chain>. [Online; accessed 21-Jan-2025].
- [8] Forbes. The billion-dollar ai gamble: Data centers as the new high-stakes game. <https://www.forbes.com/sites/emilsayegh/2024/09/30/the-billion-dollar-ai-gamble-data-centers-as-the-new-high-stakes-game/>. [Online; accessed 21-Jan-2025].
- [9] Cybersecurity and Infrastructure Security Agency. Tcyber-attack against ukrainian critical infrastructure. <https://www.cisa.gov/news-events/ics-alerts/ir-alert-h-16-056-01>. [Online; accessed 21-Jan-2025].

- [10] Joshua W Busby, Kyri Baker, Morgan D Bazilian, Alex Q Gilbert, Emily Grubert, Varun Rai, Joshua D Rhodes, Sarang Shidore, Caitlin A Smith, and Michael E Webber. Cascading risks: Understanding the 2021 winter blackout in texas. *Energy Research & Social Science*, 77:102106, 2021.
- [11] The Department of Forestry and Fire Protection. Current emergency incidents. <https://www.fire.ca.gov/incidents>. [Online; accessed 21-Jan-2025].
- [12] Cyber-Physical Systems (CPS), NSF 24-581, 2024. [December 20, 2024].
- [13] Cyber-Physical Systems Summit, 2024. [December 20, 2024].
- [14] Kenneth Joseph Arrow and Leonid Hurwicz. *Decentralization and computation in resource allocation*. Stanford University, Department of Economics, 1958.
- [15] Nancy A Lynch. *Distributed algorithms*. Elsevier, 1996.
- [16] Sourav Patel, Sandeep Attree, Saurav Talukdar, Mangal Prakash, and Murti V Salapaka. Distributed apportioning in a power network for providing demand response services. In *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 38–44. IEEE, 2017.
- [17] Sourav Patel, Blake Lundstrom, Govind Saraswat, and Murti V Salapaka. Distributed power apportioning with early dispatch for ancillary services in renewable grids. *arXiv preprint arXiv:2007.11715*, 2020.
- [18] Sourav Patel, Vivek Khatana, Govind Saraswat, and Murti V Salapaka. Distributed detection of malicious attacks on consensus algorithms with applications in power networks. In *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*, volume 1, pages 397–402. IEEE, 2020.
- [19] Maher Ala’raj and Maysam F Abbod. Classifiers consensus system approach for credit scoring. *Knowledge-Based Systems*, 104:89–105, 2016.
- [20] A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.
- [21] Angelia Nedic. Distributed gradient methods for convex machine learning problems in networks: Distributed optimization. *IEEE Signal Processing Magazine*, 37(3):92–101, 2020.

- [22] Morris H DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.
- [23] Gianni Pasolini, Davide Dardari, and Michel Kieffer. Exploiting the agent’s memory in asymptotic and finite-time consensus over multi-agent networks. *IEEE Transactions on Signal and Information Processing over Networks*, 6:479–490, 2020.
- [24] Zhiqiang Li, F Richard Yu, and Minyi Huang. A distributed consensus-based cooperative spectrum-sensing scheme in cognitive radios. *IEEE Transactions on Vehicular Technology*, 59(1):383–393, 2009.
- [25] Joel B Predd, Sanjeev R Kulkarni, and H Vincent Poor. A collaborative training algorithm for distributed learning. *IEEE Transactions on Information Theory*, 55(4):1856–1871, 2009.
- [26] Ali Jadbabaie, Jie Lin, et al. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *Automatic Control, IEEE Transactions on*, 48(6):988–1001, 2003.
- [27] J Alexander Fax and Richard M Murray. Information flow and cooperative control of vehicle formations. *IFAC Proceedings Volumes*, 35(1):115–120, 2002.
- [28] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.
- [29] John Nikolas Tsitsiklis. Problems in decentralized decision making and computation. Technical report, DTIC Document, 1984.
- [30] Vivek Borkar and Pravin Varaiya. Asymptotic agreement in distributed estimation. *IEEE Transactions on Automatic Control*, 27(3):650–655, 1982.
- [31] John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812, 1986.
- [32] Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 20–27. ACM, 2004.

- [33] Alex Olshevsky. Efficient information aggregation strategies for distributed control and signal processing. *arXiv preprint arXiv:1009.6036*, 2010.
- [34] Climate Central. Surging power outages and climate change. <https://www.climatecentral.org/climate-matters/>. [Online; accessed 27-Sept-2023].
- [35] The Washington Post. After hurricane maria, puerto rico was in the dark for 181 days, 6 hours and 45 minutes. <https://www.washingtonpost.com/graphics/2017/national/puerto-rico-hurricane-recovery/>. [Online; accessed 27-Sept-2023].
- [36] Soham Chakraborty. *Robust Dynamic Resilient Power Grids Enabled by Modern Control Framework*. PhD thesis, 2023. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2024-10-09.
- [37] Josep M Guerrero, Juan C Vasquez, José Matas, Luis García De Vicuña, and Miguel Castilla. Hierarchical control of droop-controlled ac and dc microgrids—a general approach toward standardization. *IEEE Transactions on industrial electronics*, 58(1):158–172, 2010.
- [38] Vivek Khatana and Murti V. Salapaka. Noise resilient distributed average consensus over directed graphs. *IEEE Transactions on Signal and Information Processing over Networks*, pages 1–15, 2023.
- [39] James Melbourne, Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V. Salapaka. Convex decreasing algorithms: Distributed synthesis and finite-time termination in higher dimension. *IEEE Transactions on Automatic Control*, pages 1–8, 2023.
- [40] James Melbourne, Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V. Salapaka. On the geometry of consensus algorithms with application to distributed termination in higher dimension. *IFAC-PapersOnLine*, 53(2):2951–2956, 2020. 21st IFAC World Congress.
- [41] Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V. Salapaka. Distributed finite-time termination for consensus algorithm in switching topologies. *IEEE Transactions on Network Science and Engineering*, pages 1–11, 2022.

- [42] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491. IEEE, 2003.
- [43] Christoforos N Hadjicostis and Themistoklis Charalambous. Average consensus in the presence of delays and dynamically changing directed graph topologies. *arXiv preprint arXiv:1210.4778*, 2012.
- [44] Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V Salapaka. Distributed finite-time termination for consensus algorithm in switching topologies. *IEEE Transactions on Network Science and Engineering*, 2022.
- [45] Christoforos N Hadjicostis and Themistoklis Charalambous. Average consensus in the presence of delays in directed graph topologies. *IEEE Transactions on Automatic Control*, 59(3):763–768, 2013.
- [46] Mangal Prakash, Saurav Talukdar, Sandeep Attree, Vikas Yadav, and Murti V. Salapaka. Distributed stopping criterion for consensus in the presence of delays. *IEEE Transactions on Control of Network Systems*, 7(1):85–95, 2020.
- [47] James Melbourne, Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V Salapaka. On the geometry of consensus algorithms with application to distributed termination in higher dimension. *IFAC-PapersOnLine*, 53(2):2951–2956, 2020.
- [48] James Melbourne, Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V Salapaka. Convex decreasing algorithms: Distributed synthesis and finite-time termination in higher dimension. *arXiv preprint arXiv:2007.13050*, 2020.
- [49] R. Olfati-Saber. Distributed kalman filtering for sensor networks. In *Proc. of IEEE CDC*, pages 5492–5498, New Orleans, 2007.
- [50] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48, 2009.
- [51] Alex Olshevsky and John N Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization*, 48(1):33–55, 2009.
- [52] Kai Cai and Hideaki Ishii. Average consensus on general strongly connected digraphs. *Automatica*, 48(11):2750–2761, 2012.

- [53] Ali Jadbabaie, Nader Motee, and Mauricio Barahona. On the stability of the kuramoto model of coupled nonlinear oscillators. In *Proceedings of the 2004 American Control Conference*, volume 5, pages 4296–4301. IEEE, 2004.
- [54] Antonis Papachristodoulou and Ali Jadbabaie. Synchronization in oscillator networks: Switching topologies and non-homogeneous delays. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 5692–5697. IEEE, 2005.
- [55] Nikhil Chopra and Mark W Spong. On synchronization of kuramoto oscillators. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 3916–3922. IEEE, 2005.
- [56] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. Technical report, CALIFORNIA INST OF TECH PASADENA CONTROL AND DYNAMICAL SYSTEMS, 2004.
- [57] Andrey V Savkin. Coordinated collective motion of groups of autonomous mobile robots: Analysis of vicsek’s model. *IEEE Transactions on Automatic Control*, 49(6):981–982, 2004.
- [58] Nima Moshtagh and Ali Jadbabaie. Distributed geodesic control laws for flocking of nonholonomic agents. *IEEE Transactions on Automatic Control*, 52(4):681–686, 2007.
- [59] Reza Olfati-Saber and Jeff S Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6698–6703. IEEE, 2005.
- [60] Vijay Gupta, Babak Hassibi, and Richard M Murray. On sensor fusion in the presence of packet-dropping communication channels. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 3547–3552. IEEE, 2005.
- [61] Reza Olfati-Saber and Richard M Murray. Distributed cooperative control of multiple vehicle formations using structural potential functions. *IFAC Proceedings Volumes*, 35(1):495–500, 2002.
- [62] Rodney Teo, Dusan M Stipanovic, and Claire J Tomlin. Decentralized spacing control of a string of multiple vehicles over lossy datalinks. *IEEE Transactions on control systems technology*, 18(2):469–473, 2009.

- [63] Alejandro D Dominguez-Garcia and Christoforos N Hadjicostis. Coordination and control of distributed energy resources for provision of ancillary services. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 537–542. IEEE, 2010.
- [64] Reza Olfati-Saber and Richard M Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on automatic control*, 49(9):1520–1533, 2004.
- [65] Vincent Blondel, Julien M Hendrickx, Alex Olshevsky, J Tsitsiklis, et al. Convergence in multiagent coordination, consensus, and flocking. In *IEEE Conference on Decision and Control*, volume 44, page 2996. IEEE; 1998, 2005.
- [66] Themistoklis Charalambous and Christoforos N Hadjicostis. Average consensus in the presence of dynamically changing directed topologies and time delays. In *53rd IEEE Conference on Decision and Control*, pages 709–714. IEEE, 2014.
- [67] Shuai Liu, Lihua Xie, and Huanshui Zhang. Distributed consensus for multi-agent systems with delays and noises in transmission channels. *Automatica*, 47(5):920–934, 2011.
- [68] Tao Li and Ji-Feng Zhang. Mean square average-consensus under measurement noises and fixed topologies: Necessary and sufficient conditions. *Automatica*, 45(8):1929–1936, 2009.
- [69] Lin Xiao, Stephen Boyd, and Seung-Jean Kim. Distributed average consensus with least-mean-square deviation. *Journal of parallel and distributed computing*, 67(1):33–46, 2007.
- [70] Ruggero Carli, Giacomo Como, Paolo Frasca, and Federica Garin. Average consensus on digital noisy networks. *IFAC Proceedings Volumes*, 42(20):36–41, 2009.
- [71] Tingting Pan, Lipo Mo, and Xianbing Cao. Consensus of discrete-time multi-agent systems with white noise disturbance. *IFAC-PapersOnLine*, 48(28):202–205, 2015.
- [72] Sivaraman Dasarathan, Cihan Tepedelenlioğlu, Mahesh K Banavar, and Andreas Spanias. Robust consensus in the presence of impulsive channel noise. *IEEE Transactions on Signal Processing*, 63(8):2118–2129, 2015.

- [73] Boyuan Li, Henry Leung, and Chatura Seneviratne. Analysis of noise impact on distributed average consensus. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXVII*, volume 10646, page 106460L. International Society for Optics and Photonics, 2018.
- [74] Soumya Kar and José MF Moura. Distributed average consensus in sensor networks with random link failures and communication channel noise. In *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, pages 676–680. IEEE, 2007.
- [75] Soumya Kar and José MF Moura. Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise. *IEEE Transactions on Signal Processing*, 57(1):355–369, 2008.
- [76] Loreto Pescosolido, Sergio Barbarossa, and Gesualdo Scutari. Average consensus algorithms robust against channel noise. In *2008 IEEE 9th Workshop on Signal Processing Advances in Wireless Communications*, pages 261–265. IEEE, 2008.
- [77] Tao Li and Ji-Feng Zhang. Consensus conditions of multi-agent systems with time-varying topologies and stochastic communication noises. *IEEE Transactions on Automatic Control*, 55(9):2043–2057, 2010.
- [78] Ram Rajagopal and Martin J Wainwright. Network-based consensus averaging with general noisy channels. *IEEE Transactions on Signal Processing*, 59(1):373–385, 2010.
- [79] Ioannis D Schizas, Alejandro Ribeiro, and Georgios B Giannakis. Consensus in ad hoc wsns with noisy links—part i: Distributed estimation of deterministic signals. *IEEE Transactions on Signal Processing*, 56(1):350–364, 2007.
- [80] Ioannis D Schizas, Georgios B Giannakis, Stergios I Roumeliotis, and Alejandro Ribeiro. Consensus in ad hoc wsns with noisy links—part ii: Distributed estimation and smoothing of random signals. *IEEE Transactions on Signal Processing*, 56(4):1650–1666, 2008.
- [81] Minyi Huang and Jonathan H Manton. Coordination and consensus of networked agents with noisy measurements: Stochastic algorithms and asymptotic behavior. *SIAM Journal on Control and Optimization*, 48(1):134–161, 2009.

- [82] Alain Y Kibangou. Finite-time average consensus based protocol for distributed estimation over awgn channels. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pages 5595–5600. IEEE, 2011.
- [83] Mengjie Zhou, Jianping He, Peng Cheng, and Jiming Chen. Discrete average consensus with bounded noise. In *52nd IEEE Conference on Decision and Control*, pages 5270–5275. IEEE, 2013.
- [84] Jianping He, Mengjie Zhou, Peng Cheng, Ling Shi, and Jiming Chen. Consensus under bounded noise in discrete network systems: An algorithm with fast convergence and high accuracy. *IEEE Transactions on Cybernetics*, 46(12):2874–2884, 2016.
- [85] Francisco FC Rego, Ye Pu, Andrea Alessandretti, A Pedro Aguiar, and Colin N Jones. A consensus algorithm for networks with process noise and quantization error. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 488–495. IEEE, 2015.
- [86] Ali Jadbabaie and Alex Olshevsky. On performance of consensus protocols subject to noise: Role of hitting times and network structure. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 179–184. IEEE, 2016.
- [87] Ge Chen, Chen Chen, George Yin, et al. Critical connectivity and fastest convergence rates of distributed consensus with switching topologies and additive noises. *IEEE Transactions on Automatic Control*, 62(12):6152–6167, 2017.
- [88] Oleg Granichin, Victoria Erofeeva, Yury Ivanskiy, and Yuming Jiang. Simultaneous perturbation stochastic approximation-based consensus for tracking under unknown-but-bounded disturbances. *IEEE Transactions on Automatic Control*, 66(8):3710–3717, 2020.
- [89] Yasin Yazıcıoğlu and Alberto Speranzon. High dimensional robust consensus over networks with limited capacity. *IEEE Control Systems Letters*, 5(6):2024–2029, 2020.
- [90] Behrouz Touri and Angelia Nedic. Distributed consensus over network with noisy links. In *2009 12th International Conference on Information Fusion*, pages 146–154. IEEE, 2009.

- [91] Jing Wang and Nicola Elia. Distributed agreement in the presence of noise. In *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1575–1581. IEEE, 2009.
- [92] Jing Wang and Nicola Elia. Dynamic average consensus over random networks with additive noise. In *49th IEEE Conference on Decision and Control (CDC)*, pages 4789–4794. IEEE, 2010.
- [93] Jing Wang and Nicola Elia. Distributed averaging algorithms resilient to communication noise and dropouts. *IEEE transactions on signal processing*, 61(9):2231–2242, 2013.
- [94] Shi Pu and Alfredo Garcia. A flocking-based approach for distributed stochastic optimization. *Operations Research*, 66(1):267–281, 2018.
- [95] Yushen Long, Shuai Liu, and Lihua Xie. Distributed consensus of discrete-time multi-agent systems with multiplicative noises. *International Journal of Robust and Nonlinear Control*, 25(16):3113–3131, 2015.
- [96] SI Sheipak. Reaching consensus in a variable-topology multiagent system under additive random noise. *Automation and Remote Control*, 81(5):911–921, 2020.
- [97] Yunpeng Wang, Long Cheng, Zeng-Guang Hou, Min Tan, Chao Zhou, and Ming Wang. Consensus seeking in a network of discrete-time linear agents with communication noises. *International Journal of Systems Science*, 46(10):1874–1888, 2015.
- [98] Zhongmei Wang, Wei Wang, and Huanshui Zhang. Robust consensus for linear multi-agent systems with noises. *IET Control Theory & Applications*, 10(17):2348–2356, 2016.
- [99] Xiaofeng Zong, Tao Li, and Ji-Feng Zhang. Stochastic consensus of continuous-time multi-agent systems with additive measurement noises. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 543–548. IEEE, 2015.
- [100] Long Cheng, Zeng-Guang Hou, and Min Tan. A mean square consensus protocol for linear multi-agent systems with communication noises and fixed topologies. *IEEE Transactions on Automatic Control*, 59(1):261–267, 2013.

- [101] Minyi Huang and Jonathan H Manton. Stochastic approximation for consensus seeking: Mean square and almost sure convergence. In *2007 46th IEEE Conference on Decision and Control*, pages 306–311. IEEE, 2007.
- [102] Minyi Huang and Jonathan H Manton. Stochastic consensus seeking with noisy and directed inter-agent communication: fixed and randomly varying topologies. *IEEE Transactions on Automatic Control*, 55(1):235–241, 2009.
- [103] Kenta Hanada, Takayuki Wada, Izumi Masubuchi, Toru Asai, and Yasumasa Fujisaki. Stochastic consensus algorithms over general noisy networks. *SICE Journal of Control, Measurement, and System Integration*, 13(6):274–281, 2020.
- [104] Tuncer Can Aysal and Kenneth E Barner. Convergence of consensus models with stochastic disturbances. *IEEE Transactions on Information Theory*, 56(8):4101–4113, 2010.
- [105] David Mateos-Núñez and Jorge Cortés. Noise-to-state exponentially stable distributed convex optimization on weight-balanced digraphs. *SIAM Journal on Control and Optimization*, 54(1):266–290, 2016.
- [106] Gemma Morral, Pascal Bianchi, and Gersende Fort. Success and failure of adaptation-diffusion algorithms with decaying step size in multiagent networks. *IEEE Transactions on Signal Processing*, 65(11):2798–2813, 2017.
- [107] Supriyo Chakraborty, Alun Preece, Moustafa Alzantot, Tianwei Xing, Dave Braines, and Mani Srivastava. Deep learning for situational understanding. In *2017 20th International Conference on Information Fusion (Fusion)*, pages 1–8. IEEE, 2017.
- [108] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [109] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *ICLR*, 2017.
- [110] Usman A Khan, Soumya Kar, and José MF Moura. Higher dimensional consensus: Learning in large-scale networks. *IEEE Transactions on Signal Processing*, 58(5):2836–2849, 2010.

- [111] Shreyas Sundaram and Christoforos N Hadjicostis. Finite-time distributed consensus in graphs with time-invariant topologies. In *2007 American Control Conference*, pages 711–716. IEEE, 2007.
- [112] Michel Toulouse, Bùi Quang Minh, and Quang Tran Minh. Invariant properties and bounds on a finite time consensus algorithm. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems XLI*, pages 32–58. Springer, 2019.
- [113] M. Prakash, S. Talukdar, S. Attree, S. Patel, and M. V. Salapaka. Distributed Stopping Criterion for Ratio Consensus. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 131–135, October 2018.
- [114] Vikas Yadav and Murti V Salapaka. Distributed protocol for determining when averaging consensus is reached. In *45th Annual Allerton Conf*, pages 715–720, 2007.
- [115] Mangal Prakash, Saurav Talukdar, Sandeep Attree, Vikas Yadav, and Murti V Salapaka. Distributed stopping criterion for consensus in the presence of delays. *IEEE Transactions on Control of Network Systems*, 2019.
- [116] Vivek Khatana, Govind Saraswat, Sourav Patel, and Murti V Salapaka. Gradient-consensus method for distributed optimization in directed multi-agent networks. In *2020 American Control Conference (ACC)*, pages 4689–4694. IEEE, 2020.
- [117] Sourav Patel, Vivek Khatana, Govind Saraswat, and Murti V Salapaka. Distributed detection of malicious attacks on consensus algorithms with applications in power networks. In *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*, volume 1, pages 397–402. IEEE, 2020.
- [118] Richard A. Brualdi and Herbert J. Ryser. “*Combinatorial Matrix Theory*”, volume 39. Springer, 1991.
- [119] J. Wolfowitz. Products of indecomposable, aperiodic, stochastic matrices. *Proceedings of the American Mathematical Society*, 14(5):733–737, 1963.
- [120] Angelia Nedić and Alex Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2014.

- [121] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 5(1):17–60, 1960.
- [122] Florence Bénézit, Vincent Blondel, Patrick Thiran, John Tsitsiklis, and Martin Vetterli. Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *2010 IEEE International Symposium on Information Theory*, pages 1753–1757. IEEE, 2010.
- [123] Pierre Brémaud. *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, volume 31. Springer Science & Business Media, 2013.
- [124] Vivek Khatana, Govind Saraswat, Sourav Patel, and Murti V. Salapaka. Gradconsensus: Linearly convergent algorithm for reducing disagreement in multi-agent optimization. *IEEE Transactions on Network Science and Engineering*, pages 1–13, 2023.
- [125] Kaspar Fischer. *Smallest enclosing balls of balls*. PhD thesis, ETH Zürich, 1975.
- [126] Vivek Khatana, Govind Saraswat, Sourav Patel, and Murti V Salapaka. Gradient-consensus method for distributed optimization in directed multi-agent networks. *arXiv preprint arXiv:1909.10070*, 2019.
- [127] Vivek Khatana and Murti V. Salapaka. DC-DistADMM:ADMM algorithm for constrained optimization over directed graphs. *IEEE Transactions on Automatic Control*, pages 1–16, 2022.
- [128] Vivek Khatana and Murti V. Salapaka. D-DistADMM: A $O(1/k)$ distributed admm for distributed optimization in directed graph topologies. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 2992–2997, 2020.
- [129] Vivek Khatana, Govind Saraswat, Sourav Patel, and Murti V. Salapaka. Gradient-consensus method for distributed optimization in directed multi-agent networks. In *2020 American Control Conference (ACC)*, pages 4689–4694, 2020.
- [130] Angelia Nedić and Ji Liu. Distributed optimization for control. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:77–103, 2018.
- [131] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

- [132] Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.
- [133] Angelia Nedić and Alex Olshevsky. Stochastic gradient-push for strongly convex functions on time-varying directed graphs. *IEEE Transactions on Automatic Control*, 61(12):3936–3947, 2016.
- [134] Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.
- [135] Alex Olshevsky. Linear time average consensus and distributed optimization on fixed graphs. *SIAM Journal on Control and Optimization*, 55(6):3990–4014, 2017.
- [136] Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. A proximal gradient algorithm for decentralized composite optimization. *IEEE Transactions on Signal Processing*, 63(22):6013–6023, 2015.
- [137] Zhi Li, Wei Shi, and Ming Yan. A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates. *IEEE Transactions on Signal Processing*, 67(17):4494–4506, 2019.
- [138] Ran Xin, Soumya Kar, and Usman A Khan. Gradient tracking and variance reduction for decentralized optimization and machine learning. *arXiv preprint arXiv:2002.05373*, 2020.
- [139] Jinshan Zeng and Wotao Yin. Extrapush for convex smooth decentralized optimization over directed networks. *arXiv preprint arXiv:1511.02942*, 2015.
- [140] Shi Pu, Wei Shi, Jinming Xu, and Angelia Nedić. Push–pull gradient methods for distributed optimization in networks. *IEEE Transactions on Automatic Control*, 66(1):1–16, 2020.
- [141] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.
- [142] Ermin Wei and Asuman Ozdaglar. Distributed alternating direction method of multipliers. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5445–5450. IEEE, 2012.

- [143] Ali Makhdoumi and Asuman Ozdaglar. Convergence rate of distributed admm over networks. *IEEE Transactions on Automatic Control*, 62(10):5082–5095, 2017.
- [144] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.
- [145] Ermin Wei and Asuman 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*, pages 551–554. IEEE, 2013.
- [146] Tsung-Hui Chang, Mingyi Hong, and Xiangfeng Wang. Multi-agent distributed optimization via inexact consensus admm. *IEEE Transactions on Signal Processing*, 63(2):482–497, 2014.
- [147] Franck Iutzeler, Pascal Bianchi, Philippe Ciblat, and Walid Hachem. Explicit convergence rate of a distributed alternating direction method of multipliers. *IEEE Transactions on Automatic Control*, 61(4):892–904, 2015.
- [148] Fatemeh Mansoori and Ermin Wei. A flexible framework of first-order primal-dual algorithms for distributed optimization. *arXiv preprint arXiv:1912.07526*, 2019.
- [149] Daniel Gabay. Chapter ix applications of the method of multipliers to variational inequalities. In *Studies in mathematics and its applications*, volume 15, pages 299–331. Elsevier, 1983.
- [150] Gang Chen, Qing Yang, Yongduan Song, and Frank L Lewis. A distributed continuous-time algorithm for nonsmooth constrained optimization. *IEEE Transactions on Automatic Control*, 65(11):4914–4921, 2020.
- [151] Gang Chen, Qing Yang, Yongduan Song, and Frank L Lewis. Fixed-time projection algorithm for distributed constrained optimization on time-varying digraphs. *IEEE Transactions on Automatic Control*, 2021.
- [152] Minghui Zhu and Sonia Martínez. On distributed convex optimization under inequality and equality constraints. *IEEE Transactions on Automatic Control*, 57(1):151–164, 2011.

- [153] Fangzheng Tian, Wenwu Yu, Junjie Fu, Wei Gu, and Juping Gu. Distributed optimization of multiagent systems subject to inequality constraints. *IEEE transactions on cybernetics*, 2019.
- [154] Shaofu Yang, Qingshan Liu, and Jun Wang. A multi-agent system with a proportional-integral protocol for distributed constrained optimization. *IEEE Transactions on Automatic Control*, 62(7):3461–3467, 2016.
- [155] Qingshan Liu and Jun Wang. A second-order multi-agent network for bound-constrained distributed optimization. *IEEE Transactions on Automatic Control*, 60(12):3310–3315, 2015.
- [156] Hongbing Zhou, Xianlin Zeng, and Yiguang Hong. Adaptive exact penalty design for constrained distributed optimization. *IEEE Transactions on Automatic Control*, 64(11):4661–4667, 2019.
- [157] Peng Lin, Jiahao Xu, Wei Ren, Chunhua Yang, and Weihua Gui. Angle-based analysis approach for distributed constrained optimization. *IEEE Transactions on Automatic Control*, 2021.
- [158] Qingshan Liu, Shaofu Yang, and Yiguang Hong. Constrained consensus algorithms with fixed step size for distributed convex optimization over multiagent networks. *IEEE Transactions on Automatic Control*, 62(8):4259–4265, 2017.
- [159] Angelia Nedic, Asuman Ozdaglar, and Pablo A Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938, 2010.
- [160] Huaqing Li, Qingguo Lü, Guo Chen, Tingwen Huang, and Zhaoyang Dong. Distributed constrained optimization over unbalanced directed networks using asynchronous broadcast-based algorithm. *IEEE Transactions on Automatic Control*, 66(3):1102–1115, 2020.
- [161] Pei Xie, Keyou You, Roberto Tempo, Shiji Song, and Cheng Wu. Distributed convex optimization with inequality constraints over time-varying unbalanced digraphs. *IEEE Transactions on Automatic Control*, 63(12):4331–4337, 2018.
- [162] Joao FC Mota, Joao MF Xavier, Pedro MQ Aguiar, and Markus Püschel. D-admm: A communication-efficient distributed algorithm for separable optimization. *IEEE Transactions on Signal Processing*, 61(10):2718–2723, 2013.

- [163] Deming Yuan, Daniel WC Ho, and Shengyuan Xu. Regularized primal–dual subgradient method for distributed constrained optimization. *IEEE transactions on cybernetics*, 46(9):2109–2118, 2015.
- [164] Jinlong Lei, Han-Fu Chen, and Hai-Tao Fang. Primal–dual algorithm for distributed constrained optimization. *Systems & Control Letters*, 96:110–117, 2016.
- [165] Ilan Lobel and Asuman Ozdaglar. Distributed subgradient methods for convex optimization over random networks. *IEEE Transactions on Automatic Control*, 56(6):1291–1306, 2010.
- [166] John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2011.
- [167] Konstantinos I Tsianos, Sean Lawlor, and Michael G Rabbat. Push-sum distributed dual averaging for convex optimization. In *IEEE conf. on decision and control*, pages 5453–5458. IEEE, 2012.
- [168] Ran Xin and Usman A Khan. A linear algorithm for optimization over directed graphs with geometric convergence. *IEEE Control Systems Letters*, 2(3):315–320, 2018.
- [169] Z. Feng and G. Hu. A distributed constrained optimization approach for spatiotemporal connectivity-preserving rendezvous of multi-robot systems. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 987–992, 2018.
- [170] Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. Collaborative deep learning in fixed topology networks. *Advances in Neural Information Processing Systems*, 30, 2017.
- [171] Yasaman Esfandiari, Koushik Nagasubramanian, Fateme Fotouhi, Patrick S Schnable, Baskar Ganapathysubramanian, and Soumik Sarkar. Distributed deep learning for persistent monitoring of agricultural fields. In *NeurIPS 2021 AI for Science Workshop*, 2021.
- [172] Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. On consensus-optimality trade-offs in collaborative deep learning. *Frontiers in artificial intelligence*, 4:573731, 2021.

- [173] John W Simpson-Porco, Qobad Shafiee, Florian Dörfler, Juan C Vasquez, Josep M Guerrero, and Francesco Bullo. Secondary frequency and voltage control of islanded microgrids via distributed averaging. *IEEE Transactions on Industrial Electronics*, 62(11):7025–7038, 2015.
- [174] Daniel K Molzahn, Florian Dörfler, Henrik Sandberg, Steven H Low, Sambuddha Chakrabarti, Ross Baldick, and Javad Lavaei. A survey of distributed optimization and control algorithms for electric power systems. *IEEE Transactions on Smart Grid*, 8(6):2941–2962, 2017.
- [175] Dušan Jakovetić, Joao Xavier, and José MF Moura. Fast distributed gradient methods. *IEEE Transactions on Automatic Control*, 59(5):1131–1146, 2014.
- [176] Annie I-An Chen. *Fast distributed first-order methods*. PhD thesis, Massachusetts Institute of Technology, 2012.
- [177] Bjorn Johansson, Tamás Keviczky, Mikael Johansson, and Karl Henrik Johansson. Subgradient methods and consensus algorithms for solving convex optimization problems. In *IEEE Conf. on Decision and Control*, pages 4185–4190. IEEE, 2008.
- [178] Albert S Berahas, Raghu Bollapragada, Nitish Shirish Keskar, and Ermin Wei. Balancing communication and computation in distributed optimization. *IEEE Transactions on Automatic Control*, 64(8):3141–3155, 2018.
- [179] James Melbourne, Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V Salapaka. On the geometry of consensus algorithms with application to distributed termination in higher dimension. *International Federation of Automatic Control (IFAC)*, 2020.
- [180] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.
- [181] Guannan Qu and Na Li. Accelerated distributed nesterov gradient descent. *IEEE Transactions on Automatic Control*, 2019.
- [182] Yurii Nesterov. *Introductory lectures on convex opt.: A basic course*, volume 87. Springer Science & Business Media, 2013.
- [183] Pierluigi Crescenzi, Roberto Grossi, Claudio Imbrenda, Leonardo LANZI, and Andrea Marino. Finding the diameter in real-world graphs: Experimentally turning

- a lower bound into an upper bound. In *Algorithms–ESA 2010: 18th Annual European Symposium, Liverpool, UK, September 6–8, 2010. Proceedings, Part I 18*, pages 302–313. Springer, 2010.
- [184] Fan Chung and Linyuan Lu. The diameter of sparse random graphs. *Advances in Applied Mathematics*, 26(4):257–279, 2001.
 - [185] Alejandro D Dominguez-Garcia and Christoforos N Hadjicostis. Distributed algorithms for control of demand response and distributed energy resources. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 27–32. IEEE, 2011.
 - [186] R Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976.
 - [187] Hui Zhang and Lizhi Cheng. Restricted strong convexity and its applications to convergence analysis of gradient-type methods in convex optimization. *Optimization Letters*, 9(5):961–979, 2015.
 - [188] Francis Bach. Adaptivity of averaged stochastic gradient descent to local strong convexity for logistic regression. *The Journal of Machine Learning Research*, 15(1):595–627, 2014.
 - [189] Albert S Berahas, Raghu Bollapragada, and Ermin Wei. On the convergence of nested decentralized gradient methods with multiple consensus and gradient steps. *IEEE Transactions on Signal Processing*, 69:4192–4203, 2021.
 - [190] Huan Li, Cong Fang, Wotao Yin, and Zhouchen Lin. A sharp convergence rate analysis for distributed accelerated gradient methods. *arXiv preprint arXiv:1810.01053*, 2018.
 - [191] Haishan Ye, Luo Luo, Ziang Zhou, and Tong Zhang. Multi-consensus decentralized accelerated gradient descent. *arXiv preprint arXiv:2005.00797*, 2020.
 - [192] Tianyi Lin, Shiqian Ma, and Shuzhong Zhang. On the global linear convergence of the admm with multiblock variables. *SIAM Journal on Optimization*, 25(3):1478–1497, 2015.

- [193] R Tyrrell Rockafellar. Augmented lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of operations research*, 1(2):97–116, 1976.
- [194] Angelia Nedic, Alex Olshevsky, and Wei Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597–2633, 2017.
- [195] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.
- [196] Vivek Khatana, Soham Chakraborty, and Murti V Salapaka. A plug and play distributed secondary controller for microgrids with grid-forming inverters. *2024 Annual Conference of the IEEE Industrial Electronics Society (IECON)*, *arXiv preprint arXiv:2409.19866*, 2024.
- [197] Vivek Khatana, Soham Chakraborty, Govind Saraswat, Sourav Patel, and Murti V Salapaka. A distributed malicious agent detection scheme for resilient power apportioning in microgrids. *2024 Annual Conference of the IEEE Industrial Electronics Society (IECON)*, *arXiv preprint arXiv:2409.19861*, 2024.
- [198] Vivek Khatana, Soham Chakraborty, Govind Saraswat, Blake Lundstorm, and Murti V. Salapaka. Evaluation of horizon of viability optimization engine for sustained power to critical infrastructure. In *2022 IEEE Power and Energy Conference at Illinois (PECI)*, pages 1–8. IEEE, 2022.
- [199] Josep M Guerrero, Juan C Vasquez, José Matas, Luis García De Vicuña, and Miguel Castilla. Hierarchical control of droop-controlled ac and dc microgrids—a general approach toward standardization. *IEEE Transactions on industrial electronics*, 58(1):158–172, 2010.
- [200] Mukul C Chandorkar, Deepakraj M Divan, and Rambabu Adapa. Control of parallel connected inverters in standalone ac supply systems. *IEEE transactions on industry applications*, 29(1):136–143, 1993.
- [201] Qing-Chang Zhong. Robust droop controller for accurate proportional load sharing among inverters operated in parallel. *IEEE Transactions on industrial Electronics*, 60(4):1281–1290, 2011.

- [202] Mehdi Savaghebi, Alireza Jalilian, Juan C Vasquez, and Josep M Guerrero. Secondary control scheme for voltage unbalance compensation in an islanded droop-controlled microgrid. *IEEE Transactions on Smart Grid*, 3(2):797–807, 2012.
- [203] Alexander Micallef, Maurice Apap, Cyril Spiteri-Staines, Josep M Guerrero, and Juan C Vasquez. Reactive power sharing and voltage harmonic distortion compensation of droop controlled single phase islanded microgrids. *IEEE Transactions on Smart Grid*, 5(3):1149–1158, 2014.
- [204] Qobad Shafiee, Juan C Vasquez, and Josep M Guerrero. Distributed secondary control for islanded microgrids—a networked control systems approach. In *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*, pages 5637–5642. IEEE, 2012.
- [205] Qobad Shafiee, Josep M Guerrero, and Juan C Vasquez. Distributed secondary control for islanded microgrids—a novel approach. *IEEE Transactions on power electronics*, 29(2):1018–1031, 2013.
- [206] Qobad Shafiee, Čedomir Stefanović, Tomislav Dragičević, Petar Popovski, Juan C Vasquez, and Josep M Guerrero. Robust networked control scheme for distributed secondary control of islanded microgrids. *IEEE Transactions on Industrial Electronics*, 61(10):5363–5374, 2013.
- [207] Ali Bidram, Ali Davoudi, Frank L Lewis, and Zhihua Qu. Secondary control of microgrids based on distributed cooperative control of multi-agent systems. *IET Generation, Transmission & Distribution*, 7(8):822–831, 2013.
- [208] Nima Mahdian Dehkordi, Nasser Sadati, and Mohsen Hamzeh. Fully distributed cooperative secondary frequency and voltage control of islanded microgrids. *IEEE Transactions on Energy Conversion*, 32(2):675–685, 2016.
- [209] John W Simpson-Porco, Florian Dörfler, and Francesco Bullo. Synchronization and power sharing for droop-controlled inverters in islanded microgrids. *Automatica*, 49(9):2603–2611, 2013.
- [210] Gang Chen and Ening Feng. Distributed secondary control and optimal power sharing in microgrids. *IEEE/CAA Journal of Automatica Sinica*, 2(3):304–312, 2015.

- [211] Xiaoqing Lu, Xinghuo Yu, Jingang Lai, Yaonan Wang, and Josep M Guerrero. A novel distributed secondary coordination control approach for islanded microgrids. *IEEE Transactions on Smart Grid*, 9(4):2726–2740, 2016.
- [212] Xiaoqing Lu, Xinghuo Yu, Jingang Lai, Josep M Guerrero, and Hong Zhou. Distributed secondary voltage and frequency control for islanded microgrids with uncertain communication links. *IEEE Transactions on Industrial Informatics*, 13(2):448–460, 2016.
- [213] Lei Ding, Qing-Long Han, and Xian-Ming Zhang. Distributed secondary control for active power sharing and frequency regulation in islanded microgrids using an event-triggered communication mechanism. *IEEE Transactions on Industrial Informatics*, 15(7):3910–3922, 2018.
- [214] Junhui Zhang, Yuxi Men, Lizhi Ding, and Xiaonan Lu. Secondary frequency and voltage regulation for inverter-based microgrids: A sparsity-promoting dapi control approach. *IEEE Transactions on Control Systems Technology*, 2024.
- [215] Yixin Zhu, Fang Zhuo, Feng Wang, Baoquan Liu, Ruifeng Gou, and Yangjie Zhao. A virtual impedance optimization method for reactive power sharing in networked microgrid. *IEEE Transactions on Power Electronics*, 31(4):2890–2904, 2015.
- [216] Yalong Hu, Ji Xiang, Yonggang Peng, Pengcheng Yang, and Wei Wei. Decentralised control for reactive power sharing using adaptive virtual impedance. *IET Generation, Transmission & Distribution*, 12(5):1198–1205, 2018.
- [217] Huaguang Zhang, Sunghyok Kim, Qiuye Sun, and Jianguo Zhou. Distributed adaptive virtual impedance control for accurate reactive power sharing based on consensus control in microgrids. *IEEE Transactions on Smart Grid*, 8(4):1749–1761, 2016.
- [218] P. Moeller, J. Norris, and C. A. LaFleur. Compliance filing of the midwest independent transmission system operator, inc. regarding dispatchable intermittent resources docket no. ER11-1991- 00, 2011.
- [219] R. Konidena. FERC order 841 levels the playing field for energy storage. *MRS Energy & Sustainability*, 2019.

- [220] Eneko Unamuno and Jon Andoni Barrena. Hybrid ac/dc microgrids—part ii: Review and classification of control strategies. *Renewable and Sustainable Energy Reviews*, 52:1123–1134, 2015.
- [221] Kaihua Xi, Johan LA Dubbeldam, Hai Xiang Lin, and Jan H van Schuppen. Power-imbalance allocation control of power systems-secondary frequency control. *Automatica*, 92:72–85, 2018.
- [222] Enrique Espina, Jacqueline Llanos, Claudio Burgos-Mellado, Roberto Cárdenas-Dobson, Manuel Martínez-Gómez, and Doris Sáez. Distributed control strategies for microgrids: An overview. *IEEE Access*, 8:193412–193448, 2020.
- [223] Youssef Hennane, Serge Pierfederici, Abdelmajid Berdai, Farid Meibody-Tabar, and Jean-Philippe Martin. Distributed control of islanded meshed microgrids. *IEEE Access*, 11:78262–78272, 2023.
- [224] Xinghua Huang, Yuanliang Fan, Han Wu, Gonglin Zhang, Zili Yin, Muhammad Yasir Ali Khan, Haoming Liu, and Jingjing Zhai. Distributed secondary control for islanded microgrids considering communication delays. *IEEE Access*, 12:64335–64350, 2024.
- [225] Florian Dörfler and Sergio Grammatico. Gather-and-broadcast frequency control in power systems. *Automatica*, 79:296–305, 2017.
- [226] Changhong Zhao, Enrique Mallada, and Florian Dörfler. Distributed frequency control for stability and economic dispatch in power networks. In *2015 American Control Conference (ACC)*, pages 2359–2364. IEEE, 2015.
- [227] Olivier Mégel, Tao Liu, David J Hill, and Göran Andersson. Distributed secondary frequency control algorithm considering storage efficiency. *IEEE Transactions on Smart Grid*, 9(6):6214–6228, 2017.
- [228] Sourav Patel, Sandeep Attree, Saurav Talukdar, Mangal Prakash, and Murti V. Salapaka. Distributed apportioning in a power network for providing demand response services. In *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 38–44, 2017.
- [229] Sourav Patel, Blake Lundstrom, Govind Saraswat, and Murti V. Salapaka. Distributed power apportioning with early dispatch for ancillary services in renewable grids. *ArXiv*, abs/2007.11715, 2020.

- [230] Govind Saraswat, Vivek Khatana, Sourav Patel, and Murti V. Salapaka. Distributed finite-time termination for consensus algorithm in switching topologies. *IEEE Transactions on Network Science and Engineering*, 10(1):489–499, 2023.
- [231] Vivek Khatana and Murti V. Salapaka. DC-DistADMM:ADMM algorithm for constrained optimization over directed graphs. *IEEE Transactions on Automatic Control*, 68(9):5365–5380, 2023.
- [232] Stephen Paul Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, University of Stirling, 1994.
- [233] Michel Toulouse, Hai Le, Cao Vien Phung, and Denis Hock. Defense Strategies against Byzantine Attacks in a Consensus-Based Network Intrusion Detection System. page 16, 2017.
- [234] H. Zhang and S. Sundaram. Robustness of information diffusion algorithms to locally bounded adversaries. In *American Control Conference*, pages 5855–5561, June 2012.
- [235] J. Usevitch and D. Panagou. Resilient leader-follower consensus to arbitrary reference values. In *American Control Conference*, 2018.
- [236] Sourav Patel, Vivek Khatana, Govind Saraswat, and Murti V. Salapaka. Distributed detection of malicious attacks on consensus algorithms with applications in power networks. In *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*, volume 1, pages 397–402, 2020.
- [237] Nishant Kishore, Domingo Marqués, Ayesha Mahmud, Mathew V Kiang, Irmay Rodriguez, Arlan Fuller, Peggy Ebner, Cecilia Sorensen, Fabio Racy, Jay Lemery, et al. Mortality in puerto rico after hurricane maria. *New England journal of medicine*, 379(2):162–170, 2018.
- [238] Richard J Campbell and Sean Lowry. Weather-related power outages and electric system resiliency. Congressional Research Service, Library of Congress Washington, DC, 2012.
- [239] Energy institue at UT Austin. The timeline and events of the february 2021 texas electric grid blackouts, 2021. [November 18, 2021].

- [240] Ieee standard for the specification of microgrid controllers. *IEEE Std 2030.7-2017*, pages 1–43, 2018.
- [241] Chu Sun, Geza Joos, Syed Qaseem Ali, Jean Nicolas Paquin, Carlos Mauricio Rangel, Fares Al Jajeh, Ilja Novickij, and Francois Bouffard. Design and real-time implementation of a centralized microgrid control system with rule-based dispatch and seamless transition function. *IEEE Transactions on Industry Applications*, 56(3):3168–3177, 2020.
- [242] Scott Manson, Bharath Nayak, and Will Allen. Robust microgrid control system for seamless transition between grid-tied and island operating modes. In *44th Annual Western Protective Relay Conference*, 2017.
- [243] Ammar Mohammed Ameen, Jagadeesh Pasupuleti, and Tamer Khatib. Simplified performance models of photovoltaic/diesel generator/battery system considering typical control strategies. *Energy Conversion and Management*, 99:313–325, 2015.
- [244] Bo Zhao, Xuesong Zhang, Peng Li, Ke Wang, Meidong Xue, and Caisheng Wang. Optimal sizing, operating strategy and operational experience of a stand-alone microgrid on dongfushan island. *Applied Energy*, 113:1656–1666, 2014.
- [245] Chu Sun, Geza Joos, and Francois Bouffard. Adaptive coordination for power and soc limiting control of energy storage in an islanded ac microgrid with impact load. *IEEE Transactions on Power Delivery*, 35(2):580–591, 2019.
- [246] Donatello Materassi, Saverio Bolognani, Mardavij Roozbehani, and Munther A Dahleh. Deferrable loads in an energy market: Coordination under congestion constraints. In *22nd Mediterranean Conference on Control and Automation*, pages 628–633. IEEE, 2014.
- [247] Anand Subramanian, Manuel Garcia, Alejandro Dominguez-Garcia, Duncan Callaway, Kameshwar Poolla, and Pravin Varaiya. Real-time scheduling of deferrable electric loads. In *2012 American Control Conference (ACC)*, pages 3643–3650. IEEE, 2012.
- [248] Lingwen Gan, Adam Wierman, Ufuk Topcu, Niangjun Chen, and Steven H Low. Real-time deferrable load control: handling the uncertainties of renewable generation. In *Proceedings of the fourth international conference on Future energy systems*, pages 113–124, 2013.

- [249] Saajid Haque, Donatello Materassi, Saverio Bolognani, Mardavij Roozbehani, and Munther A Dahleh. An efficient partial-order representation of feasible schedules for online decisions. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2641–2646. IEEE, 2017.
- [250] Rajesh Bhana and Thomas J Overbye. The commitment of interruptible load to ensure adequate system primary frequency response. *IEEE Transactions on Power Systems*, 31(3):2055–2063, 2015.
- [251] Xiaoyu Wu, Jinghan He, Yin Xu, Jian Lu, Ning Lu, and Xiaojun Wang. Hierarchical control of residential hvac units for primary frequency regulation. *IEEE Transactions on Smart Grid*, 9(4):3844–3856, 2017.
- [252] Blake Lundstrom, Sourav Patel, and Murti V. Salapaka. Distribution feeder-scale fast frequency response via optimal coordination of net-load resources—part i: Solution design. *IEEE Transactions on Smart Grid*, 12(2):1289–1302, 2021.
- [253] Armin Ghasem Azar, Hamidreza Nazaripouya, Behnam Khaki, Chi-Cheng Chu, Rajit Gadh, and Rune Hylsberg Jacobsen. A non-cooperative framework for coordinating a neighborhood of distributed prosumers. *IEEE Transactions on Industrial Informatics*, 15(5):2523–2534, 2018.
- [254] Jianqiang Hu, Jinde Cao, Josep M Guerrero, Taiyou Yong, and Jie Yu. Improving frequency stability based on distributed control of multiple load aggregators. *IEEE Transactions on Smart Grid*, 8(4):1553–1567, 2015.
- [255] Lei Tang and James McCalley. Two-stage load control for severe under-frequency conditions. *IEEE Transactions on Power Systems*, 31(3):1943–1953, 2015.
- [256] Alexander M Prostejovsky, Mattia Marinelli, Michel Rezkalla, Mazheruddin H Syed, and Efren Guillo-Sansano. Tuningless load frequency control through active engagement of distributed resources. *IEEE Transactions on Power Systems*, 33(3):2929–2939, 2017.
- [257] Anand Subramanian, Manuel J Garcia, Duncan S Callaway, Kameshwar Poolla, and Pravin Varaiya. Real-time scheduling of distributed resources. *IEEE Transactions on Smart Grid*, 4(4):2122–2130, 2013.

- [258] Vivek Khatana, Govind Saraswat, Sourav Patel, and Murti V. Salapaka. Grad-consensus: Linearly convergent algorithm for reducing disagreement in multi-agent optimization. *IEEE Transactions on Network Science and Engineering*, 11(1):1251–1264, 2024.
- [259] Vivek Khatana and Murti V. Salapaka. Noise resilient distributed average consensus over directed graphs. *IEEE Transactions on Signal and Information Processing over Networks*, 9:770–785, 2023.
- [260] K Strunz, RH Fletcher, R Campbell, and F Gao. Developing benchmark models for low-voltage distribution feeders. In *2009 IEEE Power & Energy Society General Meeting*, pages 1–3. IEEE, 2009.
- [261] Soham Chakraborty, Sourav Patel, and Murti V. Salapaka. μ -synthesis-based generalized robust framework for grid-following and grid-forming inverters. *IEEE Transactions on Power Electronics*, 38(3):3163–3179, 2023.
- [262] Soham Chakraborty, Sourav Patel, and Murti V. Salapaka. Robust and optimal single-loop voltage controller for grid-forming voltage source inverters. In *2020 IEEE Power and Energy Conference at Illinois (PECI)*, pages 1–7, 2020.
- [263] User Datagram Protocol. https://en.wikipedia.org/wiki/User_Datagram_Protocol, 2021. [November 18, 2021].
- [264] JA Peças Lopes, CL Moreira, and AG Madureira. Defining control strategies for microgrids islanded operation. *IEEE Transactions on power systems*, 21(2):916–924, 2006.
- [265] Blake Lundstrom, Sourav Patel, and Murti V Salapaka. Distribution feeder-scale fast frequency response via optimal coordination of net-load resources—part i: Solution design. *IEEE Transactions on Smart Grid*, 12(2):1289–1302, 2020.
- [266] Jing Wang, Soham Chakraborty, Vivek Khatana, Blake Lundstrom, Govind Sarawat, and Murti Salapaka. Evaluation of optimal net load management in microgrids using hardware-in-the-loop simulation. In *2022 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5, 2022.
- [267] National Renewable Energy Laboratory. <https://www.nrel.gov/gis/data-tools.html>, 2024. [December 23, 2024].

- [268] Laurence A Wolsey. *Integer programming*. John Wiley & Sons, 2020.
- [269] MOSEK ApS. The MOSEK optimization toolbox for MATLAB manual. Version 9.0., 2019.
- [270] Vivek Khatana and Murti V. Salapaka. Distributed difference of convex optimization. In *2024 60th Annual Allerton Conference on Communication, Control, and Computing*, pages 1–8, 2024.

Appendix A

A.1 Software-Hardware Testbed for Emulating Real-life Network Dynamical Systems

Deploying distributed algorithms directly in real-life NDS, such as power grids or transportation systems, can risk system performance and stability. To address these challenges and enable seamless testing and deployment of distributed algorithms, this dissertation develops a comprehensive experimental testbed. The testbed integrates physical hardware components, real-time emulators for system simulation, and a robust software engine to run optimization and learning algorithms. This modular setup provides a realistic environment for evaluating the performance, robustness, and scalability of distributed methods under various operational scenarios.

The testbed is designed to cater to a diverse range of users, including graduate students conducting research in academic settings and engineers at utility companies seeking practical solutions. By enabling rigorous testing and validation, it ensures that methods are field-ready and capable of maintaining system stability and efficiency. Furthermore, the testbed supports integration with emerging technologies, such as IoT devices and advanced communication protocols, to simulate future-ready applications. This platform is a significant step towards bridging the gap between theoretical advancements and their practical deployment in critical infrastructure systems.

The developed open-source software package, built using Python and Node.js computer languages, is available free of cost on GitHub^{1,2}. The software package can be readily booted onto a microcontroller or a Raspberry Pi, to interface with the physical hardware in NDS environments. The testbed is illustrated in Figure A.1.

¹https://github.com/vivekkhatana/VPP_Primary_Node

²https://github.com/vivekkhatana/VPP_Secondary_Node

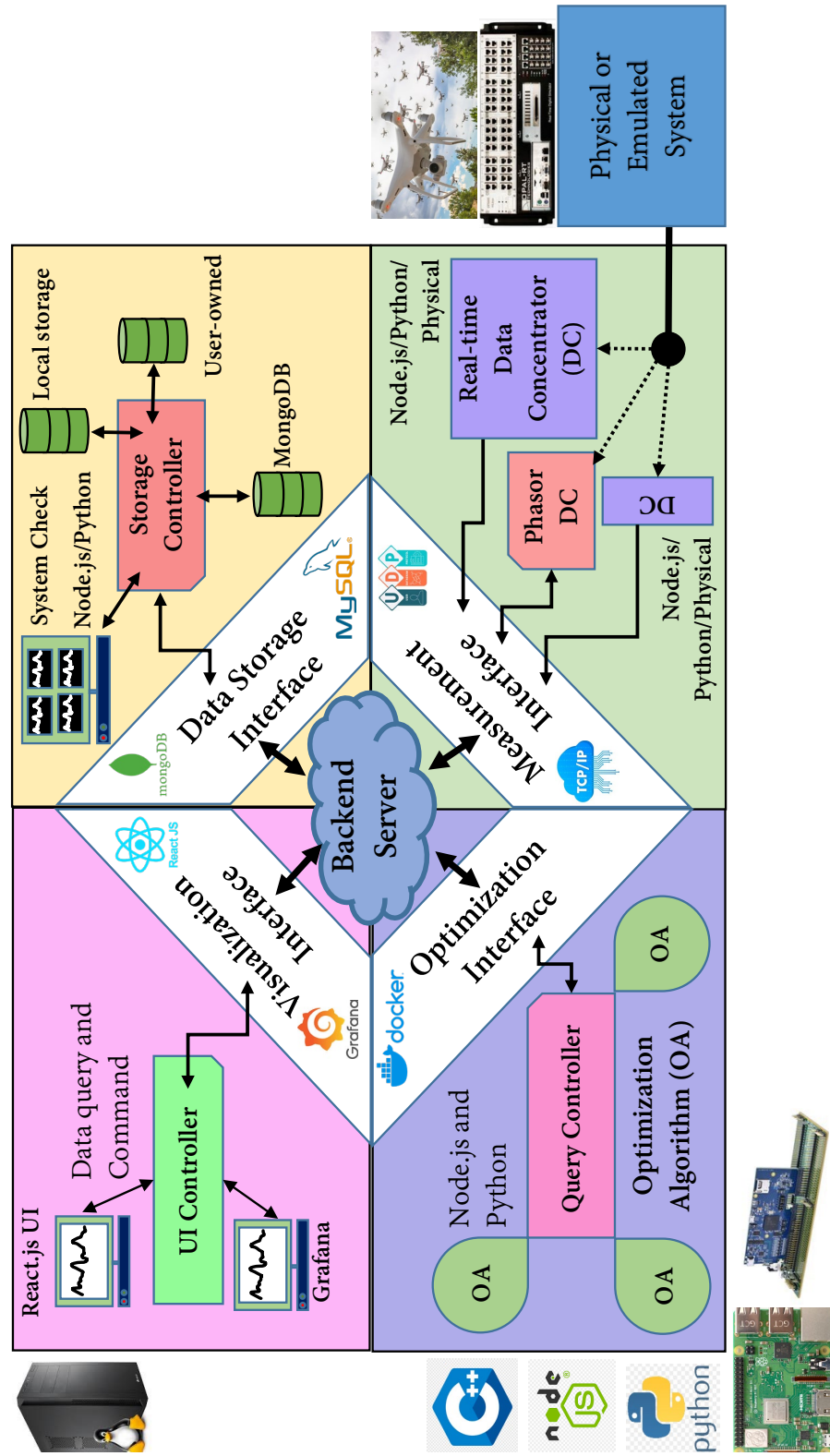


Figure A.1: Block diagram showing the integration of the five major components Software-Hardware testbed.

The high-level objectives of the testbed to:

- Emulate or create the physical network components of the NDS.
- Create an interface between the physical NDS network and the software-based decision-making engine.
- Develop a software-based distributed decision-making engine based on distributed optimization and learning algorithms
- Build a storage database to store the system conditions for visualization and information processing.

The software-hardware testbed consists of five main component layers:

- **Backend Server:** The Backend Server bridges the four interface layers. It relays requests and transfers data between different layers.
- **Measurement Interface:** It receives measurements from the NDS's physical resources via a data concentrator (DC). The DC supports both wireless and wired communication protocols including UDP, TCP, and MODBUS. The measurement interface keeps track of the latest system state. The communication channels relay the latest measurements to visualization, data storage and decision-making optimization layer via the backend server. A socket.io-based support using Node.js and Python is developed for the sensor interface.
- **Data Storage Interface:** The storage controller program routes user queries to the appropriate storage systems for data analysis and metric calculations in the Data Storage Interface. The Data Storage Interface receives measurements from the Measurement Interface and the latest decision commands from the Optimization Interface. Additionally, it relays NDS information to both the Visualization Interface and the Optimization Interface, ensuring seamless integration and data flow across all components.
- **Visualization Interface:** It receives the state of the NDS from either DCs in the Measurement Interface or the Data Storage Interface (or both). It creates monitoring and situational awareness visualization for the NDS. The developed Visualization Interface supports multiple user interface computer programming languages including React.js and Grafana.

- **Optimization Interface:** Optimization Interface is the heart of the decision-making in the NDS. Here, a single query controller computer program allows for the interface of several distributed optimization algorithms for example the **GradConsensus** and **DC-DistADMM** algorithms developed in this thesis. It allows for distributed synthesis and implementation. The optimization layer has multiple programming languages supported including Node.js and Python. Measurements from the Measurement Interface and past system states from the Data Storage Interface are relayed to the Optimization Interface. The output decisions are sent to the Data Storage and Visualization layers via the backend server.

A.1.1 An Instantiation for Decision-Making in Power Systems

In this section, we provide an example of using the software-hardware testbed for testing and emulating decision-making in a particular NDS, namely Power Systems. Figure A.2 provides the experimental setup. Here, the backend server (also referred to as the Demo Node) is a computer script running Python or Node.js. The physical layer of the power system consists of external loads, photovoltaic system simulators, power inverters, and a Real-time Simulator (RTS) which digitally emulates the dynamics of the power network. In the power system application we utilize the Opal-RT based OP5700 RTS. The optimization layer responsible for decision-making consists of a Controller Hardware-in-the-loop (CHIL) setup. The decision variables for power system operation are generated via distributed algorithms executed on microcontroller boards, Raspberry PI controllers, and computer programming scripts. The power system state and the determined decision variables are relayed to the data storage controller that runs a Python script to save system information to a database. The stored data is utilized for visualization of the state of the system. The visualization panel of the OP5700 RTS is utilized as well to provide power system situational awareness. In this instantiation UDP communication protocol is utilized between the RTS and the backend server (or Demo Node) and among the distributed agents in the optimization layer.

Power System Physical Layer interfaced with RT Simulator

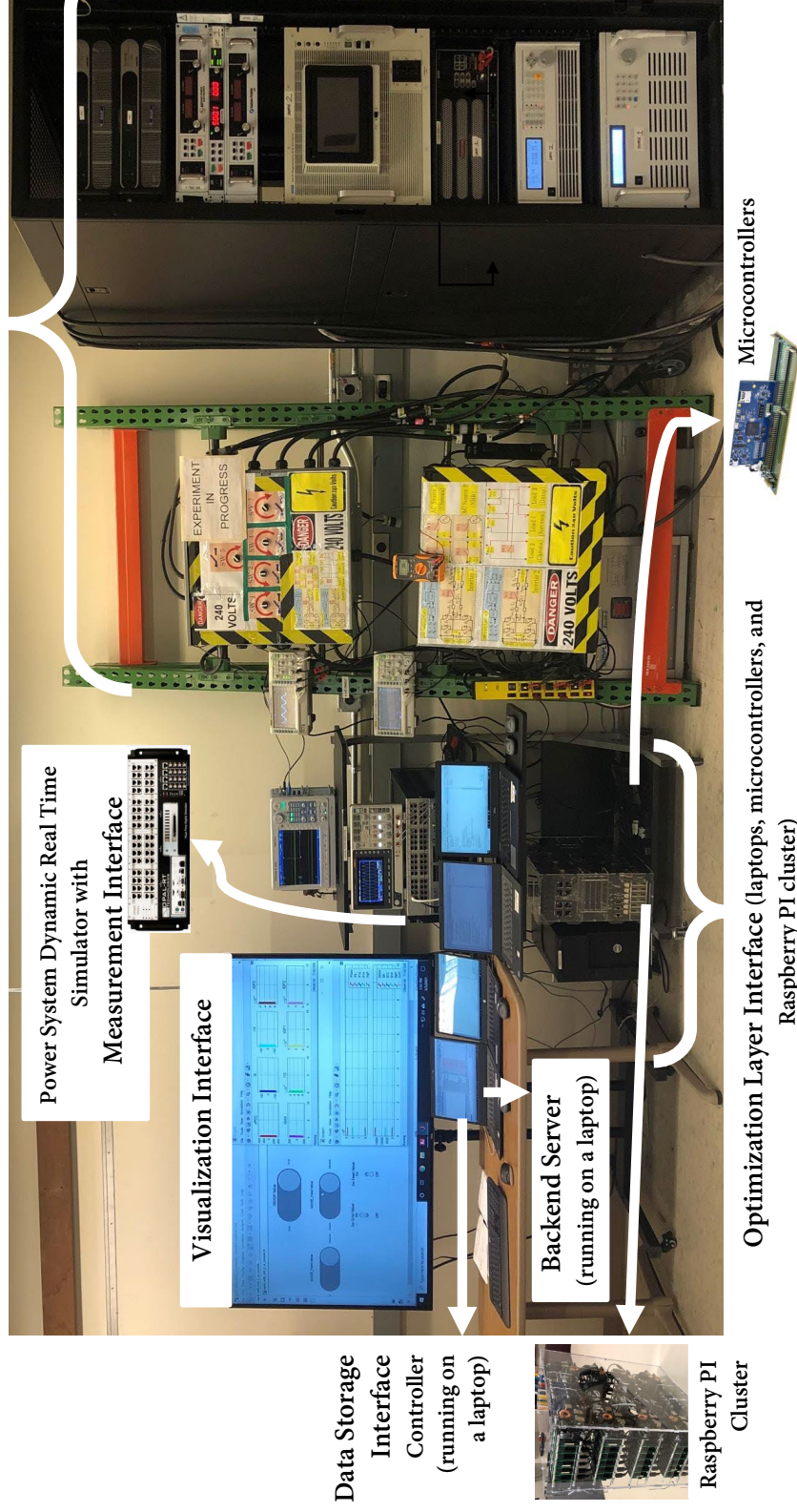


Figure A.2: Software-Hardware testbed for power system applications.

Data Storage
Interface
Controller
(running on
a laptop)

The experiments and testing proceed in the following steps:

- STEP-1: The measurements from the power system (emulated in RTS) are relayed via UDP communication channel to the backend server (or the Demo Node). During testing the user should ensure:
 - i) The IP addresses of the UDP communication channels attempted to be used by the RTS and the backend server (or Demo Node) align with their actual IP addresses at the RTS and backend server (or Demo Node) location.
 - ii) The size of the data sample expected by the backend server (or Demo Node) during the message passing on the UDP channel matches the size of the measurement vector sent by the RTS.
- STEP-2: The backend server (or Demo Node) passes the received measurement from the RTS to the optimization layer for the determination of optimal control actions. Further, the latest measurements are also sent to the data storage controller to be saved in a database. This step requires the following debugging steps:
 - i) The IP addresses of the UDP communication channels attempted to be used by the backend server (or Demo Node) and the distributed agents in the optimization layer align with their actual IP addresses at their physical locations.
 - ii) The IP addresses of the UDP communication channels attempted to be used by the backend server (or Demo Node) and the data storage controller align with their actual IP addresses at their physical locations.
- STEP-3: Once the latest measurements are received the distributed algorithm gets triggered. The agents compute their local solution and utilize UDP communication protocol to share their estimates with their neighboring agents. The algorithm converges and determines an optimal decision for each agent. The user should ensure that:
 - i) The neighborhood information of each agent in the underlying exactly dictate the number of communication channels used by each agent. Further, the IP addresses of the UDP communication channels attempted to be utilized by the agents matches their physical location IP addresses.
 - ii) For large networks, the algorithmic parameters including the graph topology and delay bounds between updates are properly initialized.
- STEP-4: The decisions generated by the optimization layer are relayed to the backend server (or Demo Node) that further passes them to the RTS. The optimal

decisions are then applied within the power system. The user should ensure that:

- i) The size of the expected data samples matches with the size of data samples sent by the distributed agents, the backend server (or Demo Node), and the RTS.
- ii) The UDP channels have the same assigned IP addresses to them in the software programs as their physical locations.

A.1.2 Example Commands for the Power Systems Application

In this section, we provide the commands to execute the software package running the backend server (or Demo Node), the data storage controller, and the distributed agents in the optimization layer. The backend server (or Demo Node) code is available on Github³. The code for optimization layer utilizes the developed **DC-DistADMM** algorithm as the decision-making engine⁴.

Run Commands:

- Backend Server (or Demo Node): The Backend Server (or Demo Node) Python script can be run via writing:
`python primary_node.py`
at the command line.
- Distributed **DC-DistADMM** based optimization script: Python script can be run via writing:
`python secondary_node.py`
at the command line.
- The data storage controller is run via the command:
`python MeasurementLogging.py`
at the command line.

Input Files:

Both the `primary_node.py` and `secondary_node.py` require an input file named as `mastercluster.txt`. The file `mastercluster.txt` initializes the IP addresses of the RTS, backend server (or Demo Node), and the optimization layer. The script `primary_node.py`

³https://github.com/vivekkhatana/VPP_Primary_Node

⁴https://github.com/vivekkhatana/VPP_Secondary_Node

also requires an input labeled as `agentConfig.txt`. The information about the communication topology is provided via the `agentConfig.txt` file.

Console Display-Startup:

Executing `python primary_node.py` in a terminal will start the module. If startup is successful, the following messages should be seen:

```
2024-12-29 18:03:07,141 [MainThread ] [INFO ] Starting primary...
2024-12-29 18:03:16,189 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52569)
2024-12-29 18:03:16,194 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52571)
2024-12-29 18:03:16,196 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52573)
2024-12-29 18:03:16,198 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52572)
2024-12-29 18:03:16,200 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52570)
2024-12-29 18:03:16,203 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52575)
2024-12-29 18:03:16,205 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52574)
2024-12-29 18:03:16,210 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52576)
2024-12-29 18:03:16,224 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52578)
2024-12-29 18:03:16,228 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52577)
2024-12-29 18:03:16,230 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52581)
2024-12-29 18:03:16,231 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52580)
2024-12-29 18:03:16,232 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52579)
2024-12-29 18:03:16,235 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52582)
2024-12-29 18:03:16,235 [MainThread ] [INFO ] Got client at ('128.101.167.123', 52583)
2024-12-29 18:03:16,236 [MainThread ] [INFO ] Total secondary nodes connected to me = 15
2024-12-29 18:03:17,253 [MainThread ] [INFO ] sent neighbor info to all the secondary nodes
2024-12-29 18:03:17,256 [MainThread ] [INFO ] sent neighbor info to all the secondary nodes
2024-12-29 18:03:17,668 [Thread-31 ] [INFO ] Bound to ('160.94.65.73', 7200) and waiting for UDP packet-based measurements
```

Figure A.3: Backend server (or Demo Node)-console log at the startup.

In this example, the backend server (or Demo Node) is connected to a network of 15 distributed agents. Here, the timestamps, number of secondary nodes, IP address, and port number may all change depending on the configuration. If startup is unsuccessful, error messages will be displayed and execution will terminate.

Executing `python secondary_node.py` in a terminal will start the module. If startup is successful, the following messages should be seen:

```
2024-12-29 18:03:16,090 [MainThread ] [INFO ] Starting script...
2024-12-29 18:03:16,090 [MainThread ] [INFO ] Connecting to the primary...
2024-12-29 18:03:16,123 [MainThread ] [INFO ] Successfully connected to the primary
2024-12-29 18:03:16,123 [MainThread ] [INFO ] Sending an initial hello to primary
2024-12-29 18:03:16,156 [MainThread ] [INFO ] My assigned ID is 8
2024-12-29 18:03:17,185 [MainThread ] [INFO ] my Nbr IDs
2024-12-29 18:03:17,187 [MainThread ] [INFO ] [4, 9]
2024-12-29 18:03:17,191 [MainThread ] [INFO ] I am bound to ('128.101.167.123', 52578)
```

Figure A.4: Distributed agent running optimization-console log at the startup.

In this example, the distributed agent running the **DC-DistADMM** algorithm is connected to the backed server. Here, the agent is ID number 8 out 15 connected nodes to the backend server, and nodes 4 and 9 are its neighbors. Here, the timestamps, assigned ID, the neighbor IDs, IP address and port number may all change depending on the configuration. If startup is unsuccessful, error messages will be displayed and execution will terminate.

Executing `python MeasurementLogging.py` in a terminal will start the module. If startup is successful, the following messages should be seen:

```
241229 18:06:00.818 - Bound to ('10.0.0.91', 1500) and waiting for UDP packet-based measurements
```

Figure A.5: Data storage controller-console log at the startup.

In this example, the data storage controller is running at the IP address 10.0.0.91 : 1500 and waiting for the incoming information sent via the UDP communication channels. Here, the timestamps, the IP address and port number may change depending on the configuration. If startup is unsuccessful, error messages will be displayed and execution will terminate.