UNIVERSITY OF CALIFORNIA SAN DIEGO

Distributed Newton-like Algorithms and Learning for Optimized Power Dispatch

A dissertation submitted in partial satisfaction of the
requirements for the degree of Doctor of Philosophy

in

Mechanical Engineering

by

Tor Anderson

Committee in charge:

Professor Sonia Martínez, Chair
Professor Jorge Cortés
Professor Miroslav Krstić
Professor Jiawang Nie
Professor Behrouz Touri

2020

The Dissertation of Tor Anderson is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California San Diego

2020

DEDICATION

To my loving parents, Brian and Karen, my sister, Louise,

and to the memory of my dear friend, Marcus.

EPIGRAPH

I think that it is a relatively good approximation to truth – which is much too complicated to allow anything but approximations – that mathematical ideas originate in empirics, although the genealogy is sometimes long and obscure. But, once they are so conceived, the subject begins to live a peculiar life of its own and is better compared to a creative one, governed by almost entirely aesthetical motivations, than to anything else and, in particular, to an empirical science. (. . .) In any event, whenever this stage is reached, the only remedy seems to me to be rejuvenating return to the source: the reinjection of more or less directly empirical ideas. I am convinced that this was a necessary condition to conserve the freshness and the vitality of the subject and that this will remain equally true in the future.

– John von Neumann

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I extend the maximum amount of thanks and appreciation to my advisor, Sonia Martínez. On a personal level, Sonia is kind, funny, and understanding, and professionally, she is insightful, diligent, and above all else, patient. I credit her for seeding what would become the broad theme of this thesis in the early stages of my Ph.D. work, and I expect to reap the benefits of the practices, habits, and modes of thinking that she instilled in me for years to come.

Next, my thanks goes out to the other members of my committee: Professors Jorge Cortés, Miroslav Krstić, Jiawang Nie, and Behrouz Touri; it is a privilege to benefit from their volunteered time and feedback during the development and submission of this work. Further, I would be remiss not to mention those who previously supervised me and molded my professional development in some way. In reverse chronological order, I extend thanks to: Jorge Cortés (again) and Jan Kleissl for their supervision and insight during the work that became the content of Chapter 6 of this thesis; Sonja Glavaski, in part for her vision of the NODES project which directly inspired the work of Chapter 6, but also for supervising and mentoring me at ARPA-e during the summer between my undergraduate and graduate study; James (Mike) Sigler, for being my first boss in engineering and exposing me to how the sauce is made; and, finally, Elias Lemon, my first boss, who constantly trusted and challenged me in my first "real" job to take on more than I thought I was capable of at a large martial arts studio.

I would now like to expand on the people mentioned in the dedication. My father, Brian, has been a steady source of inspiration and motivation throughout my life to pursue ideas that are interesting and challenging. Without his influence, I likely would not have ever considered attempting graduate study. My mother, Karen, has always been a grounding influence in my life. On the pragmatic side, I credit her for teaching me the writing habits that made this thesis possible, and on the counseling side, she can reliably talk me back into a relaxed and clear state-of-mind when pressure and obligations otherwise seem overwhelming. My sister, Louise, is a frequent reminder to me that problems I face are fairly universal, and that one's accomplishments are always relative to the lens through which one views them. The cliché "wise

beyond one's years" applies to people like her. Finally, my late friend Marcus was extremely formative; his friendship and shared affinity for math, science, and engineering in high school and undergraduate were irreplaceable. His continued encouragement throughout my graduate study was essential to completing this thesis.

Additional thanks go to current and former members of our group, including (but not limited to!) Ashish, Aaron, Erfan, Eduardo, Yifu, Chin-Yao, Dimitris, Miguel, Priyank, Pio, Dan, and Aamodh, for their friendship and, of course, frequent technical discussions. My late grandparents, Tor and Phyllis, were passionate educators, and my grandmother, Darlene, and my late grandfather, Ken, were successful entrepreneurs, so they also have directly and indirectly helped me get to this point. Lastly, I give thanks to my wonderful partner, Amie, whose companionship over the last year and a half has helped me get through this "home-run stretch," and to my adorable dog, Teddy, who can inject a quick shot of joy into my day at literally any moment.

The material in Chapter 2, in full, is a reprint of *Distributed Approximate Newton Algorithms and Weight Design for Constrained Optimization*, T. Anderson, C.Y. Chang and S. Martínez, Automatica, 109, article 108538, November 2019. A preliminary version of the work appeared in the proceedings of the Conference on Control Technology and Applications (CCTA), Mauna Lani, HI, 2017, pp. 632-637, as *Weight Design of Distributed Approximate Newton Algorithms for Constrained Optimization*, T. Anderson, C.Y. Chang and S. Martínez. The dissertation author was the primary investigator and author of these papers.

Chapter 3, in full, is being revised and prepared for submission to the Systems & Control Letters. It may appear as *Distributed Stochastic Nested Optimization via Cubic Regularization*, T. Anderson and S. Martínez. The dissertation author was the primary investigator and author of this paper.

The content in Chapter 4, in full, is provisionally accepted in Automatica. It is expected to appear as *Distributed Resource Allocation with Binary Decisions via Newton-like Neural Network Dynamics*, T. Anderson and S. Martínez. The dissertation author was the primary

investigator and author of this paper.

The material in Chapter 5, in full, is a reprint of *Maximizing Algebraic Connectivity of Constrained Graphs in Adversarial Environments*, T. Anderson, C.Y. Chang and S. Martínez, 2018 European Control Conference (ECC), Limassol, 2018, pp. 125-130. The dissertation author was the primary investigator and author of this paper.

Chapter 6, in full, is under revision for publication in IEEE Transactions on Smart Grid. It may appear as *Frequency Regulation with Heterogeneous Energy Resources: A Realization using Distributed Control*, T. Anderson, M. Muralidharan, P. Srivastava, H.V. Haghi, J. Cortés, J. Kleissl, S. Martínez and B. Washom. The dissertation author was one of three primary investigators and authors of this paper.

VITA

2020 Ph.D., University of California San Diego

2017 M.S., University of California San Diego

2015 B.S., University of Minnesota Twin Cities

PUBLICATIONS

1. *Distributed Stochastic Nested Optimization via Cubic Regularization* , T. Anderson and S. Martínez, In preparation for submission to Systems & Control Letters.

2. *Frequency Regulation with Heterogeneous Energy Resources: A Realization using Distributed Control*, T. Anderson, M. Muralidharan, P. Srivastava, H.V. Haghi, J. Cortés, J. Kleissl, S. Martínez and B. Washom, IEEE Transactions on Smart Grid, Under revision.

3. *Distributed Resource Allocation with Binary Decisions via Newton-like Neural Network Dynamics*, T. Anderson and S. Martínez, Automatica, Provisionally accepted.

4. *Distributed Approximate Newton Algorithms and Weight Design for Constrained Optimization*, T. Anderson, C.Y. Chang and S. Martínez, Automatica, 109, article 108538, November 2019.

5. *Maximizing Algebraic Connectivity of Constrained Graphs in Adversarial Environments*, T. Anderson, C.Y. Chang and S. Martínez, 2018 European Control Conference (ECC), Limassol, 2018, pp. 125-130.

6. *Weight Design of Distributed Approximate Newton Algorithms for Constrained Optimization*, T. Anderson, C.Y. Chang and S. Martínez, 2017 IEEE Conference on Control Technology and Applications (CCTA), Mauna Lani, HI, 2017, pp. 632-637.

ABSTRACT OF THE DISSERTATION

Distributed Newton-like Algorithms and Learning for Optimized Power Dispatch

by

Tor Anderson

Doctor of Philosophy in Mechanical Engineering

University of California San Diego, 2020

Professor Sonia Martínez, Chair

This thesis explores a particular class of distributed optimization methods for various separable resource allocation problems, which are of high interest in a wide array of multi-agent settings. A distinctly motivating application for this thesis is real-time power dispatch of distributed energy resources for providing frequency control in a distribution grid or microgrid with high renewable energy penetration. In this application, it is paramount that agent data be shared as sparsely as possible in the interest of conserving user privacy, and it is required that algorithms scale gracefully as the network size increases to the order of thousands or millions of resources and devices. Distributed algorithms are naturally well-poised to address these challenges, in contrast to more traditional centralized algorithms which scale poorly and require

global access to information.

The class of distributed optimization methods explored here can be broadly described as *Newton-like* or *second-order*, implying utilization of second-derivative information of the cost functions, in contrast to well-studied *gradient-based* or *first-order* methods. We consider three formulations of separable resource-allocation problems and develop a Newton-like algorithm for each. First, the cost function is given by the sum of local agent costs, supplemented with individual linear box constraints and a global matching-constraint in which the sum of agent states must equal a prescribed constant. Second, we consider a stochastic, nested scenario, in which batches of realizations of problems of the first type must be used to gradually learn the optimal value of a parameter which is coupled with the agent costs. Third, we further constrain the agent states to be binary, and we embed the global matching-constraint as a squared penalty term in the cost. The analysis and simulation studies in the subsequent chapters demonstrate the advantages of our approaches over existing methods; most commonly, we note that convergence rates are substantially improved. We supplement our algorithm development for these three problem formulations with a network design technique, in which we can construct a maximally-connected network by adding some edges to the underlying communication graph, and a real demonstration of distributed algorithms on a large set of heterogeneous devices on the UC San Diego microgrid.

Introduction

The natural universe is made up of and governed by distributed interactions. This is evidenced on every relevant scale and setting: particle interactions, signal exchanges between neurons, cooperation between biological organs, tight-knit and mass-scale social interactions, the interacting physics of distant Earth biomes, and gravitational forces between planets, stars, and galaxies. These distributed physical dynamics are *prescribed* by the universe (sometimes indirectly, e.g. via human evolution in the social dynamics case); however, with the advent of modern technology, current and future engineers can benefit from the *imposition* of distributed intelligence and algorithms. It may be the case that this technological imposition is not only inevitable, but that we are already deeply in the midst of it.

Let us take a step back by considering a specific analogy to biological evolution. It is estimated that Earth's first prokaryotic life (cells *without* a nucleus) originated about 3.5–3.8 billion years ago, while the first eukaryotic life (cells *with* a nucleus) is estimated to have evolved around 1.7–2.2 billion years ago.[1] The implication is that the time-scale of progression from eukaryotic cells to modern multi-cellular life, in all its richness and complexity, is roughly equal to the time-scale of the comparatively miniscule progression of developing the cell nucleus. Consider, then, the fairly-new (in the scope of human history) technological development of semi-conductor based computers. Most present-day algorithms and computer intelligence are designed for *centralized* architectures, but the technological shift towards decentralized and distributed computing and information systems is undeniable: one need look no further than the

[1] Source: Carl Woese, J Peter Gogarten, "When did eukaryotic cells (cells with nuclei and other internal organelles) first evolve? What do we know about how they evolved from earlier life-forms?" Scientific American, October 21, 1999. https://www.scientificamerican.com/article/when-did-eukaryotic-cells/

recent proliferation of blockchain technologies, advancements in cloud computing and storage architectures, and the abundance of personal cell phones and daily influence from social media networks. Hence, if any kind of analogy can be drawn to biological evolution, computers and intelligence systems seem to find themselves on the precipice of (or already in the midst of) a dramatic shift toward "multi-cellular" architectures. State-of-the-art computers in the coming decades might barely resemble their centralized technological ancestors, similarly to how modern multi-cellular life barely resembles its eukaryotic ancestors.

The above discussion can serve as a philosophical motivation for the ideas and algorithms that are developed in this thesis, though more pragmatic and immediate motivations exist.[2] In particular, we study three closely related formulations of what is broadly referred to as the distributed resource allocation problem. Namely, (i) a nominal convex formulation, in which agent states can take continuous values in a convex set; (ii) a nested stochastic formulation, in which problems of type (i) are nested in a broader stochastic, nonconvex optimization which aims to optimization a parameterization or design variable over realizations of (i); and (iii) a further-constrained instance of (i) in which agent states must belong to a binary set. The approaches for each of (i), (ii), (iii) vary significantly, but they each possess the unifying theme of being *distributed* and using *Newton-like* updates, i.e. the updates utilize second-derivative information of the local agent costs.

The aforementioned work for (i), (ii), (iii) is contained in Chapters 2, 3, and 4, respectively. In Chapter 5, we supplement the results with a design technique for adding edges to a communication graph, and in Chapter 6 we describe a demonstration that we performed with distributed algorithms performing a frequency control application on the UC San Diego microgrid. We give more specific descriptions and motivation tailored to each chapter in the following subsections.

[2]For more concrete examples in present-day engineering, the reader can refer to the following subsections and the "Biological Comments" section of each chapter.

Nominal Convex Formulation

Networked systems endowed with distributed, multi-agent intelligence are becoming pervasive in modern infrastructure systems such as power, traffic, and large-scale distribution networks. However, these advancements lead to new challenges in the coordination of the multiple agents operating the network, which are mindful of the network dynamics, and subject to partial information and communication constraints. To this end, distributed convex optimization is a rapidly emerging field which seeks to develop useful algorithms to manage network resources in a scalable manner. Motivated by the rapid emergence of distributed energy resources, a problem that has recently gained large attention is that of economic dispatch. In this problem, a total net load constraint must be satisfied by a set of generators which each have an associated cost of producing electricity. However, the existing distributed techniques to solve this problem are often limited by rate of convergence. Motivated by this, we investigate the design of topology weighting strategies that build on the Newton method and lead to improved convergence rates.

Nested Stochastic Formulation

As applications emerge which are high dimensional and described by large data sets, the need for powerful optimization tools has never been greater. In particular, agents in distributed settings are commonly given a global optimization task where they must sparingly exchange local information with a small set of neighboring agents for the sake of privacy and robust scalability. This architecture can, however, slow down convergence compared to centralized ones, which is concerning if obtaining the iterative update information is costly. Gradient-based methods are commonly used due to their simplicity, but they tend to be vulnerable to slow convergence around saddle points. Newton-based methods use second-derivative information to improve convergence, but they are still liable to be slow in areas where higher order terms dominate the objective function and even unstable when the Hessian is ill conditioned. A powerful tool for combating these Newton-based vulnerabilities is imposing a cubic regularization on the

function's second-order Taylor approximation, but the current work on this technique does not unify *distributed*, *stochastic*, and *nonconvex* elements. Motivated by this, we study the adaptation of the Stochastic Cubic Regularized Newton approach to solve a distributed nested optimization problem.

Binary Formulation

There has been an explosion of literature surrounding the design of distributed algorithms for convex optimization problems and how these pertain to the operation of future power grids. A common assumption of these algorithms is the property of *convexity*, which lends itself to provably optimal solutions which are *scalable* and *fast*. However, some settings give rise to nonconvex decision sets. For example, in an optimal power dispatch setting, devices available for providing load-side frequency regulation such as HVAC systems, household appliances, and manufacturing systems are often limited to discrete on/off operational modes. It is even preferable to charge populations of electric vehicles in a discrete on/off manner due to nonlinear battery chemistries. The available tools in optimization for these nonconvex settings are less mature, and when considering a distributed setting in which devices act as agents that collectively compute a solution over a sparse communication graph, the available tools are significantly less developed. With this in mind, we are motivated to develop a *scalable*, *fast* approach for these binary settings which is amenable to a *distributed* implementation.

Topology Design

Multi-agent systems are pervasive in new technology spaces such as power networks with distributed energy resources like solar and wind, mobile sensor networks, and large-scale distribution systems. In such systems, communication amongst agents is paramount to the propagation of information, which often lends itself to robustness and stability of the system. Network connectivity is well studied from a graph-theoretic standpoint, but the problem of

designing topologies when confronted by engineering constraints or adversarial attacks is not well addressed by current works. We are motivated to study the NP-hard graph design problem of adding edges to an initial topology and to develop a method to solve it which has both improved performance and allows for direct application to the aforementioned constrained and adversarial settings.

Application: Frequency Regulation with Heterogeneous Energy Resources

Many recent efforts seek to integrate renewable energy resources with the power grid to reduce the carbon footprint. The high variability associated with wind and solar power can be balanced using distributed energy resources (DERs) providing ancillary services such as frequency regulation. Consequently, there is a growing interest among market operators in DER aggregations with flexible generation and load capabilities to balance fluctuations in grid frequency and minimize area control errors (ACE). The fast ramping rate and minimal marginal standby cost put many DERs at an advantage against conventional generators and make them suitable for participation in the frequency regulation market.

The fast ramping rates reduce the required power capacity of DERs to only 10% of an equivalent generator to balance a frequency drop within 30s [79]. However, most individual DERs have small capacities, typically on the order of kWs compared to 10 s of MW for conventional frequency control resources. Commanding the required thousands to millions of DERs to replace existing frequency regulation resources over a large balancing area entails aggregating DERs that are distributed at end points all over the grid on customer premises. The dynamic nature, large number, and distributed location of DERs requires coordination. This is in contrast to existing frequency regulation [63] implementation with conventional energy resources. For example, CAISO requires all generators to submit their bids once per regulation interval. Then, the setpoints are assigned centrally to all resources every 2-4 sec without any consideration of

operational costs [19]. While distributed control has the potential to enable DER participation in the frequency regulation market (e.g., [91]), there is a general lack of large-scale testing to prove its effectiveness for widespread adoption by system operators. The 2017 National Renewable Energy Laboratory Workshop on Autonomous Energy Grids [64] concluded that "A major limitation in developing new technologies for autonomous energy systems is that there are no large-scale test cases (...). These test cases serve a critical role in the development, validation, and dissemination of new algorithms".

Chapter 1

Notation and Preliminaries

1.1 Notation

Let $\mathbb{R}$ and $\mathbb{R}_+$ denote the set of real and positive real numbers, respectively, and let $\mathbb{N}$ denote the set of natural numbers. For a vector $x \in \mathbb{R}^n$, we denote by x_i the i^{th} entry of x. For a matrix $A \in \mathbb{R}^{n\times m}$, we write A_i as the i^{th} row of A and A_{ij} as the element in the i^{th} row and j^{th} column of A, and for A square, $A^\dagger$ is the Moore-Penrose pseudoinverse of A. The transpose of a vector or matrix is denoted by $x^\top$ and $A^\top$, respectively. We use the shorthand notations $\mathbf{1}_n = [1,\ldots,1]^\top \in \mathbb{R}^n$, $\mathbf{0}_n = [0,\ldots,0]^\top \in \mathbb{R}^n$, I_n to denote the $n \times n$ identity matrix, and define $\mathbb{I}_n \triangleq I_n - \dfrac{\mathbf{1}_n\mathbf{1}_n^\top}{n}$. We refer to this matrix as a *pseudo-identity matrix*; note that $\text{null}\,(\mathbb{I}_n) = \text{span}\{\mathbf{1}_n\}$. The standard inner product of two vectors $x, y \in \mathbb{R}^n$ is written $\langle x, y\rangle$, and $x \perp y$ indicates $\langle x, y\rangle = 0$. The orthogonal complement to a span of vectors a_i is written $\text{span}\{a_i\}^\perp$, meaning $x \perp y, \forall x \in \text{span}\{a_i\}, \forall y \in \text{span}\{a_i\}^\perp$. For a real-valued function $f : \mathbb{R}^n \to \mathbb{R}$, the gradient vector of f with respect to x is denoted by $\nabla_x f(x)$ and the Hessian matrix with respect to x by either $\nabla_{xx} f(x)$ or $\nabla^2 f(x)$. When $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ takes multiple arguments, we specify the differentiation variable(s) as a subscript of ∇. Cartesian products of sets are denoted by a superscript, for example, $\{0,1\}^n = \{0,1\} \times \cdots \times \{0,1\}$. The positive (semi) definiteness and negative (semi) definiteness of a matrix $A \in \mathbb{R}^{n\times n}$ is indicated by $A \succ 0$ and $A \prec 0$ (resp. $A \succeq 0$ and $A \preceq 0$). The same symbols are used to indicate componentwise inequalities on vectors of equal sizes. The set of eigenvalues of a symmetric matrix $A \in \mathbb{R}^{n\times n}$ is ordered as $\mu_1(A) \leq \cdots \leq \mu_n(A)$ with associated

eigenvectors $v_1, \ldots, v_n \in \mathbb{R}^n$. An orthogonal matrix $T \in \mathbb{R}^{n\times n}$ has the property $T^\top T = TT^\top = I_n$ and $T^\top = T^{-1}$. For a finite set $\mathscr{S}$, $|\mathscr{S}|$ is the cardinality of the set. The standard Euclidean norm and the Kronecker product are indicated by $\|\cdot\|$, $\otimes$, respectively. We denote elementwise operations on vectors $x, y \in \mathbb{R}^n$ as $(x_i y_i)_i = (x_1 y_1, \ldots, x_n y_n)^\top$, $(x_i)_i^2 = (x_1^2, \ldots, x_n^2)^\top$, $(c/x_i)_i = (c/x_1, \ldots, c/x_n)^\top$, $\log(x_i)_i = (\log(x_1), \ldots, \log(x_n))^\top$, and $(e^{x_i})_i = (e^{x_1}, \ldots, e^{x_n})^\top$. The notation $\mathrm{diag}(x)$ indicates the diagonal matrix with entries given by elements of x, and $\mathscr{B}(x, \eta)$ denotes the closed ball of radius η centered at x. Probabilities and expectations are indicated by $\mathbb{P}$ and $\mathbb{E}$, respectively. The Dirac delta function centered at $a \in \mathbb{R}$ is denoted by δ_a, and the uniform distribution on $[a, b]$ is denoted by $\mathscr{U}[a, b]$. We define the projection

$$[u]_v^+ := \begin{cases} u, & v > 0, \\ \max\{0, u\}, & v \leq 0. \end{cases}$$

1.2 Graph Theory

We refer to [47] as a supplement for the concepts we describe throughout this section. A network of agents is represented by a graph $\mathscr{G} = (\mathscr{N}, \mathscr{E})$, assumed undirected, with a node set $\mathscr{N} = \{1, \ldots, n\}$ and edge set $\mathscr{E} \subseteq \mathscr{N} \times \mathscr{N}$. The edge set $\mathscr{E}$ has elements $(i, j) \in \mathscr{E}$ for $j \in \mathscr{N}_i$, where $\mathscr{N}_i \subset \mathscr{N}$ is the set of neighbors of agent $i \in \mathscr{N}$. The union of neighbors to each agent $j \in \mathscr{N}_i$ are the 2-hop neighbors of agent i, and denoted by $\mathscr{N}_i^2$. More generally, $\mathscr{N}_i^p$, or set of p-hop neighbors of i, is the union of neighbors of agents in $\mathscr{N}_i^{p-1}$. In Chapter 2, we consider *weighted* edges for the sake of defining the graph Laplacian; the role of edge weightings and the associated design problem is described in Section 2.3. The graph $\mathscr{G}$ then has a *weighted*

Laplacian $L \in \mathbb{R}^{n\times n}$ defined as

$$L_{ij} = \begin{cases} -w_{ij}, & j \in \mathscr{N}_i, j \neq i, \\ w_{ii}, & j = i, \\ 0, & \text{otherwise}, \end{cases}$$

with weights $w_{ij} = w_{ji} > 0, j \in \mathscr{N}_i, j \neq i$, and total incident weight w_{ii} on $i \in \mathscr{N}$, $w_{ii} = \sum_{j\in\mathscr{N}_i} w_{ij}$. From Chapter 3 onward, L is taken to be *unweighted*, i.e. $w_{ij} = 1, j \in \mathscr{N}_i, j \neq i$. Evidently, L has an eigenvector $v_1 = \mathbf{1}_n$ with an associated eigenvalue $\mu_1 = 0$, and $L = L^\top \succeq 0$. The graph is connected i.f.f. 0 is a simple eigenvalue, i.e. $0 = \mu_1 < \mu_2 \leq \cdots \leq \mu_n$, and it is well known that the multiplicity of the zero eigenvalue is equal to the number of connected components in the graph [47].

The Laplacian L can be written via its incidence matrix $E \in \{-1,0,1\}^{|\mathscr{E}|\times n}$ and a diagonal matrix $X \in \mathbb{R}_+^{|\mathscr{E}|\times|\mathscr{E}|}$ whose entries are weights w_{ij}. Each row of E is associated with an edge (i,j) whose i^{th} element is 1, j^{th} element is -1, and all other elements zero. Then, $L = E^\top XE$.

1.3 Schur Complement

The following lemma will be used in the sequel.

Lemma 1. *[114](**Matrix Definiteness via Schur Complement**). Consider a symmetric matrix M of the form*

$$M = \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}.$$

If C is invertible, then the following properties hold:

(1) $M \succ 0$ if and only if $C \succ 0$ and $A - BC^{-1}B^\top \succ 0$.

(2) If $C \succ 0$, then $M \succeq 0$ if and only if $A - BC^{-1}B^\top \succeq 0$.

1.4 Taylor Series Expansion for Matrix Inverses

A full-rank matrix $A \in \mathbb{R}^{n\times n}$ has a matrix inverse, A^{-1}, which is characterized by the relation $AA^{-1} = I_n$. In principle, it is not straightforward to compute this inverse via a distributed algorithm. However, if the eigenvalues of A satisfy $|1-\mu_i(A)| < 1, \forall\, i \in \mathcal{N}$, then we can employ the Taylor expansion to compute its inverse [92]:

$$A^{-1} = \sum_{p=0}^{\infty} (I_n - A)^p.$$

To quickly see this holds, substitute $B = I_n - A$, multiply both sides by $I_n - B$ and reason with $\lim_{p\to\infty}$. Note that, if the sparsity structure of A represents a network topology, then traditional matrix inversion techniques such as Gauss-Jordan elimination still necessitate all-to-all communication. However, agents can communicate and compute locally to obtain each term in the previous expansion. If A is normal, it can be seen via the diagonalization of $I_n - A$ that the terms of the sum become small as p increases due to the assumption on the eigenvalues of A [43]. The convergence of these terms is exponential and limited by the slowest converging mode, i.e. $\max|1-\mu_i(A)|$.

We can compute an approximation of A^{-1} in finite steps by computing and summing the terms up to the q^{th} power. We refer to this approximation as a *q-approximation* of A^{-1}.

1.5 Cubic-Regularized Newton Algorithm

We now provide a brief background on the Cubic-Regularized Newton method, which will be referred to in Chapter 3. See [78] and [25,26] for more information. Consider the problem of minimizing a (possibly nonconvex) function $f : \mathbb{R}^d \to \mathbb{R}$:

$$\min_{x\in\mathbb{R}^d} f(x). \tag{1.1}$$

As nonconvex optimization is typically intractable in high dimensions, a typical objective is to converge to an ε-second-order stationary point.

Definition 1. ***(ε-Second-Order Stationary Point).*** *A point $x^\star$ is an ε-second-order stationary point of f if*

$$\|\nabla_x f(x^\star)\| \leq \varepsilon \quad \textit{and} \quad \lambda_{min}(\nabla^2_{xx} f(x^\star)) \geq -\sqrt{\rho\varepsilon}. \tag{1.2}$$

Here, ρ is commonly taken to be the Lipschitz constant of $\nabla^2_{xx} f$, which we will formalize in Section 3.2.

One useful iterative model for minimizing $f(x^k)$ when the function is strictly convex at the current iterate x^k (or, more accurately, if it is strictly convex on some neighborhood of x^k) is descent on a second-order Taylor expansion around x^k:

$$x^{k+1} = \operatorname*{argmin}_x \left\{ f(x^k) + (x-x^k)^\top \nabla f(x^k) + \frac{1}{2}(x-x^k)^\top \nabla^2 f(x^k)(x-x^k) \right\} = x^k - \nabla^{-2} f(x^k) \nabla f(x^k). \tag{1.3}$$

This closed form expression for x^{k+1} breaks down when f is nonconvex due to some eigenvalues of $\nabla^2 f(x^k)$ having negative sign. Further, when $\nabla^2 f(x^2)$ is nearly-singular, the update becomes very large in magnitude and can lead to instability. For this reason, consider amending the second-order model with a cubic-regularization term, to obtain the cubic-regularized, third-order model of f at x^k as:

$$m_k(x) \triangleq \left\{ f(x^k) + (x-x^k)^\top \nabla f(x^k) + \frac{1}{2}(x-x^k)^\top \nabla^2 f(x^k)(x-x^k) + \frac{\rho}{6}\|x-x^k\|^3 \right\}. \tag{1.4}$$

The update is naturally given by a minimizer to this model: $x^{k+1} \in \operatorname*{argmin}_x m_k(x)$. Unfortunately, this model does not beget a closed-form minimizer as in (1.3), nor is it convex if f is not convex. The model does, however, become convex for x very far from x^k, which can be seen by computing the Hessian of m_k as $\nabla^2 m_k(x) = \nabla^2 f(x^k) + \rho\|x-x^k\| I_n$. Additionally, m_k is an *over-estimator* for f, i.e. $m_k(x) \geq f(x), \forall x$. This is seen by considering the cubic term and recalling Lipschitz properties of $\nabla^2 f$; we describe this observation in more detail in Chapter 3. Therefore, m_k

possesses some advantages over other simpler submodels as it possesses properties of a more standard Newton-based, second-order model while being sufficiently conservative.

Finally, [24] recently showed that simply initializing $x = x^k - r\nabla f(x^k)/\|\nabla f(x^k)\|$ for $r \geq 0$ is sufficient to show that gradient descent on m_k converges to the global minimizer of (1.4) (under light conditions on r and the gradient step size).

1.6 PT-Inverse

Next, we introduce the Positive-definite Truncated inverse (PT-inverse) and its relevance to nonconvex Newton methods.

Definition 2 ([85]). ***(PT-inverse).*** *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix with an orthonormal basis of eigenvectors $Q \in \mathbb{R}^{n\times n}$ and diagonal matrix of eigenvalues $\Lambda \in \mathbb{R}^{n\times n}$. Consider a constant $m > 0$ and define $|\Lambda|_m \in \mathbb{R}^{n\times n}$ by:*

$$(|\Lambda|_m)_{ii} = \begin{cases} |\Lambda_{ii}|, & |\Lambda_{ii}| \geq m, \\ m, & \textit{otherwise.} \end{cases}$$

The PT-inverse of A with parameter m is defined by $(|A|_m)^{-1} = Q^\top (|\Lambda|_m)^{-1} Q \succ 0$.

The PT-inverse operation flips the sign on the negative eigenvalues of A and truncates near-zero eigenvalues to a (small) positive value m before conducting the inverse. Effectively, this generates a positive definite matrix bounded away from zero to be inverted, circumventing near-singular cases. In terms of computational complexity, it is on the order of standard eigendecomposition (or more generally, singular value decomposition), which is roughly $O(n^3)$ [80]. However, we note in Section 4.4 that the matrix to be PT-inverted is diagonal, which is $O(n)$.

The PT-inverse is useful for nonconvex Newton approaches [85] in the following sense: first, recall that the Newton descent direction of f at x is computed as $-(\nabla_{xx} f(x))^{-1} \nabla_x f(x)$. For f strictly convex, it holds that $\nabla_{xx} f(x) \succ 0$ and the Newton direction is well defined and

decreases the cost. For (non-strictly) convex or nonconvex cases, $\nabla_{xx} f(x)$ will be singular, indefinite, or negative definite. A PT-inverse operation remedies these cases and preserves the descent quality of the method. Additionally, saddle points are a primary concern for first-order methods in nonconvex settings [36], and the Newton flavor endowed by the PT-inverse effectively performs a change of coordinates on saddles with "slow" unstable manifolds compared to the stable manifolds. We discuss this further in Section 4.3.

1.7 Set Theory

A limit point p of a set P is a point such that any neighborhood $\mathscr{B}_\varepsilon(p)$ contains a point $p' \in \mathsf{P}$. A set is closed if it contains all of its limit points, it is bounded if it is contained in a ball of finite radius, and it is compact if it is both closed and bounded. Let $\mathscr{A}_i = \{p \mid a_i^\top p \geq b_i\}$ be a closed half-space and $\mathsf{P} = \mathscr{A}_1 \cap \cdots \cap \mathscr{A}_r \subset \mathbb{R}^m$ be a finite intersection of closed half-spaces. If P is compact, we refer to it as a polytope. Consider a set of points $\mathscr{F} = \{p \in \mathsf{P} \mid a_i^\top p = b_i, i \in \mathscr{I} \subseteq \{1,\ldots,r\}; a_j^\top p \geq b_j, j \in \{1,\ldots,r\} \setminus \mathscr{I}\}$. Let $h = \dim(\operatorname{span}\{a_i\})$ be the dimension of the subspace spanned by the vectors $\{a_i\}_{i\in\mathscr{I}}$. Then, we refer to $\mathscr{F}$ as an $(m-h)$-dimensional face of P. Lastly, denote the affine hull of $\mathscr{F}$ as $\operatorname{aff}(\mathscr{F}) = \{p+w \mid p, w \in \mathbb{R}^m, p \in \mathscr{F}, w \perp \operatorname{span}\{a_i\}_{i\in\mathscr{I}}\}$ and define the relative interior of $\mathscr{F}$ as $\operatorname{relint}(\mathscr{F}) = \{p \mid \exists \varepsilon > 0 \text{ s. t. } \mathscr{B}_\varepsilon(p) \cap \operatorname{aff}(\mathscr{F}) \subset \mathscr{F}\}$.

Chapter 2

Distributed Approximate Newton Algorithms and Weight Design for Constrained Optimization

Motivated by economic dispatch and linearly-constrained resource allocation problems, this chapter proposes a class of novel DISTRIBUTED APPROX-NEWTON algorithms that approximate the standard Newton optimization method. We first develop the notion of an optimal edge weighting for the communication graph over which agents implement the second-order algorithm, and propose a convex approximation for the nonconvex weight design problem. This weight design formulates to a nonconvex bilinear optimization, and we propose a convex approximation that is loosely based on completing the square to compute adequate solutions. We next build on the optimal weight design to develop a DISCRETE DISTRIBUTED APPROX-NEWTON algorithm which converges linearly to the optimal solution for economic dispatch problems with unknown cost functions and relaxed local box constraints. For the full box-constrained problem, we develop a CONTINUOUS DISTRIBUTED APPROX-NEWTON algorithm which is inspired by first-order saddle-point methods and rigorously prove its convergence to the primal and dual optimizers. A main property of each of these distributed algorithms is that they only require agents to exchange constant-size communication messages, which lends itself to scalable implementations. Simulations demonstrate that the DISTRIBUTED APPROX-NEWTON algorithms with our weight design have superior convergence properties compared to existing weighting

strategies for first-order saddle-point and gradient descent methods.

2.1 Bibliographical Comments

The Newton method for minimizing a real-valued multivariate objective function is well characterized for centralized contexts in [17]. Another centralized method for solving general constrained convex problems by seeking the saddle-point of the associated Lagrangian is developed in [34]. This method, which implements a saddle-point dynamics is attractive because its convergence properties can be established. Other first-order or primal-dual based methods for approaching distributed optimization include [22, 39, 53, 70]. However, these methods typically do not incorporate second-order information of the cost function, which compromises convergence speeds. The notion of computing an *approximate* Newton direction in distributed contexts has gained popularity recently, such as [74] and [105, 106]. In the former work, the authors propose a method which uses the Taylor series expansion for inverting matrices. However, it assumes that each agent keeps an estimate of the entire decision variable, which does not scale well in problems where this variable dimension is equal to the number of agents in the network. Additionally, the optimization is unconstrained, which helps to keep the problem decoupled but is narrower in scope. The latter works pose a separable optimization with an equality constraint characterized by the incidence matrix. The proposed method may be not directly applied to networks with constraints that involve the information of all agents. The papers [23, 59, 112] incorporate multi-timescaled dynamics together with a dynamic consensus step to speed up the convergence of the agreement subroutine. These works only consider uniform edge weights, while sophisticated design of the weighting may improve the convergence. In [107], the Laplacian weight design problem for separable resource allocation is approached from a DISTRIBUTED GRADIENT DESCENT perspective. Solution post-scaling is also presented, which can be found similarly in [75] and [86] for improving the convergence of the Taylor series expression for matrix inverses. In [88], the authors consider edge weight design to minimize the

spectrum of Laplacian matrices. However, in the Newton descent framework, the weight design problem formulates as a nonconvex bilinear problem, which is challenging to solve. Overall, the current weight-design techniques that are computable in polynomial time are only mindful of first-order algorithm dynamics. A second-order approach has its challenges, which manifest themselves in a bilinear design problem and more demanding communication requirements, but using second-order information is more heedful of the problem geometry and leads to faster convergence speeds.

Statement of Contributions

In this chapter, we propose a novel framework to design a weighted Laplacian matrix that is used in the solution to a multi-agent optimization problem via sparse approximated Newton algorithms. Motivated by economic dispatch, we start by formulating a separable resource allocation problem subject to a global linear constraint and local box constraints, and then derive an equivalent form without the global constraint by means of a Laplacian matrix, which is well suited for a distributed framework. We use this to motivate weighting design of the elements of the Laplacian matrix and formulate this problem as a bilinear optimization. We develop a convex approximation of this problem whose solution can be computed offline in polynomial time. A bound on the *best-case* solution of the original bilinear problem is also given.

We aim to bridge the gap between classic Newton and DISTRIBUTED APPROX-NEWTON methods. To do this, we first relax the box constraints and develop a class of constant step-size discrete-time algorithms. The Newton step associated with the unconstrained optimization problem do not inherit the same sparsity as the distributed communication network. To address this issue, we consider approximations based on a Taylor series expansion, where the first few terms inherit certain level of sparsity as prescribed by the Laplacian matrix. We analyze the approximate algorithms and show their convergence for any truncation of the series expansion.

We next study the original problem with local box constraints, which has never been

considered in the framework of a distributed Newton method, and present a novel continuous-time DISTRIBUTED APPROX-NEWTON algorithm. The convergence of this algorithm to the optimizer is rigorously studied and we give an interpretation of the convergence in the Lyapunov function sense. Furthermore, through a formal statement of the proposed DANA (DISTRIBUTED APPROX-NEWTON algorithm), we find several interesting insights on second-order distributed methods. We compare the results of our design and algorithm to a generic weighting design of DISTRIBUTED GRADIENT DESCENT (DGD) implementations in simulation. Our weighting design shows superior convergence to DGD.

2.2 Problem Statement

Motivated by the economic dispatch problem, in this section we pose the separable resource allocation problem that we aim to solve distributively. We reformulate it as an unconstrained optimization problem whose decision variable is in the span of the graph Laplacian, and motivate the characterization of a second-order Newton-inspired method.

Consider a group of agents $\mathcal{N}$, indexed by $i \in \mathcal{N}$, and a communication topology given by $\mathcal{G}$. Each agent is associated with a local convex cost function $f_i : \mathbb{R} \to \mathbb{R}$. These agents can be thought of as generators in an electricity market, where each function argument $x_i \in \mathbb{R}$, $i \in \mathcal{N}$ represents the power that agent i produces at a cost characterized by f_i. The economic dispatch problem aims to satisfy a global load-balancing constraint $\sum_{i=1}^{n} x_i = d$ for minimal global cost $f : \mathbb{R}^n \to \mathbb{R}$, where d is the total demand. In addition, each agent is subject to a local linear box constraint on its decision variable given by the interval $[\underline{x}_i, \overline{x}_i]$. Then, the economic dispatch

optimization problem is stated as:

$$\mathscr{P}1: \min_{x} \quad f(x) = \sum_{i=1}^{n} f_i(x_i) \tag{2.1a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_i = d, \tag{2.1b}$$

$$\underline{x}_i \leq x_i \leq \bar{x}_i, \quad i = \{1, \ldots, n\}. \tag{2.1c}$$

Distributed optimization algorithms based on a gradient descent approach to solve $\mathscr{P}1$ are available [115]. However, by only taking into account first-order information of the cost functions, these methods tend to be inherently slow. As for a Newton (second-order) method, the constraints make the computation of the descent direction non-distributed. To see this, consider only (2.1a)–(2.1b). Recall the unconstrainted Newton step defined as $x_{\text{nt}} := -\nabla_{xx} f(x)^{-1} \nabla_x f(x)$, see e.g. [17]. In this context, the equality constraint can be eliminated by imposing $x_n = d - \sum_{i=1}^{n-1} x_i$. Then, (2.1a) becomes $f(x) = \sum_{i=1}^{n-1} f_i(x_i) + f_n(d - \sum_{i=1}^{n-1} x_i)$. In general, the resulting Hessian $\nabla_{xx} f(x)$ is fully populated and its inverse requires all-to-all communication among agents in order to compute the second-order descent direction. If we additionally consider (2.1c), interior point methods are often employed, such as introducing a log-barrier function to the cost in (2.1a) [17]. The value of the log-barrier parameter is updated online to converge to a feasible solution, which exacerbates the non-distributed nature of this approach. This motivates the design of distributed Newton-like methods which are cognizant of (2.1b)–(2.1c).

We eliminate (2.1b) by introducing a network topology as encoded by a Laplacian matrix L associated with $\mathscr{G}$ and an initial condition $x^0 \in \mathbb{R}^n$ with some assumptions.

Assumption 1. ***(Undirected and Connected Graph).*** *The weighted graph characterized by L is undirected and connected, i.e. $L = L^\top$ and 0 is a simple eigenvalue of L.*

Assumption 2. ***(Feasible Initial Condition).*** *The initial state x^0 satisfies* (2.1b)*, i.e.*

$$\sum_{i=1}^{n} x_i^0 = d.$$

If the problem context does not lend itself well to satisfying Assumption 2, there is a distributed algorithmic solution to rectify this via dynamic consensus that can be found in [32] which could be modified for a Newton-like method. Given these assumptions, $\mathscr{P}1$ is equivalent to:

$$\mathscr{P}2 : \min_{z} \quad f(x^0 + Lz) = \sum_{i=1}^{n} f_i(x_i^0 + L_i z) \tag{2.2a}$$

$$\text{subject to} \quad \underline{x} - x^0 - Lz \preceq \mathbf{0}_n, \tag{2.2b}$$

$$x^0 + Lz - \overline{x} \preceq \mathbf{0}_n\,. \tag{2.2c}$$

Using the property that $\mathbf{1}_n$ is an eigenvector of L associated with the eigenvalue 0, we have that $\mathbf{1}_n^\top (x^0 + Lz) = d$. Newton descent for centralized solvers is given in [17]; in our distributed framework, the row space of the Laplacian is a useful property to address (2.1b).

Remark 1. ***(Relaxing Assumption 2).*** *The assumption on the initial condition can render the formulation vulnerable to implementation errors and cannot easily accommodate packet drops in a distributed algorithm. A potential workaround for this is outlined here. Consider, instead of* (2.1b) *in* $\mathscr{P}1$*, the n linear constraints:*

$$x + Lz = \overline{d}, \tag{2.3}$$

where $\overline{d} \in \mathbb{R}^n, \mathbf{1}_n^\top \overline{d} = d$ *and* (2.1b) *can be recovered by multiplying* (2.3) *from the left by* $\mathbf{1}_n$*. (As an aside, it may be desirable to impose sparsity on* $\overline{d}$ *so that only some agents need access to global problem data). Both* $x \in \mathbb{R}^n$ *and* $z \in \mathbb{R}^n$ *become decision variables, and agent i can verify the* i^{th} *component of* (2.3) *with one-hop neighbor information. Further, a distributed*

saddle-point algorithm can be obtained by assigning a dual variable to (2.3) *and proceeding as in [34].*

We provide a simulation justification for this approach in Section 2.6.4, although the analysis of robustness to perturbations and packet drops is ongoing and outside the scope of this chapter. For now we strictly impose Assumption 2.

We aim to leverage the freedom given by the elements of L in order to compute an approximate Newton direction to $\mathscr{P}2$. To this end, we adopt the following assumption.

Assumption 3. ***(Cost Functions).*** *The local costs f_i are twice continuously differentiable and strongly convex with bounded second-derivatives given by*

$$0 < \delta_i \leq \frac{\partial^2 f_i}{\partial x_i^2} \leq \Delta_i,$$

for every $i \in \mathscr{N}$ with given $\delta_i, \Delta_i \in \mathbb{R}_+$.

This assumption is common in other distributed Newton or Newton-like methods, e.g. [59, 74] and in classical convex optimization [17,77]. Assumption 3 is necessary to attain convergence in our computation of the Newton step/direction and to construct the notion of an optimal edge weighting L. We adopt the shorthands $H(x) := \nabla_{xx} f(x)$, $H_\delta := \mathrm{diag}(\delta)$, and $H_\Delta := \mathrm{diag}(\Delta)$ as the diagonal matrices with elements given by $\partial^2 f_i(x_i)/\partial x_i^2$, δ_i, and Δ_i, respectively.

Next, for the purpose of developing a distributed Newton-like method, we must slightly rethink the idea of inverting a Hessian matrix. By application of the chain rule, we have that $\nabla_{zz} f(x^0 + Lz) = LH(x^0 + Lz)L$. Clearly, $\nabla_{zz} f$ is non-invertible due to the smallest eigenvalue of L fixed at zero, a manifestation of the equality constraint in the original problem $\mathscr{P}1$. We instead focus on the $n-1$ nonfixed eigenvalues of $\nabla_{zz} f$ to employ the Taylor expansion outlined in Section 1.4. To this end, we project $LH(x^0 + Lz)L$ to the $\mathbb{R}^{(n-1)\times(n-1)}$ space with a coordinate transformation; the justification for this and relation to the traditional Newton method are made explicitly clear in Section 2.4. We seek a matrix $T \in \mathbb{R}^{n\times n}$ satisfying $T^\top T = I_n - \mathbf{1}_n \mathbf{1}_n^\top / n$ [43];

the particular matrix T we employ is given as

$$T = \begin{bmatrix} n\text{–}1+\sqrt{n} & -1 & \cdots & -1 & \frac{1}{\sqrt{n}} \\ -1 & \ddots & \cdots & \vdots & \\ \vdots & & \ddots & -1 & \vdots \\ -1 & \cdots & -1 & n\text{–}1+\sqrt{n} & \\ -1-\sqrt{n} & \cdots & \cdots & -1-\sqrt{n} & \frac{1}{\sqrt{n}} \end{bmatrix} \text{diag}\left(\begin{bmatrix} \rho \\ 1 \end{bmatrix}\right),$$

where $\rho = \sqrt{n(n+1+2\sqrt{n})}^{-1} \mathbf{1}_{n-1}$. This choice of T has the effect of projecting the null-space of the Hessian onto the n^{th} row and n^{th} column, which is demonstrated by defining $M(x) := JT^\top LH(x)LTJ^\top \in \mathbb{R}^{(n-1)\times(n-1)}$, where $J = \begin{bmatrix} I_{n-1} & \mathbf{0}_{n-1} \end{bmatrix}$. The matrix $M(x)$ shares its $n-1$ eigenvalues with the $n-1$ nonzero eigenvalues of $LH(x)L$ at each x, and $M(x)^{-1}$ is well defined, which provides us with a concrete notion of an inverse Hessian. We now adopt the following assumption.

Assumption 4. ***(Convergent Eigenvalues).*** *For any x, the eigenvalues of $I_{n-1} - M(x)$, corresponding to the $n-1$ smallest eigenvalues of $I_n - LH(x)L$, are contained in the unit ball, i.e. $\exists\, \varepsilon < 1$ such that*

$$-\varepsilon I_{n-1} \preceq I_{n-1} - M(x) \preceq \varepsilon I_{n-1}.$$

Technically speaking, we are only concerned with arguments of M belonging to the $n-1$ dimensional hyperplane $\{x^0 + Lz \mid z \in \mathbb{R}^n\}$, although we consider all $x \in \mathbb{R}^n$ for simplicity. In the following section, we address Assumption 4 (Convergent Eigenvalues) by minimizing ε via weight design of the Laplacian. By doing this, we aim to obtain a good approximation of M^{-1} from the Taylor expansion with small q, which lends itself well to the convergence of the distributed algorithms in Sections 2.4 and 2.5.

2.3 Weight Design of the Laplacian

In this section, we pose the nonconvex weight design problem on the elements of L, which formulates as a bilinear optimization to be solved by a central authority. To make this problem tractable, we develop a convex approximation and demonstrate that the solution is guaranteed to satisfy Assumption 4. Next, we provide a lower bound on the solution to the nonconvex problem. This gives a measure of performance for evaluating our approximation.

2.3.1 Formulation and Convex Approximation

Our approach hearkens to the intuition on the rate of convergence of the q-approximation of $M(x)^{-1}$. We design a weighting scheme for a communication topology characterized by L which lends itself to a scalable, fast approximation of a Newton-like direction. To this end, we minimize $\max_{i,x}|1-\mu_i(M(x))|$:

$$\mathscr{P}3: \quad \min_{\varepsilon,L} \quad \varepsilon \tag{2.4a}$$

$$\text{s.t.} \quad -\varepsilon I_{n-1} \preceq I_{n-1} - M(x) \preceq \varepsilon I_{n-1}, \forall x, \tag{2.4b}$$

$$L\mathbf{1}_n = \mathbf{0}_n,\ L \succeq 0,\ L = L^\top, \tag{2.4c}$$

$$L_{ij} \leq 0, j \in \mathscr{N}_i,\ L_{ij} = 0,\ j \notin \mathscr{N}_i\,. \tag{2.4d}$$

Naturally, $\mathscr{P}3$ must be solved offline by a central authority because it requires complete information about the local Hessians embedded in $M(x)$, in addition to being a semidefinite program for which distributed solvers are not mature. Even for a centralized solver $\mathscr{P}3$ is hard for a few reasons, the first being that (2.4b) is a function over all possible $x \in \mathbb{R}^n$. To reconcile with this, we invoke Assumption 3 on the cost functions and write $M_\delta = JT^\top LH_\delta LTJ^\top$ and

$M_\Delta = JT^\top LH_\Delta LTJ^\top$. Then, (2.4b) is equivalent to

$$-(\varepsilon_- + 1)I_{n-1} + M_\delta \preceq 0, \tag{2.5a}$$

$$(1-\varepsilon_+)I_{n-1} - M_\Delta \preceq 0, \tag{2.5b}$$

$$\varepsilon_- = \varepsilon_+, \tag{2.5c}$$

where the purpose of introducing ε_- and ε_+ will become clear in the discussion that follows.

The other difficult element of $\mathscr{P}3$ is the nonconvexity stemming from (2.5a)–(2.5b) being bilinear in L. There are path-following techniques available to solve bilinear problems of this form [54], but simulation results do not produce satisfactory solutions for problems of the form $\mathscr{P}3$. Instead, we aim to develop a convex approximation of $\mathscr{P}3$ which exploits its structure. Consider (2.5a) and (2.5b) separately by relaxing (2.5c). In fact, (2.5a) may be rewritten in a convex manner. To do this, write L as a weighted product of its incidence matrix, $L = E^\top XE$. Applying Lemma 1 makes the constraint become

$$\begin{bmatrix} (\varepsilon_- + 1)I_{n-1} & JT^\top E^\top XE \\ E^\top XETJ^\top & H_\delta^{-1} \end{bmatrix} \succeq 0. \tag{2.6}$$

As for (2.5b), consider the approximation $LH_\Delta L \approx \left(\frac{\sqrt{H_\Delta}L + L\sqrt{H_\Delta}}{2}\right)^2$. This approximation can be thought of as a rough completion of squares, which lends itself well to our approach of convexifying (2.5b). One should not expect the approximation to be reliably "better" or "worse" than the BMI; rather, it is only intended to reflect the original constraint more than a simple linearization. To this end, substitute this in M_Δ to get

$$\begin{aligned}
&\frac{1}{4}JT^\top(\sqrt{H_\Delta}L + L\sqrt{H_\Delta})^2TJ^\top \succeq (1-\varepsilon_+)I_{n-1} \\
&\frac{1}{2}JT^\top(\sqrt{H_\Delta}L + L\sqrt{H_\Delta})TJ^\top \succeq \sqrt{(1-\varepsilon_+)}I_{n-1} \\
&\frac{1}{2}JT^\top(\sqrt{H_\Delta}L + L\sqrt{H_\Delta})TJ^\top \succeq (1 - \frac{\varepsilon_+}{2} + \frac{\varepsilon_+^2}{8} + O(\varepsilon_+^3))I_{n-1},
\end{aligned}$$

where the second line uses the property that $TJ^\top JT^\top = I_n - \mathbf{1}_n\mathbf{1}_n^\top/n$ is idempotent and that

$$\left(\frac{1}{2}JT^\top(\sqrt{H_\Delta}L + L\sqrt{H_\Delta})TJ^\top\right)^2 \succeq (1-\varepsilon_+)I_{n-1} \succeq 0$$
$$\Leftrightarrow \frac{1}{2}JT^\top(\sqrt{H_\Delta}L + L\sqrt{H_\Delta})TJ^\top \succeq \sqrt{1-\varepsilon_+}I_{n-1} \succeq 0,$$

see [99]. The third line expresses the right-hand side as a Taylor expansion about $\varepsilon_+ = 0$. Neglecting the higher order terms $O(\varepsilon_+^3)$ and applying Lemma 1 gives

$$\begin{bmatrix} \frac{1}{2}JT^\top(\sqrt{H_\Delta}L + L\sqrt{H_\Delta})TJ^\top - (1-\frac{1}{2}\varepsilon_+)I_{n-1} & \frac{1}{\sqrt{8}}\varepsilon_+ I_{n-1} \\ \frac{1}{\sqrt{8}}\varepsilon_+ I_{n-1} & I_{n-1} \end{bmatrix} \succeq 0. \tag{2.7}$$

Returning to $\mathscr{P}3$, note that the latter three constraints are satisfied by $L = E^\top XE$. Then, the approximate reformulation of $\mathscr{P}3$ can be written as

$$\begin{aligned} \mathscr{P}4: \quad \min_{\varepsilon_-,\varepsilon_+,X} \quad & \max(\varepsilon_-,\varepsilon_+) \\ \text{s.t.} \quad & \varepsilon_- \geq 0, \varepsilon_+ \geq 0, \\ & X \succeq 0, (2.6), (2.7). \end{aligned}$$

This is a convex problem in X and solvable in polynomial time. To improve the solution, we perform some post-scaling. Take $L_0^\star = E^\top X_0^\star E$, where $X_0^\star$ is the solution to $\mathscr{P}4$, and let $M_{\Delta 0}^\star = JT^\top L_0^\star H_\Delta L_0^\star TJ^\top, M_{\delta 0}^\star = JT^\top L_0^\star H_\delta L_0^\star TJ^\top$. Then, consider

$$\beta = \sqrt{\frac{2}{\mu_1(M_{\delta 0}^\star) + \mu_{n-1}(M_{\Delta 0}^\star)}},$$

and take $L^\star = \beta L_0^\star$. This shifts the eigenvalues of $M_0^\star(x)$ to $M^\star(x)$ (defined similarly via $L^\star$) such that $1-\mu_1(M_\delta^\star) = -(1-\mu_{n-1}(M_\Delta^\star))$, which shrinks $\max_{i,x}(|1-\mu_i(M^\star(x))|)$. We refer to this metric as $\varepsilon_{L^\star} := \max_{i,x}(|1-\mu_i(M^\star(x))|)$, and it can be verified that this post-scaling satisfies Assumption 4 with regard to $\varepsilon_{L^\star}$. To see this, first consider scaling L by an arbitrarily small

constant, which places the eigenvalues of $I_{n-1} - M(x)$ very close to 1 and satisfies Assumption 4. Then, consider gradually increasing this constant until the lower bound on the minimum eigenvalue and upper bound on the maximum eigenvalue of $I_{n-1} - M(x)$ are equal in magnitude. This is precisely the scaling produced by β. Then, the solution to $\mathscr{P}4$ followed by a post scaling by β given by $L^\star$ is an approximation of the solution to the nonconvex problem $\mathscr{P}3$ with the sparsity structure preserved.

Remark 2. ***(Unknown Local Hessian Bounds).*** *It may be the case that a central entity tasked with computing some $L^\star$ does not have access to the local bounds $\delta_i, \Delta_i, \forall i$. In this case, globally known bounds $\delta \leq \delta_i, \Delta_i \leq \Delta, \forall i$ can be substituted in place of the local values in the formulation of $\mathscr{P}4$. It can be verified that this will result in a more conservative formulation, and that the resulting $L^\star$ will still satisfy Assumption 4 at the expense of possibly larger ε.*

2.3.2 A Bound on Performance

We are motivated to find a "best-case scenario" for our solution given the structural constraints of the network. Instead of solving $\mathscr{P}3$ for L, we solve it for some A where $A_{ij} = 0$ for $j \notin \mathscr{N}_i^2$, i.e. the two-hop neighbor structure of the network and sparsity structure of $LH(x)L$. Define $M_A := JT^\top ATJ^\top$. This problem is:

$$\begin{aligned} \mathscr{P}5: \quad \min_{\varepsilon, A} \quad & \varepsilon \\ \text{s.t.} \quad & -\varepsilon I_{n-1} \preceq I_{n-1} - M_A \preceq \varepsilon I_{n-1}, \\ & A\mathbf{1}_n = \mathbf{0}_n,\ A \succeq 0, \\ & A_{ij} = 0, j \notin \mathscr{N}_i^2 . \end{aligned}$$

This problem is convex in A and produces a solution ε_A, which serves as a lower bound for the solution to $\mathscr{P}3$. It should not be expected that this lower bound is tight or achievable by "reverse engineering" an $L^\star$ with the desired sparsity from the solution $A^\star$ to $\mathscr{P}5$, rather, $\varepsilon_{L^\star} - \varepsilon_A$ gives just a rough indication of how close $\varepsilon_{L^\star}$ is to the conservative lower bound of $\mathscr{P}3$.

2.4 Discrete Time Algorithm for Relaxed Economic Dispatch

In this section, we focus on a *relaxed* version of $\mathscr{P}2$ to develop a direct relation between traditional discrete-time Newton descent and our distributed, approximate method. First, we state the relaxed problem and define the approximate Newton step. We then state the DISCRETE DISTRIBUTED APPROX-NEWTON algorithm and provide a rigorous study of its convergence properties.

2.4.1 Characterization of the Approximate Newton Step

Even the traditional centralized Newton method is not well-suited to solve $\mathscr{P}1$ due to the box constraints (2.1c). For this reason, for now we focus on the relaxed problem

$$\mathscr{P}6: \min_{x} \quad f(x) = \sum_{i=1}^{n} f_i(x_i), \tag{2.8a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_i = d. \tag{2.8b}$$

The equivalent unconstrained problem in z is

$$\mathscr{P}7: \min_{z} \quad g(z) := f(x^0 + Lz) = \sum_{i=1}^{n} f_i(x_i^0 + L_i z). \tag{2.9}$$

Remark 3. ***(Nonuniqueness of Solution).*** *Given a $z^\star$ which solves $\mathscr{P}7$, the set of solutions can be characterized by $\{z^{\star\prime} \mid z^{\star\prime} = z^\star + \gamma \mathbf{1}_n \;\; \gamma \in \mathbb{R}\}$. The fact that $z^{\star\prime}$ is a solution is due to* $\text{null}\,(L) = \text{span}\,(\mathbf{1}_n)$*, and the fact that this characterizes the entire set of solutions is due to* $\text{null}\,(\nabla_{zz} g(z)) = \text{span}\,(\mathbf{1}_n)$.

To solve $\mathscr{P}6$, we aim to implement a descent method in x via the dynamics

$$x^+ = x + \alpha L \tilde{z}_{\text{nt}}, \tag{2.10}$$

where $\tilde{z}_{\text{nt}}$ is the *approximate* Newton step that we seek to compute distributively, and $\alpha > 0$ is a fixed step size.

It is true that $\mathscr{P}7$ is unconstrained with respect to z, although we have already alluded to the fact that the Hessian matrix $\nabla_{zz} g(z) = LH(x+Lz)L$ is rank-deficient stemming from (2.8b). We now reconcile this by deriving a well defined Newton step in a reduced variable $\hat{z} \in \mathbb{R}^{n-1}$. Consider a change of coordinates by the orthogonal matrix T defined in Section 2.2 and write $z = TJ^\top \hat{z}$. Taking the gradient and Hessian of $g(z)$ with respect to $\hat{z}$ gives

$$\nabla_{\hat{z}} g(z) = JT^\top \nabla_z g(z) = JT^\top L \nabla_x f(x + LTJ^\top \hat{z})$$
$$\nabla_{\hat{z}\hat{z}} g(z) = JT^\top LH(x + LTJ^\top \hat{z}) LTJ^\top = M(x + LTJ^\top \hat{z}).$$

Notice that the zero eigenvalue of $\nabla_{zz} g(z)$ is eliminated by this projection and the other eigenvalues are preserved. Evaluating at $x + LTJ^\top \hat{z}\big|_{\hat{z}=0}$, the Newton step in $\hat{z}$ is now well defined as $\hat{z}_{\text{nt}} := -\nabla_{\hat{z}\hat{z}} g(0)^{-1} \nabla_{\hat{z}} g(0) = -M(x)^{-1} JT^\top L \nabla_x f(x)$.

Consider now a q-approximation of $M(x)^{-1}$ given by $\sum_{p=0}^{q} (I_{n-1} - M(x))^p$ and return to the original coordinates to obtain the *approximate* Newton direction $L\tilde{z}_{nt}$:

$$L\tilde{z}_{nt} = -LTJ^\top \sum_{p=0}^{q} (I_{n-1} - M(x))^p JT^\top L \nabla_x f(x).$$

With the property that $LTJ^\top JT^\top L = L^2$, rewrite $L\tilde{z}_{\text{nt}}$:

$$L\tilde{z}_{\text{nt}} = -L \sum_{p=0}^{q} (I_n - LH(x)L)^p L \nabla_x f(x). \tag{2.11}$$

It can be seen via eigendecomposition of $I_n - LHL$, which is normal, and application of Assumption 4 that the terms $L(I_n - LHL)^p$ become small with $p \to \infty$ at a rate dictated by ε. Note that there is a nonconverging mode of the sum corresponding to the eigenspace spanned by $\mathbf{1}_n$, but this is mapped to zero by left multiplication by L. This expression can be computed distributively: each multiplication by L encodes a communication with the neighbor set of

each agent, and we utilize recursion to perform the computation efficiently, which is formally described in Algorithm 1.

2.4.2 The DISTRIBUTED APPROX-NEWTON Algorithm

We now have the tools to introduce the

DISCRETE DISTRIBUTED APPROX-NEWTON algorithm, or DANA-D.

Algorithm 1. DANA-D$_i$

Require: L_{ij} for $j \in \{i\} \cup \mathcal{N}_i$ and communication with nodes $j \in \mathcal{N}_i \cup \mathcal{N}_i^2$

1: **procedure** NEWTON$_i$(x_i^0, L_i, f_i, q)
2: Initialize $x_i \leftarrow x_i^0$
3: **loop**
4: Compute $\dfrac{\partial f_i}{\partial x_i}$, $\dfrac{\partial^2 f_i}{\partial x_i^2}$; send to $j \in \mathcal{N}_i$, $\mathcal{N}_i^2$
5: $y_i \leftarrow L_{ii}\dfrac{\partial f_i}{\partial x_i} + \sum_{j\in\mathcal{N}_i} L_{ij}\dfrac{\partial f_j}{\partial x_j}$
6: $z_i \leftarrow -y_i$
7: $p_i \leftarrow 1$
8: **while** $p_i \leq q$ **do**
9: Acquire y_j from $j \in \mathcal{N}_i^2$
10: $w_i = (I_n - LH(x)L)_i y$
11: $y_i \leftarrow w_i$
12: $z_i \leftarrow z_i - y_i$
13: $p_i \leftarrow p_i + 1$
14: Acquire z_j for $j \in \mathcal{N}_i$
15: $x_i \leftarrow x_i + \alpha\left(L_{ii}z_i + \sum_{j\in\mathcal{N}_i} L_{ij}z_j\right)$
16: **return** x_i

The algorithm is constructed directly from (2.10) and (2.11). The $L\nabla_x f(x^k)$ factor of (2.11) is computed first in the loop starting on line 4. Then, each additional term of the sum is computed recursively in the loop starting on line 8, where y implicitly embeds the exponentiation by p indicated in (2.11), z accumulates each term of the summation of (2.11), w is used as an intermediate variable, and p_i is used as a simple counter. We introduce some abuse of notation by switching to vector and matrix representations of local variables in line 10; this is done for compactness and to avoid undue clutter. Note that the diagonal elements of $H(x)$ are given

by $\partial^2 f_j/\partial x_j^2$ and the matrix and vector operations can be implemented locally for each agent using the corresponding elements y_j, L_{ij}, and L_{ij}^2. The one-hop and two-hop communications of the algorithm are contained in lines 4 and 9, where line 4 calls upon local evaluations of the gradient and Hessian. (In principle, Hessian information could be acquired along with y_j in the first iteration of the inner loop to utilize one fewer two-hop communication, but it need only be acquired once per outer loop.) The information is utilized in local computations indicated the next line in each case. It is understood that agents perform communications and computations synchronously.

The outer loop of the algorithm corresponding to (2.10) is performed starting on line 14. If only one-hop communications are available, each outer loop of the algorithm requires $2q+1$ communications. The process repeats until desired accuracy is achieved. If q is increased, it requires additional communications, but the step approximation gains accuracy.

2.4.3 Convergence Analysis

This section establishes convergence properties of the DANA-D algorithm for problems of the form $\mathscr{P}6$. For the sake of cleaner analysis, we will reframe the algorithm as solving $\mathscr{P}7$ via

$$z^+ = z - \alpha A_q(z)\nabla_z g(z), \tag{2.12}$$

where $A_q(z) := \sum_{p=0}^{q}(I_n - LH(x^0+Lz)L)^p$. Then, note that the solution $z^\star$ to $\mathscr{P}7$ solves $\mathscr{P}6$ by $x^\star = x^0 + Lz^\star$ and that (2.12) is equivalent to (2.10)-(2.11) and Algorithm 1.

Remark 4. ***(Initial Condition, Trajectories, & Solution).*** *Consider an initial condition $z(0) \in \mathbb{R}^n$ with $\mathbf{1}_n^\top z(0) = \omega$. Due to $A_q(z)\nabla_z g(z) \perp \mathbf{1}_n$, the trajectories under* (2.12) *are contained in the set $\{z \mid z = \tilde{z} + (\omega/n)\mathbf{1}_n,\ \tilde{z} \perp \mathbf{1}_n\}$. The solution $x^\star = x^0 + Lz^\star$ to $\mathscr{P}6$ is agnostic to $(\omega/n)\mathbf{1}_n$ due to* $\mathrm{null}\,(L) = \mathrm{span}\,(\mathbf{1}_n)$*, so we consider the solution $z^\star$ uniquely satisfying $\mathbf{1}_n^\top z^\star = \omega$.*

Theorem 1. ***(Convergence of* DANA-D*).*** Given an initial condition $z(0) \in \mathbb{R}^n$, if Assumption 1, on the bidirectional connected graph, Assumption 2, on the feasibility of the initial condition,

Assumption 3, on bounded Hessians, and Assumption 4, on convergent eigenvalues, hold, then the DANA-D dynamics (2.12) converge asymptotically to an optimal solution $z^\star$ of $\mathscr{P}7$ uniquely satisfying $\mathbf{1}_n^\top z^\star = \mathbf{1}_n^\top z(0)$ for any $q \in \mathbb{N}$ and $\alpha < \dfrac{2(1-\varepsilon)}{(n-1)(1+\varepsilon)(1-\varepsilon^{q+1})}$.

Proof. Consider the discrete-time Lyapunov function

$$V(z) = g(z) - g(z^\star)$$

defined on the domain $\mathrm{dom}\,(V) = \{z \mid \mathbf{1}_n^\top z = \mathbf{1}_n^\top z(0)\}$. From the theorem statement and in consideration of Remark 4, the trajectories of z under (2.12) are contained in the domain of V, and $V(z) > 0, \forall z \in \mathrm{dom}\,(V), z \neq z^\star$. To prove convergence to $z^\star$, we must show negativity of

$$V(z^+) - V(z) = g(z^+) - g(z). \tag{2.13}$$

From the weight design of L (Assumption 4), we have $\nabla_{zz} g(z) \preceq (1+\varepsilon) I_n$, $\varepsilon \in [0,1)$. This implies

$$\begin{aligned} g(z^+) &= g(z) + \nabla_z g(z)^\top (z^+ - z) + \frac{1}{2}(z^+ - z)^\top \nabla_{zz} g(z')(z^+ - z) \\ &\leq g(z) + \nabla_z g(z)^\top (z^+ - z) + \frac{1+\varepsilon}{2}\|z^+ - z\|_2^2, \end{aligned}$$

which employs the standard quadratic expansion of convex functions via some z' in the segment extending from z to z^+ (see e.g. §9.1.2 of [17]). Substituting (2.12) gives

$$g(z^+) \leq g(z) - \alpha \nabla_z g(z)^\top A_q(z) \nabla_z g(z) + \frac{(1+\varepsilon)\alpha^2}{2}\|A_q(z)\nabla_z g(z)\|_2^2. \tag{2.14}$$

We now show $A_q(z) \succ 0$ by computing its eigenvalues. Note $\mu_i(I_n - LH(x^0 + Lz)L) \in [-\varepsilon, \varepsilon] \cup \{1\}$. Let $\mu_i(I_n - LH(x^0 + Lz)L) = \eta_i(z)$ for $i \in \{1, \ldots, n-1\}$. The terms of $A_q(z)$ commute and

it is normal, so it can be diagonalized as

$$A_q(z) = W(z) \begin{bmatrix} \ddots & & & \\ & \dfrac{1-\eta_i(z)^{q+1}}{1-\eta_i(z)} & & \\ & & \ddots & \\ & & & q+1 \end{bmatrix} W(z)^\top,$$

where the columns of $W(z)$ are the eigenvectors of $A_q(z) \succ 0$, the last column being $\mathbf{1}_n$, and the terms of the diagonal matrix are its eigenvalues computed by a geometric series.

For now, we only use the fact that $A_q(z) \succ 0$ to justify the existence of $A_q(z)^{1/2}$. Returning to (2.14),

$$g(z^+) \leq g(z) - \alpha \left(\|A_q(z)^{1/2} \nabla_z g(z)\|_2^2 - \frac{(1+\varepsilon)\alpha}{2} \|A_q(z) \nabla_z g(z)\|_2^2 \right). \tag{2.15}$$

Recall $\nabla_z g(z) \perp \mathbf{1}_n$ and that $\mathbf{1}_n$ is an eigenvector of $A_q(z)$ associated with the eigenvalue $q+1$. Consider a matrix $\widetilde{A}_q(z)$ whose rows are projected onto the subspace spanning the orthogonal complement of $\mathbf{1}_n$. More precisely, writing $\widetilde{A}_q(z)$ via its diagonalization gives

$$\widetilde{A}_q(z) = W(z) \begin{bmatrix} \ddots & & & \\ & \dfrac{1-\eta_i(z)^{q+1}}{1-\eta_i(z)} & & \\ & & \ddots & \\ & & & 0 \end{bmatrix} W(z)^\top, \tag{2.16a}$$

$$\widetilde{A}_q(z) \nabla_z g(z) = A_q(z) \nabla_z g(z), \tag{2.16b}$$

$$\widetilde{A}_q(z)^{1/2} \nabla_z g(z) = A_q(z)^{1/2} \nabla_z g(z). \tag{2.16c}$$

Combining (2.15)–(2.16) gives the sufficient condition on α:

$$\alpha < \frac{2\|\widetilde{A}_q(z)^{1/2}\nabla_z g(z)\|_2^2}{(1+\varepsilon)\|\widetilde{A}_q(z)\nabla_z g(z)\|_2^2}. \tag{2.17}$$

Multiply the top and bottom of the righthand side of (2.17) by $\|\widetilde{A}_q(z)^{1/2}\|_2^2$ and apply submultiplicativity of $\|\cdot\|_2^2$:

$$\frac{2}{(1+\varepsilon)\|\widetilde{A}_q(z)^{1/2}\|_2^2} \leq \frac{2\|\widetilde{A}_q(z)^{1/2}\nabla_z g(z)\|_2^2}{(1+\varepsilon)\|\widetilde{A}_q(z)\nabla_z g(z)\|_2^2}. \tag{2.18}$$

Finally, we bound the lefthand side of (2.18) from below by substituting $\eta_i(z)$ with ε:

$$\|\widetilde{A}_q(z)^{1/2}\|_2^2 = \sum_i^{n-1} \frac{1-\eta_i(z)^{q+1}}{1-\eta_i(z)} \leq (n-1)\frac{1-\varepsilon^{q+1}}{1-\varepsilon}, \quad \forall z \in \mathbb{R}^n. \tag{2.19}$$

Combining (2.19) with (2.18) gives the condition on α in the theorem statement and completes the proof. □

In practice, we find this to be a very conservative bound on α due to the employment of many inequalities which simplify the analysis. We note that designing L effectively such that ε is close to zero allows for more flexibility in choosing α large, which intuitively indicates the Taylor approximation of the Hessian inverse converging with greater accuracy in fewer terms q.

Theorem 2. ***(Linear Convergence of* DANA-D*).*** Given an initial condition $z(0) \in \mathbb{R}^n$ and step size $\alpha = \dfrac{(1-\varepsilon)}{(n-1)(1+\varepsilon)(1-\varepsilon^{q+1})}$, if Assumption 1, on the bidirectional connected graph, Assumption 2, on the feasibility of the initial condition, Assumption 3, on bounded Hessians, and Assumption 4, on convergent eigenvalues, hold, the DANA-D dynamics (2.12) converge *linearly* to an optimal solution $z^\star$ of $\mathscr{P}7$ uniquely satisfying $\mathbf{1}_n^\top z^\star = \mathbf{1}_n^\top z(0)$ in the sense that $g(z^+) - g(z) \leq -\dfrac{(1-\varepsilon)^4(1+\varepsilon(-\varepsilon)^q)^2\|z-z^\star\|_2^2}{2(n-1)^2(1+\varepsilon)^3(1-\varepsilon^{2(q+1)})}$ for any $q \in \mathbb{N}$.

Proof. Define

$$c_1(z) = \|\widetilde{A}_q(z)^{1/2}\nabla_z g(z)\|_2^2, \quad c_2(z) = \frac{(1+\varepsilon)}{2}\|\widetilde{A}_q(z)\nabla_z g(z)\|_2^2,$$

with $\widetilde{A}_q(z)$ defined as in (2.16a). Recalling (2.15)–(2.16), consider $\bar{\alpha} = 2\alpha$ as the smallest step size such that $-\bar{\alpha}c_1(z) + \bar{\alpha}^2 c_2(z)$ is not strictly negative for all z, which is obtained from the result of Theorem 1. Then,

$$-\bar{\alpha}c_1(z) + \bar{\alpha}^2 c_2(z) \leq 0 \Rightarrow -\alpha c_1(z) + \alpha^2 c_2(z) \leq -\alpha^2 c_2(z). \tag{2.20}$$

The implication is obtained from the first by substituting $\bar{\alpha} = 2\alpha$. We now consider an implementation of DANA-D with α. From (2.15) and substituting via (2.16b)–(2.16c), we obtain $g(z^+) - g(z) \leq -\alpha c_1(z) + \alpha^2 c_2(z)$. Combining this with the second line of (2.20),

$$g(z^+) - g(z) \leq -\alpha^2 c_2(z). \tag{2.21}$$

We seek a lower bound for $\widetilde{A}_q(z)$. Consider its definition (2.16a), where a lower bound can be obtained by substituting each $\eta_i(z)$ by $-\varepsilon$. Then,

$$\widetilde{A}_q(z) \succeq \frac{1+\varepsilon(-\varepsilon)^q}{1+\varepsilon}\left(I_n - \frac{\mathbf{1}_n\mathbf{1}_n^\top}{n}\right).$$

Returning to (2.21) and applying the definition of $c_2(z)$,

$$g(z^+) - g(z) \leq -\frac{\alpha^2(1+\varepsilon(-\varepsilon)^q)^2}{2(1+\varepsilon)}\|\nabla_z g(z)\|_2^2, \tag{2.22}$$

due to $\text{null}(I_n - \mathbf{1}_n\mathbf{1}_n^\top/n) = \text{span}(\mathbf{1}_n)$ and $\nabla_z g(z) \perp \mathbf{1}_n$.

Next, we bound $\|\nabla_z g(z)\|_2^2$. Apply the Fundamental Theorem of Calculus to compute

$\nabla_z g(z)$ via a line integral. Let $z(s) = sz + (1-s)z^\star$. Then,

$$\nabla_z g(z) = \int_0^1 \nabla_{zz} g(z(s))(z - z^\star) ds. \tag{2.23}$$

Applying Assumption 4 (convergent eigenvalues) gives a lower bound on the Hessian of g, implying a lower bound on its line integral:

$$\begin{gathered} \nabla_{zz} g(z) \succeq (1-\varepsilon)(I - \mathbf{1}_n \mathbf{1}_n^\top / n) \Rightarrow \\ \int_0^1 \nabla_{zz} g(z(s)) ds \succeq (1-\varepsilon)(I - \mathbf{1}_n \mathbf{1}_n^\top / n). \end{gathered} \tag{2.24}$$

Factoring out $z - z^\star$ from (2.23) and applying the second line of (2.24) gives the lower bound

$$\|\nabla_z g(z)\|_2^2 \geq (1-\varepsilon)^2 \|z - z^\star\|_2^2, \tag{2.25}$$

due to $\text{null}(I_n - \mathbf{1}_n \mathbf{1}_n^\top / n) = \text{span}(\mathbf{1}_n)$ and $z - z^\star \perp \mathbf{1}_n$. Combining (2.25) with (2.22) and substituting α:

$$g(z^+) - g(z) \leq -\frac{(1-\varepsilon)^4 (1 + \varepsilon(-\varepsilon)^q)^2 \|z - z^\star\|_2^2}{2(n-1)^2 (1+\varepsilon)^3 (1 - \varepsilon^{2(q+1)})}.$$

□

In principle, this result can be extended to any α which is compliant with Theorem 1; we have chosen this particular α for simplicity. The methods we employ to arrive at the results of Theorems 1 and 2 are necessarily conservative. However, in practice, we find that choosing substantially larger α generally converges to the solution faster. Additionally, we find clear-cut improved convergence properties for larger q (more accurate step approximation) and smaller ε (more effective weight design). Simulations confirm this in Section 2.6.

2.5 Continuous Time Distributed Approximate Newton Algorithm

In this section, we develop a continuous-time Newton-like algorithm to distributively solve $\mathscr{P}2$ for quadratic cost functions. Our method borrows from and expands upon known results of gradient-based saddle-point dynamics [34]. We provide a rigorous proof of convergence and an interpretation of the convergence result for various parameters of the proposed algorithm.

2.5.1 Formulation of Continuous Time Dynamics

First, we adopt a stronger version of Assumption 3:

Assumption 5. ***(Quadratic Cost Functions).*** *The local costs f_i are strongly convex and quadratic, i.e. they take the form*

$$f_i(x_i) = \frac{1}{2}a_i x_i^2 + b_i x_i, \quad i \in \{1,\dots,n\}.$$

Note that the Hessian of f with respect to x is now constant, so we omit the arguments of H and A_q for the remainder of this section. The dynamics we intend to use to solve $\mathscr{P}2$ are substantially more complex than those for the problem with no box constraints, which makes this simplification necessary. In fact, the quadratic model is very commonly used for generator costs in power grid operation [5].

We aim to solve $\mathscr{P}2$ by finding a saddle point of the associated Lagrangian $\mathcal{L}$. Introduce the dual variable $\lambda \in \mathbb{R}^{2n}$ corresponding to (2.2b)–(2.2c), and define $P(z)$ as

$$P(z) = \begin{bmatrix} \underline{P}(z) \\ \overline{P}(z) \end{bmatrix} = \begin{bmatrix} \underline{x} - x^0 - Lz \\ x^0 + Lz - \overline{x} \end{bmatrix} \in \mathbb{R}^{2n}.$$

The Lagrangian of $\mathscr{P}2$ is given by

$$\mathcal{L}(z,\lambda) = g(z) + \lambda^\top P(z). \tag{2.26}$$

We aim to design distributed dynamics which converge to a saddle point $(z^\star,\lambda^\star)$ of (2.26), which solves $\mathscr{P}2$. A saddle point has the property

$$\mathcal{L}(z^\star,\lambda) \le \mathcal{L}(z^\star,\lambda^\star) \le \mathcal{L}(z,\lambda^\star), \quad \forall z \in \mathbb{R}^n, \lambda \in \mathbb{R}^n_{\ge 0}.$$

To solve this, consider Newton-like descent dynamics in the primal variable z and gradient ascent dynamics in the dual variable λ (Newton dynamics are not well defined for linear functions). First, we state some equivalencies:

$$\begin{aligned}
&\nabla_z \mathcal{L}(z,\lambda) = \nabla_z g(z) + \begin{bmatrix} -L & L \end{bmatrix} \lambda, \qquad \nabla_\lambda \mathcal{L}(z,\lambda) = P(z), \\
&\nabla_{zz} \mathcal{L}(z,\lambda) = LHL, \quad \nabla_{\lambda\lambda} \mathcal{L}(z,\lambda) = \mathbf{0}_{2n\times 2n}, \quad \nabla_{\lambda z} \mathcal{L}(z,\lambda) = \nabla_{z\lambda} \mathcal{L}(z,\lambda)^\top = \begin{bmatrix} -L & L \end{bmatrix}.
\end{aligned} \tag{2.27}$$

The CONTINUOUS DISTRIBUTED APPROX-NEWTON, or DANA-C, dynamics are given by

$$\begin{aligned}
\dot{z} &= -A_q \nabla_z \mathcal{L}(z,\lambda), \\
\dot{\lambda} &= \left[\nabla_\lambda \mathcal{L}(z,\lambda)\right]^+_\lambda .
\end{aligned} \tag{2.28}$$

The descent in the primal variable z is the approximate Newton direction as (2.10), augmented with dual ascent dynamics in λ (one-hop communication) and implemented in continuous time. The projection on the dynamics in λ ensures that if $\lambda_i(t_0) \ge 0$ then $\lambda_i(t) \ge 0$ for all $t \ge t_0$.

Define $\mathcal{Z}_q : \mathbb{R}^n \times \mathbb{R}^{2n}_{\ge 0} \to \mathbb{R}^n \times \mathbb{R}^{2n}$ as the map in (2.28) implemented by DANA-C. We now make the following assumptions on initial conditions and the feasibility set.

Assumption 6. ***(Initial Dual Feasibility).*** *The initial condition $\lambda(0)$ is dual feasible, i.e. $\lambda(0) \succeq 0$.*

Assumption 7. ***(Nontrivial Primal Feasibility).*** *The feasibility set of $\mathscr{P}2$ is such that $\exists z$ with $P(z) \prec 0$.*

The dynamics $\mathcal{Z}_q$ are not well suited to handle λ infeasible, so Assumption 6 is necessary. As for Assumption 7, if it does not hold, then either $d = \sum \underline{x}$ or $d = \sum \bar{x}$ or $\mathscr{P}1$ is infeasible, which are trivial cases. Assuming it does hold, Slater's condition is satisfied and KKT conditions are necessary and sufficient for solving $\mathscr{P}2$.

Due to the structure of L, $\dot{z}$ is computed using only $(2q+1)$-hop neighbor information. In practice, the quantity $A_q \nabla_z \mathcal{L}(z, \lambda)$ may be computed recursively over multiple one-hop or two-hop rounds of communication, with a discrete step taken in the direction indicated by $(\dot{z}, \dot{\lambda})$. Note that a table statement of this discretized algorithm would be quite similar to Algorithm 1 (with the addition of one-hop dynamics in λ), so we omit it here for brevity. Discrete-time algorithms to solve this problem do exist, see e.g. [83] in which the authors achieve convergence to a ball around the optimizer whose radius is a function of the step size. However, the analysis of discrete-time algorithms to solve $\mathscr{P}2$ via a Newton-like method is outside the scope of this work.

2.5.2 Convergence Analysis

This section provides a rigorous proof of convergence of the distributed dynamics $\mathcal{Z}_q$ to the optimizer $(z^\star, \lambda^\star)$ of $\mathscr{P}2$. The solution $x^\star$ to $\mathscr{P}1$ may then be computed via a one-hop neighbor communication by $x^\star = x^0 + Lz^\star$.

Theorem 3. ***(Convergence of Continuous Dynamics $\mathcal{Z}_q$).*** If Assumption 1, on the undirected and connected graph, Assumption 2, on the feasible initial condition, Assumption 4, on convergent eigenvalues, Assumption 5, on quadratic cost functions, Assumption 6, on the feasible dual initial condition, and Assumption 7, on nontrivial primal feasibility, hold, then the solution trajectories under $\mathcal{Z}_q$ assymptotically converge to an optimal point $(z^\star, \lambda^\star)$ of $\mathscr{P}2$, where $z^\star$ uniquely satisfies $\mathbf{1}_n^\top z^\star = \mathbf{1}_n^\top z(0)$.

Proof. Consider $Q = \begin{bmatrix} A_q^{-1} & 0 \\ 0 & I_{2n} \end{bmatrix} \succ 0$ and define the Lyapunov function

$$V_Q(z,\lambda) := \frac{1}{2}\begin{bmatrix} z-z^\star \\ \lambda-\lambda^\star \end{bmatrix}^\top Q \begin{bmatrix} z-z^\star \\ \lambda-\lambda^\star \end{bmatrix} = \frac{1}{2}\Big(\|A_q^{-1/2}(z-z^\star)\|_2^2 + \|(\lambda-\lambda^\star)\|_2^2\Big). \tag{2.29}$$

The time derivative of V_Q along the trajectories of $\mathcal{Z}_q$ is

$$\begin{aligned}
\dot{V}_Q(z,\lambda) = \begin{bmatrix} z-z^\star \\ \lambda-\lambda^\star \end{bmatrix}^\top Q \begin{bmatrix} \dot{z} \\ \dot{\lambda} \end{bmatrix} &= -(z-z^\star)^\top A_q^{-1} A_q \nabla_z \mathcal{L}(z,\lambda) + (\lambda-\lambda^\star)^\top [\nabla_\lambda \mathcal{L}(z,\lambda)]_\lambda^+ \\
&\overset{(a)}{\leq} -(z-z^\star)^\top \nabla_z \mathcal{L}(z,\lambda) + (\lambda-\lambda^\star)^\top \nabla_\lambda \mathcal{L}(z,\lambda) \\
&\overset{(b)}{=} -(z-z^\star)^\top LHL(z-z^\star) - (z-z^\star)^\top \begin{bmatrix} -L & L \end{bmatrix} (\lambda-\lambda^\star) \\
&\quad + (\lambda-\lambda^\star)^\top \begin{bmatrix} -L & L \end{bmatrix}^\top (z-z^\star) = -\|H^{1/2}L(z-z^\star)\|_2^2 \overset{(c)}{<} 0, \quad z \neq z^\star.
\end{aligned} \tag{2.30}$$

The inequality (a) follows from the componentwise relation $(\lambda_i - \lambda_i^\star)(\big[\nabla_{\lambda_i}\mathcal{L}\big]_{\lambda_i}^+ - \nabla_{\lambda_i}\mathcal{L}) \leq 0$. To see this, if $\lambda_i > 0$, the projection is inactive and this term equals zero. If $\lambda_i = 0$, then the inequality follows from $\lambda_i^\star \geq 0$ and $\big[\nabla_{\lambda_i}\mathcal{L}\big]_{\lambda_i}^+ - \nabla_{\lambda_i}\mathcal{L} \geq 0$. The equality (b) is obtained from an application of the Fundamental Theorem of Calculus and computing the line integral along the line $(z(s),\lambda(s)) = s(z,\lambda) + (1-s)(z^\star,\lambda^\star)$ as follows:

$$\begin{aligned}
\nabla_z \mathcal{L}(z,\lambda) &= \int_0^1 \Big(\nabla_{zz}\mathcal{L}(z(s),\lambda(s))(z-z^\star) + \nabla_{\lambda z}\mathcal{L}(z(s),\lambda(s))(\lambda-\lambda^\star)\Big)ds \\
&= \nabla_{zz}\mathcal{L}(z,\lambda)(z-z^\star) + \nabla_{\lambda z}\mathcal{L}(z,\lambda)(\lambda-\lambda^\star), \\
\nabla_\lambda \mathcal{L}(z,\lambda) &= \int_0^1 \Big(\nabla_{\lambda\lambda}\mathcal{L}(z(s),\lambda(s)(\lambda-\lambda^\star) + \nabla_{z\lambda}\mathcal{L}(z(s),\lambda(s))(z-z^\star)\Big)ds \\
&= \nabla_{z\lambda}\mathcal{L}(z,\lambda)(z-z^\star),
\end{aligned}$$

where the integrals can be simplified due to $\nabla_{zz}\mathcal{L}$ and $\nabla_{\lambda z}\mathcal{L}$ constant, as per (2.27). Recalling

Remark 4, which applies similarly here, and noticing $\dot{z} \perp \mathbf{1}_n$, it follows from the theorem statement that $(z - z^\star) \perp \mathbf{1}_n$. Additionally, zero is a simple eigenvalue of $H^{1/2}L$ with a corresponding right eigenvector $\mathbf{1}_n$, implying that (c), the last line of (2.30), is strict for $z \neq z^\star$.

Let $\mathscr{S} := \left\{ (z, \lambda) \mid z = z^\star, \lambda \succeq 0 \right\}$ be an asymptotically stable set under the dynamics $\mathcal{Z}_q$ defined in (2.28). We aim to show the largest invariant set contained in $\mathscr{S}$ is the optimizer $\{(z^\star, \lambda^\star)\}$, so we reason with KKT conditions to complete the convergence argument for λ. For $(z, \lambda) \in \mathscr{S}$, clearly primal feasibility is satisfied. Assumption 6 gives feasibility of $\lambda(0)$, which is maintained along the trajectories of $\mathcal{Z}_q$. The stationarity condition $\nabla_z \mathcal{L}(z^\star, \lambda^\star) = 0$ is also satisfied for $(z, \lambda) \in \mathscr{S}$: examine the dynamics $\dot{z}(t) = -A_q \nabla_z \mathcal{L}(z, \lambda) \equiv 0$. It follows that $\nabla_z \mathcal{L}(z, \lambda)_{(z,\lambda) \in \mathscr{S}} = 0$ due to A_q being full rank. Then, each KKT condition has been satisfied for $(z, \lambda) \in \mathscr{S}$ except complementary slackness: $P_i(z)\lambda_i = 0$ for $i \in \{1, \ldots, 2n\}$. We now address this.

Notice the relation $\dot{z} \equiv 0$ implies

$$\lambda(t) = \hat{\lambda} + \phi_{\underline{\lambda}}(t) \begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_n \end{bmatrix} + \phi_{\overline{\lambda}}(t) \begin{bmatrix} \mathbf{0}_n \\ \mathbf{1}_n \end{bmatrix} \tag{2.31}$$

for some constant $\hat{\lambda} \in \mathbb{R}^{2n}$ and possibly time varying $\phi_{\underline{\lambda}}(t), \phi_{\overline{\lambda}}(t) \in \mathbb{R}$. This is due to $\operatorname{null} L = \operatorname{span}\{\mathbf{1}_n\}$ and inferring from $\dot{z} \equiv 0$ that $\begin{bmatrix} -L & L \end{bmatrix} \lambda(t)$ must be constant. Additionally, we may infer from the map $\mathcal{Z}_q$ that $\phi_{\underline{\lambda}}(t), \phi_{\overline{\lambda}}(t)$ are continuous and piecewise smooth. The dynamics $\dot{\lambda}$ and differentiating (2.31) in time gives

$$\dot{\lambda} = [\nabla_\lambda \mathcal{L}(z^\star, \lambda)]^+_\lambda = [P(z^\star)]^+_\lambda \in \partial\phi_{\underline{\lambda}}(t) \begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_n \end{bmatrix} + \partial\phi_{\overline{\lambda}}(t) \begin{bmatrix} \mathbf{0}_n \\ \mathbf{1}_n \end{bmatrix}, \tag{2.32}$$

where $\partial\phi_{\underline{\lambda}}(t)$ and $\partial\phi_{\overline{\lambda}}(t)$ are subdifferentials with respect to time of $\phi_{\underline{\lambda}}(t)$ and $\phi_{\overline{\lambda}}(t)$, respectively. Then, $\phi_{\underline{\lambda}}(t)$ and $\phi_{\overline{\lambda}}(t)$ are additionally piecewise linear due to $P(z^\star)$ constant. We now state two cases for $\underline{P}(z^\star)$ to prove $\underline{\lambda}(t) \to \underline{\lambda}^\star$.

Case 1: $\underline{P}_i(z^\star) = 0$ for at least one $i \in \{1,\dots,n\}$. Then, $\dot{\underline{\lambda}}_i = 0$ and from (2.32) this implies $\dot{\underline{\lambda}} = \mathbf{0}_n$. Reasoning from the projection dynamics, this implies either $\underline{\lambda}_j = 0$ or $\underline{P}_j(z^\star) = 0$ for each j, which satisfies the complementary slackness condition $\underline{\lambda}_j^\star \underline{P}_j(z^\star) = 0$ for every $j \in \{1,\dots,n\}$, and we conclude that $\underline{\lambda} = \underline{\lambda}^\star$ for $(z,\lambda) \in \mathscr{S}$.

Case 2: $\underline{P}(z^\star) \prec 0$. Complementary slackness states $\underline{\lambda}_i^\star \underline{P}_i(z^\star) = 0$ for each $i \in \{1,\dots,n\}$, implying $\underline{\lambda}^\star = \mathbf{0}_n$. The dynamics preserve $\lambda(t) \succeq 0$, so the quantity $\underline{\lambda}_i - \underline{\lambda}_i^\star$ is strictly positive for any $\underline{\lambda}_i \neq \underline{\lambda}_i^\star$. Applying this to the term $(\lambda - \lambda^\star)^\top [\nabla_\lambda \mathcal{L}(z,\lambda)]_\lambda^+$ obtained from the second equality (third line) of (2.30), and also applying $\underline{P}(z^\star) = \nabla_\lambda \mathcal{L}(z^\star,\lambda) \prec 0$, we obtain $\dot{V}_Q < 0$ for $z = z^\star, \underline{\lambda} \neq \underline{\lambda}^\star$.

The inferences of Case 1 (satisfying complementary slackness) and Case 2 (reasoning with $\dot{V}_Q$) hold similarly for $\overline{\lambda}$. Then, we have shown that $\dot{V}_Q(z,\lambda) < 0, \forall (z,\lambda) \in \mathscr{S} \setminus \{(z^\star,\lambda^\star)\}$. Asymptotic convergence to the primal and dual optimizers of $\mathscr{P}2$ follows from the LaSalle Invariance Principle [62]. □

2.5.3 Interpretation of the Convergence Result

For fast convergence, it is desirable for the ratio $\dot{V}_Q / V_Q < 0$ to be large in magnitude for any $(z,\lambda) \in \mathbb{R}^n \times \mathbb{R}_+^{2n}$. Recall the diagonalization of A_q and use this to compute A_q^{-1}:

$$A_q = W \begin{bmatrix} \dfrac{1-\eta_1^{q+1}}{1-\eta_1} & & & \\ & \ddots & & \\ & & \dfrac{1-\eta_{n-1}^{q+1}}{1-\eta_{n-1}} & \\ & & & q+1 \end{bmatrix} W^\top,$$

$$A_q^{-1} = W \begin{bmatrix} \frac{1-\eta_1}{1-\eta_1^{q+1}} & & & \\ & \ddots & & \\ & & \frac{1-\eta_{n-1}}{1-\eta_{n-1}^{q+1}} & \\ & & & (q+1)^{-1} \end{bmatrix} W^\top.$$

Next, write $z - z^\star = \zeta_1 w_1 + \cdots + \zeta_{n-1} w_{n-1}$ as a weighted sum of the eigenvectors w_i of $I_n - LHL$. Note that we do not need $w_n = \mathbf{1}_n$ for this representation due to $z - z^\star \perp w_n$. Then, $V_Q = \sum_{i=1}^{n-1} \zeta_i^2 (1-\eta_i)/(1-\eta_i^{q+1}) + V_\lambda$, where $V_\lambda := ||\lambda - \lambda^\star||_2^2$. Additionally, note that LHL and A_q^{-1} share eigenvectors, so $\dot{V}_Q \leq -\sum_{i=1}^{n-1} \zeta_i^2 (1-\eta_i)$. Toward this end, we can write

$$\frac{\dot{V}_Q}{V_Q} \leq \frac{-\sum_{i=1}^{n-1} \zeta_i^2 (1-\eta_i)}{\sum_{i=1}^{n-1} \zeta_i^2 \left(\frac{1-\eta_i}{1-\eta_i^{q+1}} \right) + V_\lambda}.$$

To interpret this, first reason with the values of q. Consider $q = 0$, which is analogous to a gradient-based method. Then, the rational in the sum contained in the denominator is equal to one and there is no *weighting*, in a sense, to the step direction. In other words, if the value of ζ_i happens to be large in magnitude corresponding to the eigenvector w_i of $\nabla_{zz}\mathcal{L}$ whose corresponding eigenvalue $(1-\eta_i)$ is small in magnitude, then that term does not appropriately dominate the numerator relative to each other term and the quantity $\dot{V}_Q/V_Q$ is small in magnitude. On the other hand, if q is large, then the quantity $1-\eta_i^{q+1}$ is close to 1, and the terms of the sums in the numerator and denominator have the effect of "cancelling" one another, which provides more uniform convergence on the trajectories of z. In addition, if the values of η_i are small in magnitude, i.e. our weight design on L was relatively successful, the quantity $1-\eta_i^{q+1}$ approaches 1 more quickly and the effect of a particular ζ_i being large relative to the other terms in the sum is diminished for any particular q.

Note that, although we have framed this argument as an improvement over the gradient technique, it may be the case that for a particular time t the decomposition on $z(t)$ may have a

large ζ_i corresponding to $1-\eta_i$ large. This actually provides superior momentary convergence compared to a Newton-like method. However, we contend that the oscillatory nature of the trajectories over the entire time horizon gives way to improved convergence from the Newton flavor of our algorithm. This is confirmed in simulation.

Finally, it is apparent that choosing q even is (generally speaking) superior to q odd: the quantity $1-\eta_i^{q+1}$ may take values in $\left[1-\varepsilon^{q+1}, 1+\varepsilon^{q+1}\right]$, as opposed to odd q for which $1-\eta_i^{q+1}$ takes values in $\left[1-\varepsilon^{q+1}, 1\right]$. We would like this quantity to be large so the magnitude of $\dot{V}_Q/V_Q$ is large. This observation of choosing even q to prompt superior convergence is confirmed in simulation.

This discussion neglects the V_λ term which may be large for arbitrarily "bad" initial conditions $\lambda(0) \succeq 0$. However, the ascent direction in λ is clearly more effective for z nearly optimal, so this term is "cooperative" in the sense that its decay roughly corresponds to the decay of the Lyapunov term in z.

To summarize, gradient methods neglect the curvature of the underlying cost function, which dictates the convergence properties of descent algorithms. By weighting the descent direction by A_q, we elegantly capture this curvature in a distributed fashion and the solution trajectory reflects this property. We now provide a remark on convergence of the algorithm for nonquadratic costs that are well approximated by quadratic functions.

Remark 5. ***(Convergence of* DANA-C *for Approximately Quadratic Costs).*** *Instead of Assumption 5 (quadratic costs), let Assumption 3 (general costs) hold and consider the dynamics*

$$\begin{aligned} \dot{z} &= -A_q(z)\nabla_z \mathcal{L}(z,\lambda), \\ \dot{\lambda} &= \left[\nabla_\lambda \mathcal{L}(z,\lambda)\right]_\lambda^+ . \end{aligned} \tag{2.33}$$

Let $H' := \dfrac{H_\Delta + H_\delta}{2}$ *and* $A_q' := \sum_{p=0}^{q}(I_n - LH'L)^p$. *In a sense, these matrices are obtained from quadratic approximations of the nonquadratic costs* f_i, *i.e.* $\left|\dfrac{\partial^2 f_i}{\partial x_i^2} - H_{ii}'\right| \leq \dfrac{\Delta_i - \delta_i}{2}$. *Use*

$Q = \begin{bmatrix} A_q'^{-1} & 0 \\ 0 & I_{2n} \end{bmatrix}$ *to define the quadratic Lyapunov function* $V_Q(z,\lambda)$ *as in* (2.29)*. Differentiating along the trajectories of* (2.33) *now gives*

$$\dot{V}_Q(z,\lambda) = \dot{V}_Q'(z,\lambda) + U(e,z,\lambda),$$

where e gives some measure of how much the functions deviate from quadratic and $U(0,z,\lambda) = 0$*. The* $\dot{V}_Q'(z,\lambda)$ *is obtained by decomposing the dynamics* (2.33) *as*

$$\begin{aligned} \dot{z} &= -A_q' \nabla_z \mathcal{L}(z,\lambda) + u(e,z,\lambda), \\ \dot{\lambda} &= [\nabla_\lambda \mathcal{L}(z,\lambda)]_\lambda^+ . \end{aligned}$$

and including only the terms without $u(e,z,\lambda)$*, where the remaining terms are captured by* $U(e,z,\lambda)$*. U and u are continuous functions of e, and* $u(0,z,\lambda) = 0$*. Applying the convergence argument of Theorem 3 to* $V_Q'(z,\lambda)$*, the continuity of U and u imply* $\dot{V}_Q'(z,\lambda) < -U(\bar{e},z,\lambda)$ *for sufficiently small* $\bar{e}$*. Therefore,* $\dot{V}_Q(z,\lambda) < 0$ *for functions that are well approximated by quadratic functions.*

2.6 Simulations and Discussion

In this section, we implement our weight design and verify the convergence of the DISTRIBUTED APPROX-NEWTON algorithm in each of the discrete-time (relaxed) and continuous time (box-constrained) settings.

2.6.1 Weight Design

To evaluate the weight design posed in Section 2.3 we use quadratic costs in accordance with Assumption 5, i.e. $\delta_i = \Delta_i = a_i, \forall i$. We do this in order to isolate the other parameters for this part of the study. Consider the following metrics: the solution to $\mathscr{P}4$ followed by the

post-scaling by β gives $\varepsilon_{L^\star} := \max(|1-\mu_i(M^\star)|)$; this metric represents the convergence speed of DISTRIBUTED APPROX-NEWTON when applying our proposed weight design of L. Using the same topology $(\mathscr{N},\mathscr{E})$, the solution to $\mathscr{P}5$ gives the metric ε_A. Note that ε_A is a *best-case* estimate of the weight design problem; however, "reverse engineering" an $L^\star$ from the solution $A^\star$ to $\mathscr{P}5$ is both intractable and generally likely to be infeasible. With this in mind, the metric ε_A is a very conservative lower bound, whereas $\varepsilon_{L^\star}$ is the metric for which we can compute a feasible $L^\star$. The objective of each problem is to minimize the associated ε; to this end, we aim to characterize the relationship between network parameters and these metrics. We ran 100 trials on each of 16 test cases which encapsulate a variety of parameter cases: two cases for the cost coefficients, a *tight* distribution $a_i \in \mathscr{U}[0.8,1.2]$ and a *wide* distribution $a_i \in \mathscr{U}[0.2,5]$. For topologies, we randomly generated connected graphs with network size $n \in \{10,20,30,40,50\}$, a *linearly* scaled number of edges $|\mathscr{E}| = 3n$, and a *quadratically* scaled number of edges $|\mathscr{E}| = 0.16n^2$ for $n \in \{30,40,50\}$. The linearly scaled connectivity case corresponds to keeping the average degree of a node constant for increasing network sizes, while the quadratically scaled case roughly preserves the proportion of connected edges to total possible edges, which is a quadratic function of n and equal to $n(n-1)/2$ for an undirected network. The results are depicted in Table 2.1, where the quadratically scaled cases are indicated by boldface. This gives the mean Σ and standard deviation σ of the distributions for *performance* $\varepsilon_{L^\star}$ and *performance gap* $\varepsilon_{L^\star} - \varepsilon_A$.

From these results, first note that the *tightly* distributed coefficients a_i result in improved $\varepsilon_{L^\star}$ across the board compared to the *widely* distributed coefficients. We attribute this to the approximation $LHL \approx \left(\frac{\sqrt{H}L + L\sqrt{H}}{2}\right)^2$ being more accurate for roughly homogeneous $H = \mathrm{diag}(a_i)$. Next, it is clear that in the cases with *linearly* scaled edges, $\varepsilon_{L^\star}$ worsens as network size increases. This is intuitive: the *proportion* of connected edges in the graph decreases as network size increases in these cases. This also manifests itself in the performance gap $\varepsilon_{L^\star} - \varepsilon_A$ shrinking, indicating the *best-case* solution ε_A (for which a valid L does not necessarily exist) degrades even

Table 2.1. Laplacian Design. Quadratically-scaled number-of-edge cases are indicated by boldface.

$a_i \in \mathscr{U}[0.8, 1.2]$ $b_i \in \mathscr{U}[0, 1]$	$\Sigma(\varepsilon_{L^\star})$	$\sigma(\varepsilon_{L^\star})$	$\Sigma(\varepsilon_{L^\star} - \varepsilon_A)$	$\sigma(\varepsilon_{L^\star} - \varepsilon_A)$
$n = 10$ $\lvert\mathscr{E}\rvert = 30$	0.6343	0.0599	0.2767	0.0186
$n = 20$ $\lvert\mathscr{E}\rvert = 60$	0.8655	0.0383	0.2879	0.0217
$n = 30$ $\lvert\mathscr{E}\rvert = 90$	0.9100	0.0250	0.2666	0.0233
$n = 40$ $\lvert\mathscr{E}\rvert = 120$	0.9303	0.0201	0.2501	0.0264
$n = 50$ $\lvert\mathscr{E}\rvert = 150$	0.9422	0.0175	0.2375	0.0264
$n = 30$ $\lvert\mathscr{E}\rvert = \mathbf{144}$	0.7266	0.0324	0.2973	0.0070
$n = 40$ $\lvert\mathscr{E}\rvert = \mathbf{256}$	0.6528	0.0366	0.2829	0.0091
$n = 50$ $\lvert\mathscr{E}\rvert = \mathbf{400}$	0.5840	0.0281	0.2641	0.0101
$a_i \in \mathscr{U}[0.2, 5]$ $b_i \in \mathscr{U}[0, 1]$	$\Sigma(\varepsilon_{L^\star})$	$\sigma(\varepsilon_{L^\star})$	$\Sigma(\varepsilon_{L^\star} - \varepsilon_A)$	$\sigma(\varepsilon_{L^\star} - \varepsilon_A)$
$n = 10$ $\lvert\mathscr{E}\rvert = 30$	0.6885	0.0831	0.3288	0.0769
$n = 20$ $\lvert\mathscr{E}\rvert = 60$	0.8965	0.0410	0.3241	0.0437
$n = 30$ $\lvert\mathscr{E}\rvert = 90$	0.9389	0.0254	0.2878	0.0395
$n = 40$ $\lvert\mathscr{E}\rvert = 120$	0.9539	0.0189	0.2830	0.0355
$n = 50$ $\lvert\mathscr{E}\rvert = 150$	0.9628	0.0168	0.2590	0.0335
$n = 30$ $\lvert\mathscr{E}\rvert = \mathbf{144}$	0.7997	0.0520	0.3587	0.0524
$n = 40$ $\lvert\mathscr{E}\rvert = \mathbf{256}$	0.7339	0.0550	0.3688	0.0569
$n = 50$ $\lvert\mathscr{E}\rvert = \mathbf{400}$	0.6741	0.0487	0.3543	0.0425

quicker as a function of network size than our solution $\varepsilon_{L^\star}$. On the other hand, $\varepsilon_{L^\star}$ substantially improves as network size increases in the *quadratically* scaled cases, with a roughly constant performance gap $\varepsilon_{L^\star} - \varepsilon_A$. Considering this relationship between the linear and quadratic scalings on $|\mathscr{E}|$ and the metrics $\varepsilon_{L^\star}$ and ε_A, we get the impression that both proportion of connectedness and average node degree play a role in both the effectiveness of our weight-designed solution $L^\star$ and the best-case solution. For this reason, we postulate that $\varepsilon_{L^\star}$ remains roughly constant in large-scale applications if the number of edges is scaled subquadratically as a function of network size; equivalently, the convergence properties of DISTRIBUTED APPROX-NEWTON algorithm remain relatively unchanged when using our proposed weight design and growing the number of communications per agent sublinearly as a function of n.

2.6.2 Discrete-Time Distributed Approx-Newton

Consider solving $\mathscr{P}6$ with DANA-D for a network of $n = 100$ generators and $|\mathscr{E}| = 250$ communication links. The local computations required of each generator are simple vector operations whose dimension scales linearly with the network size, which can be implemented on a microprocessor. The graph topology is plotted in Figure 2.1. The problem parameters are given by

$$f_i(x_i) = \frac{1}{2}a_i x_i^2 + b_i x_i + c_i \sin(x_i + \theta_i), \qquad a_i \in \mathscr{U}[2,4], \quad b_i \in \mathscr{U}[-1,1],$$
$$c_i \in \mathscr{U}[0,1], \quad \theta_i \in \mathscr{U}[0,2\pi], \qquad d = 200, \quad x^0 = (d/n)\,\mathbf{1}_n.$$

Note that $0 < a_i - c_i \leq \dfrac{\partial^2 f_i}{\partial x_i^2} \leq a_i + c_i$ satisfies Assumption 3. We compare to the DGD and weight design policies for resource allocation described in [107], along with an "unweighted" version of [107] in the sense that L is taken to be the degree matrix minus the adjacency matrix of the graph, followed by the post-scaling described in Section 2.3.1 to guarantee convergence. The results are given in Figure 2.2, which show linear convergence to the optimal value as the number of iterations increases, with fewer iterations needed for larger q. We note a substantially

improved convergence over the DGD methods, even for the $q = 0$ case which utilizes an equal number of agent-to-agent communications as DGD. This can be attributed in-part to the superior weight design of our method, which is cognizant of second-order information.

In addition, in Figure 2.2 we plot convergence of DGD, weighted by the one-sided design scheme in [107], compared to our two-sided design with $q = 0$, for cases in which only a universal bound on δ_i, Δ_i is known (namely, using $\underline{\delta} \leq \delta_i, \Delta_i \leq \overline{\Delta}, \forall i$, as in Remark 2). We note an improved convergence in each case for the locally known bounds versus the universal bound, while the locally weighted DGD method outperforms our $q = 0$ two-sided globally weighted method by a slight margin.

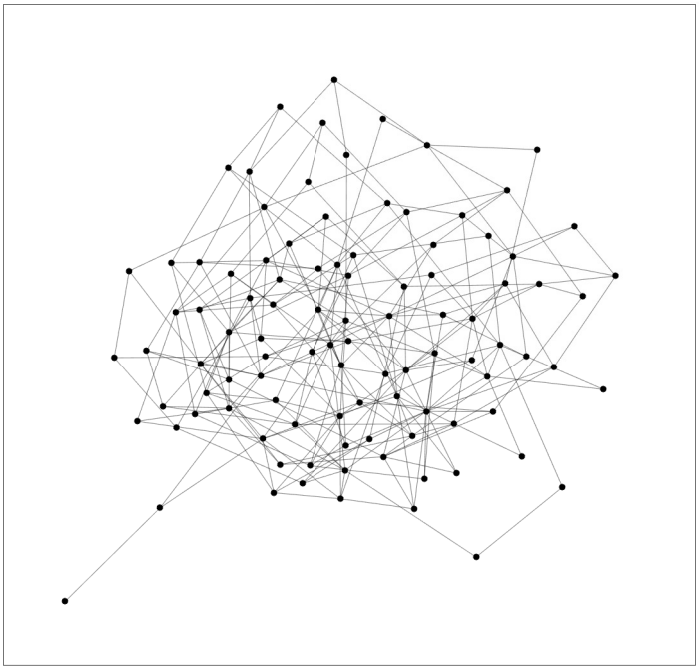


Figure 2.1. Communication topology used for discrete-time numerical study; $n = 100, |\mathscr{E}| = 250$.

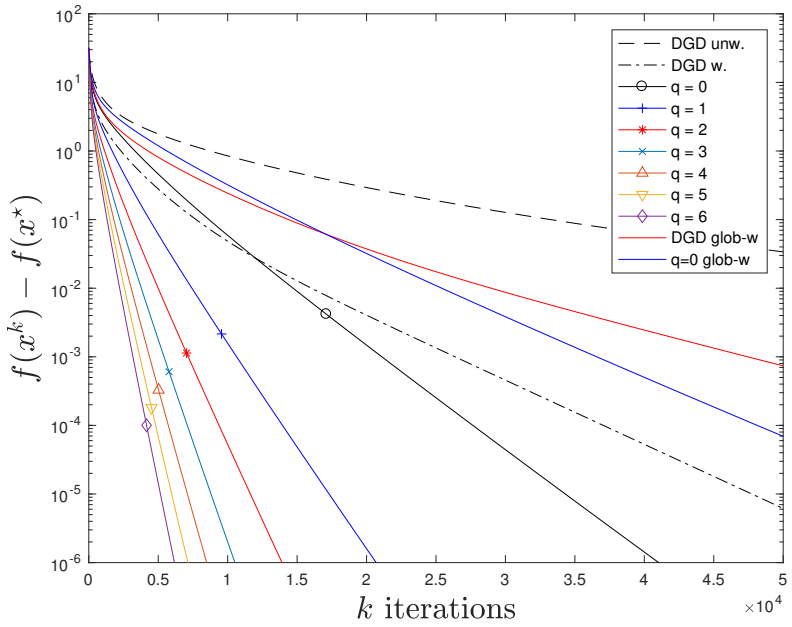


Figure 2.2. Comparison of weighted and unweighted DGD versus DANA-D with various q for solving $\mathscr{P}6$; $n = 100, |\mathscr{E}| = 250$.

2.6.3 Continuous-Time Distributed Approx-Newton

We now study DANA-C for solving $\mathscr{P}1$ for a simple 3 node network with two edges $\mathscr{E} = \{\{1,2\},\{2,3\}\}$ for the sake of visualizing trajectories. The problem parameters are given by

$$f_1(x_1) = \frac{1}{4}x_1^2 + \frac{1}{2}x_1, \quad f_2(x_2) = \frac{3}{4}x_2^2 + \frac{1}{2}x_2, \quad f_3(x_3) = 2x_3^2 + \frac{1}{2}x_3,$$
$$\underline{x} = \begin{bmatrix} 0.2 & 2.5 & 1.5 \end{bmatrix}^\top, \quad \overline{x} = \begin{bmatrix} 1 & 6 & 4 \end{bmatrix}^\top, \quad d = 6,$$
$$x^0 = \begin{bmatrix} 5 & -1 & 2 \end{bmatrix}^\top, \quad z(0) = \mathbf{0}_3, \quad \underline{\lambda}(0) = \begin{bmatrix} 1.5 & .5 & 0 \end{bmatrix}, \quad \overline{\lambda}(0) = \begin{bmatrix} 0 & 2 & 1 \end{bmatrix}$$

Note that x^0 is infeasible with respect to $\underline{x}, \overline{x}$; all that we require is it satisfies Assumption 2 (feasible with respect to d). We plot the trajectories of the 3-dimensional state projected onto the plane orthogonal to $\mathbf{1}_3$ under various q. Figure 2.3 shows this, with a zoomed look at the optimizer in Figure 2.4.

It is clear that choosing q even versus q odd has a qualitative effect on the shape of

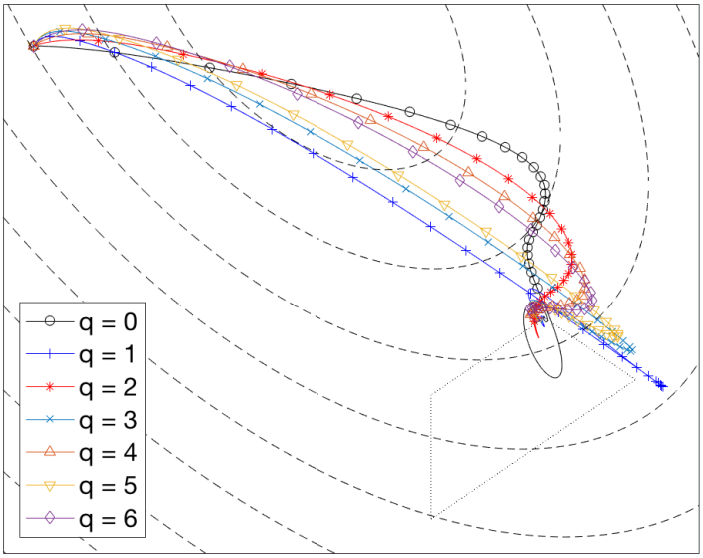


Figure 2.3. Three node case: projection of $x^0 + Lz(t) \in \mathbb{R}^3$ onto the 2-dimensional plane $\{x \mid \sum_i x_i = d\}$. Markers plotted for $t = 0, 0.2, 0.4, \ldots, 5$ seconds. Dashed line ellipses indicate intersection of ellipsoid level sets with the plane; dotted lines indicate intersection of box constraints with the plane.

the trajectories, as noted in Section 2.5.3. Looking at Figure 2.3, it seems the trajectories are intially pulled toward the unconstrained optimizer (center of the level sets) with some bias due to $\lambda(0) \neq \mathbf{0}_6$. As λ is given time to evolve, these trajectories are pulled back toward satisfying the box constraints indicated by the dotted quadrilateral, i.e. the intersection of the box constraints and the plane defined by $\{x \mid \sum_i x_i = d\}$.

For a quantitative comparison, we consider $n = 40$ generators with $|\mathscr{E}| = 156$ communication links whose graph is given by Figure 2.5 and the following parameters.

$$f_i(x_i) = \frac{1}{2} a_i x_i^2 + b_i x_i,\ a_i \in \mathscr{U}[0.5, 3],\ b_i \in \mathscr{U}[-2, 2],$$

$$\underline{x}_i \in \mathscr{U}[1.5, 3], \quad \bar{x}_i \in \mathscr{U}[3, 4.5], \quad i \in \{1, \ldots, 100\},$$

$$d = 120,\ x^0 = 3 * \mathbf{1}_{40},\ z(0) = \mathbf{0}_{40},\ \lambda(0) = \mathbf{0}_{80}.$$

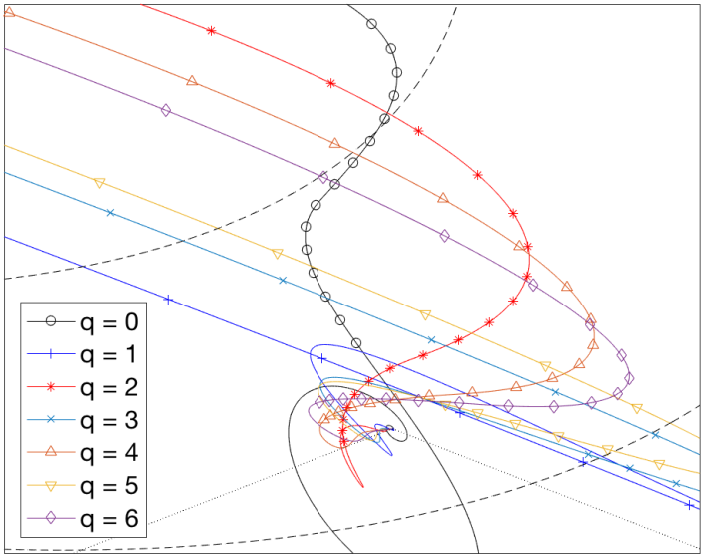


Figure 2.4. Three node case: trajectories zoomed closer to the optimizer. Markers plotted in $0.2s$ increments up to $t = 5s$.

Note from Figure 2.6 that convergence with respect to $\|x^0 + Lz(t) - x^\star\| + \|\lambda(t) - \lambda^\star\|$ is not monotonic for some q. This is resolved in Figure 2.7 by examining V_Q as defined by (2.29). We also note the phenomenon of faster convergence for even q over odd $q+1$; the reason for this is related to the modes of $I_n - LHL$ and was discussed in Section 2.5.3. However, increasing q on a whole lends itself to superior convergence compared to smaller q. As for the metric $g(z) - g(z^\star)$ in Figure 2.8, note that these values become significantly negative before eventually stabilizing around zero. The reason for this is simple: in order for the $\mathcal{Z}_q$ dynamics (2.28) in λ to "activate," the primal variable must become infeasible with respect to the box constraints. In this sense, the stabilization to zero of the plots in Figure 2.8 represents the trajectories converging to feasible points of $\mathscr{P}2$.

2.6.4 Robust DANA Implementation

Lastly, we provide a simulation justification for relaxing Assumption 2 via the method described in Remark 1. Figure 2.9 plots the error in the primal and dual states over time of

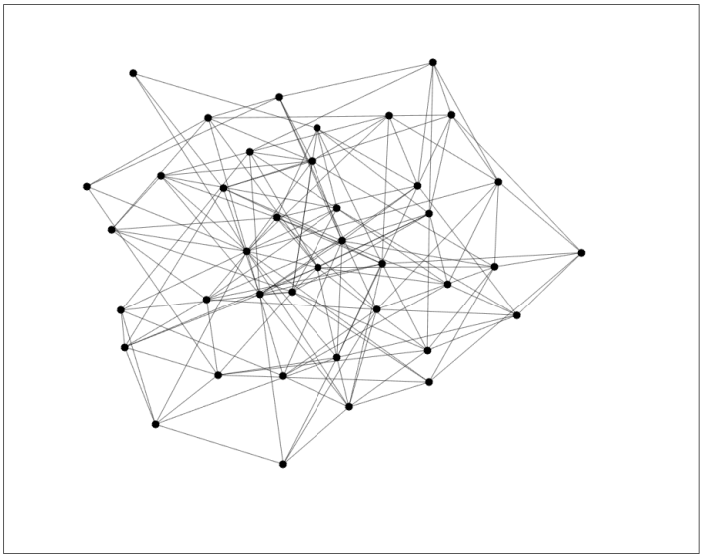


Figure 2.5. Communication graph for continuous-time numerical study: 40 nodes and 156 edges.

the modified "robust" method, which tends to approach zero for all observed values of q, and Figure 2.10 demonstrates that the violation of the equality constraint stablizes to zero very quickly. Noisy state perturbations are injected at $t = 25, 50, 75$, and we observe a rapid re-approach to the plane satisfying the equality constraint. However, even though the algorithm presents a faster convergence than gradient methods, here do not observe as clear of a relationship between performance and increased q as in previous settings. The investigation of the properties of this algorithm is left as future work.

Acknowledgements

The material in this chapter, in full, is a reprint of *Distributed Approximate Newton Algorithms and Weight Design for Constrained Optimization*, T. Anderson, C.Y. Chang and S. Martínez, Automatica, 109, article 108538, November 2019. A preliminary version of the work appeared in the proceedings of the Conference on Control Technology and Applications

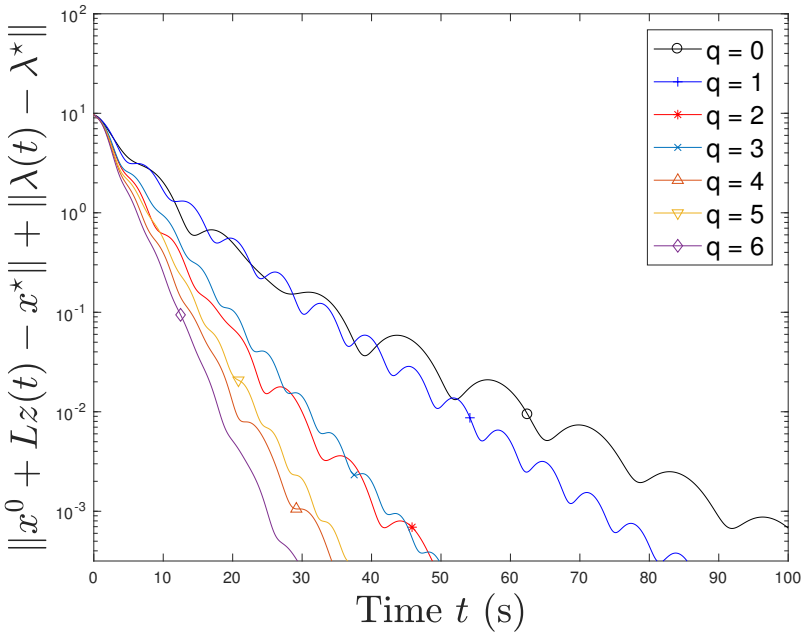


Figure 2.6. Error in the primal and dual state variables versus time for various q; $n = 40, |\mathscr{E}| = 156$.

(CCTA), Mauna Lani, HI, 2017, pp. 632-637, as *Weight Design of Distributed Approximate Newton Algorithms for Constrained Optimization*, T. Anderson, C.Y. Chang and S. Martínez. The dissertation author was the primary investigator and author of these papers.

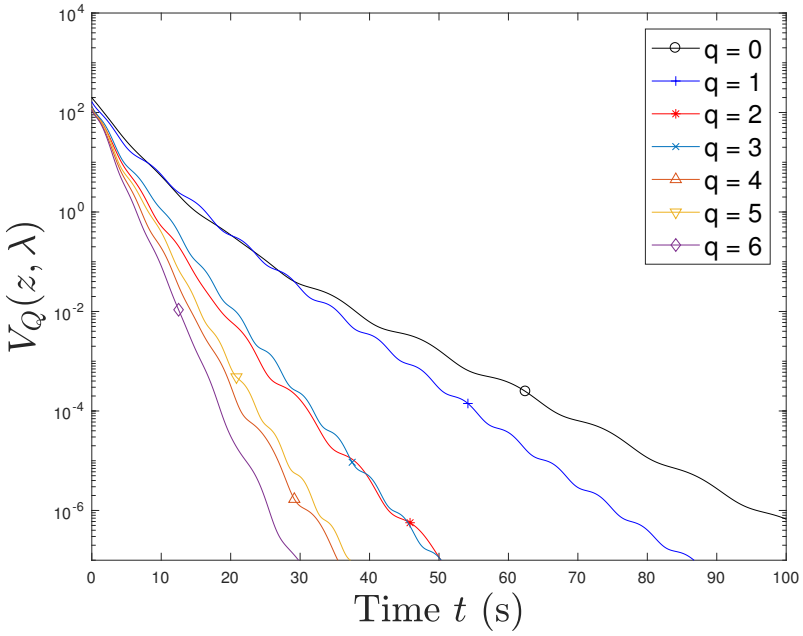


Figure 2.7. Value of the Lyapunov function V_Q versus time for various q; $n = 40, |\mathscr{E}| = 156$.

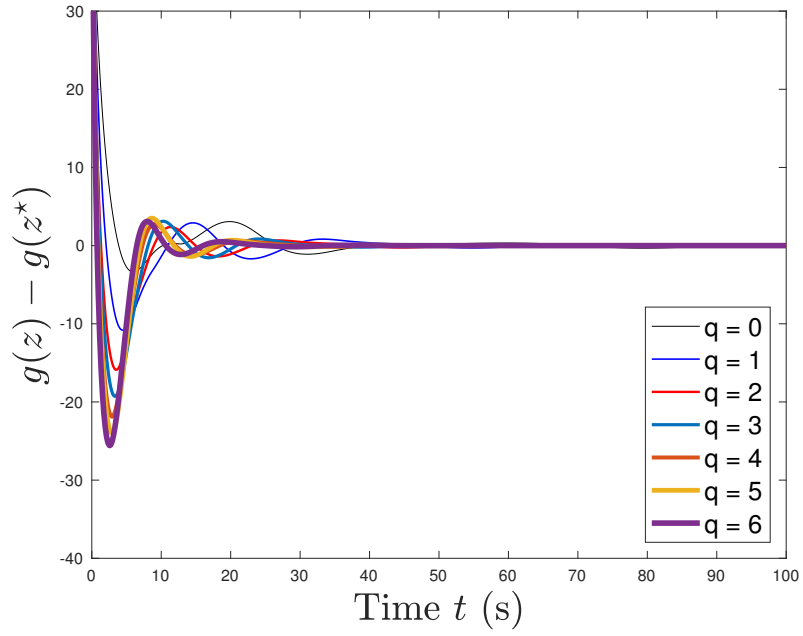


Figure 2.8. Value of the objective function versus time for various q; $n = 40, |\mathscr{E}| = 156$.

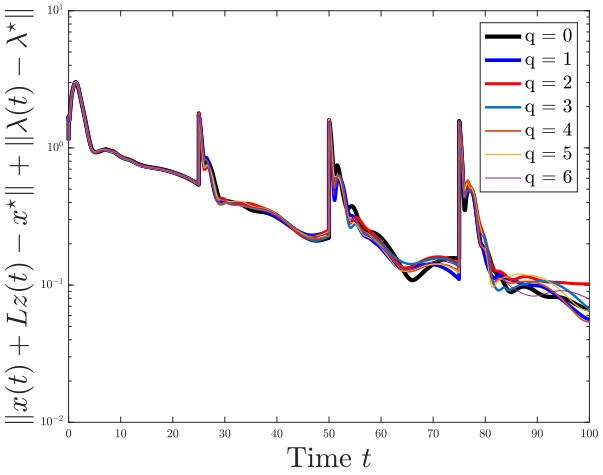


Figure 2.9. Error in the primal and dual states for a robust implementation of DANA; $n = 20, |\mathscr{E}| = 40$. Initialization does not satisfy Assumption 2, and perturbations are injected at $t = 25, 50, 75$.

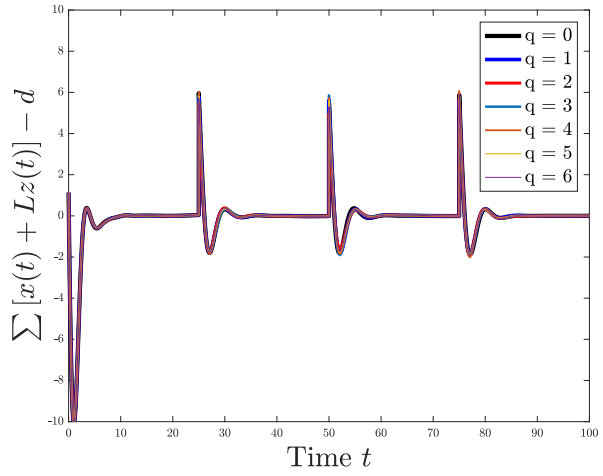


Figure 2.10. Violation of the resource constraint over time for robust DANA; $n = 20, |\mathscr{E}| = 40$. Perturbations are injected at $t = 25, 50, 75$.

Chapter 3

Distributed Stochastic Nested Optimization via Cubic Regularization

This chapter considers a nested stochastic distributed optimization problem. In it, approximate solutions to realizations of the inner-problem are leveraged to obtain a Distributed Stochastic Cubic Regularized Newton (DiSCRN) update to the decision variable of the outer problem. We provide an example involving electric vehicle users with various preferences which demonstrates that this model is appropriate and sufficiently complex for a variety of data-driven multi-agent settings, in contrast to non-nested models. The main two contributions of the chapter are: (i) development of local stopping criterion for solving the inner optimization problem which guarantees sufficient accuracy for the outer-problem update, and (ii) development of the novel DiSCRN algorithm for solving the outer-problem and a theoretical justification of its efficacy. Simulations demonstrate that this approach is more stable and converges faster than standard gradient and Newton outer-problem updates in a highly nonconvex scenario.

3.1 Bibliographical Comments

One of the most widely used stochastic optimization method is stochastic gradient-based (first-order) methods, see [13, 14, 44] as broad references. These methods are powerful because they necessitate only a small sampling of the data set to compute an update direction at each iterate. However, these first-order algorithms suffer from slow convergence around

saddle-points [40], which are disproportionately more present in higher-dimensional nonconvex problems [36]. By contrast, higher-order Newton-based methods tend to perform more strongly across applications in terms of number of calls to an oracle or total iterations, see [103, 111] for examples in stochastic non-strongly convex and nonconvex settings, respectively, and [6, 74, 96] for various multi-agent examples.

An issue with many of the aforementioned algorithms is they are vulnerable to slow convergence or instability in the presence of saddle-points and/or an ill-conditioned Hessian matrix. A growing body of works thus focuses on using a cubic-regularization term in the second-order Taylor approximation of the objective function. Nesterov and Polyak laid significant groundwork for this method in [78], and substantial follow-ups are contained in [25, 26], which study adaptive batch sizes and the effect of inexactness in the cubic submodel on convergence. Excitement about this topic has grown substantially in the last few years, with [24] showing how the global optimizer of the nonconvex cubic submodel can be obtained under certain initializations of gradient descent, and [94] being one of the first thorough analyses of the algorithm in the traditional stochastic optimization setting. In [28], the authors consider the stochastic setting from an adaptive batch-size perspective and [97] is, to our knowledge, the only existing work in a distributed application, with an alternative approach that allows for a communication complexity analysis. Both [28] and [97] assume convexity, and [97] is nonstochastic. As far as we know, no current work has unified *distributed*, *stochastic*, and *nonconvex* elements, particularly in a nested optimization scenario.

Statement of Contributions

We begin the chapter by formulating a nested distributed stochastic optimization problem, where approximate solutions to realizations of the inner-problem are needed to obtain iterative updates to the outer problem, and we motivate this model with an example based on electric vehicle charging preferences. The contributions of this chapter are then twofold. First, we

develop a stopping criterion for a Laplacian-gradient subsolver of the inner-problem. The stopping criterion can be validated locally by each agent in the network, and the relationship to solution accuracy aids the synthesis with the outer-problem update. Second, to that end, we formulate a distributed optimization model of the stochastic outer problem and develop a cubic regularization of its second-order approximation. This formulation lends itself to obtaining a Distributed Stochastic Cubic-Regularized Newton (DiSCRN) algorithm, and we provide theoretical justification of its convergence.

3.2 Problem Formulation

This section details the two problem formulations which are of interest, where the first problem $\mathscr{P}1$ takes the form of a stochastic approximation whose cost is a parameterization of the cost of the second problem $\mathscr{P}2$. Problem $\mathscr{P}2$ is a separable resource allocation problem in which n agents $i \in \mathscr{N}$ must collectively obtain a solution that satisfies a linear equality constraint while minimizing the sum of their local costs. (This problem commonly appears in real-time optimal dispatch for electric grids with flexible loads and distributed generators, see e.g. [1].) Thus, $\mathscr{P}1$ can be treated as a nested optimization, with an objective F that takes stochastic arguments, and is not necessarily available in closed form if $\mathscr{P}2$ cannot be solved directly and/or the distribution $\mathscr{D}$ being unknown. These problems are stated as

$$\mathscr{P}1: \min_{x\in\mathbb{R}^d} F(x) = \mathbb{E}_{\chi\sim\mathscr{D}}\left[F_\chi(x)\right].$$

$$\begin{aligned}\mathscr{P}2: \min_{p\in\mathbb{R}^n} f(x,p) &= \sum_{i=1}^{n} f_i(x,p_i),\\ \text{subject to } \sum_{i=1}^{n} p_i &= P_{\text{ref}} + \hat{\chi} = P_{\text{ref}} + \sum_{i=1}^{n} \hat{\chi}_i.\end{aligned}$$

In $\mathscr{P}1$, each $f_i : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$, and $F_\chi(x) \equiv f(x,p^\star)$, where $p^\star$ is the solution to $\mathscr{P}2$ for particular realizations $\hat{\chi}_i$, where $\chi_i \sim \mathscr{D}_i$, i.e. $\chi \sim \mathscr{D} = \mathscr{D}_1 \times \cdots \times \mathscr{D}_n$. The elements $p_i \in \mathbb{R}$ of $p \in \mathbb{R}^n$ and

terms $\hat{\chi}_i$ are each associated with and locally known by agents $i \in \mathcal{N}$, and $P_{\text{ref}} \in \mathbb{R}$ is a given constant known by a subset of agents (we discuss its interpretation shortly with an example). First, for F_χ to be well defined, it helps if solutions $p^\star$ to $\mathscr{P}2$ are unique for fixed x and $\hat{\chi}$, which we now justify with convexity assumptions for f_i.

Assumption 8. ***(Function Properties: Inner-Problem Argument).*** *The local cost functions f_i are twice differentiable and ω_i-strongly convex in p_i for any fixed x. Further, the second derivatives are lower and upper bounded:*

$$0 < \omega_i \leq \nabla^2_{p_i} f_i(x, p_i) \leq \theta_i, \qquad \forall x \in \mathbb{R}^d, p_i \in \mathbb{R}, \text{ and } i \in \mathcal{N}\,.$$

This implies $\forall x \in \mathbb{R}^d, p_i, \hat{p}_i \in \mathbb{R}$ and $i \in \mathcal{N}$:

$$\omega_i \|p_i - \hat{p}_i\| \leq \|\nabla_{p_i} f_i(x, p_i) - \nabla_{p_i} f_i(x, \hat{p}_i)\| \leq \theta_i \|p_i - \hat{p}_i\|.$$

We also use the shorthands $\omega \triangleq \min_i \omega_i$ and $\theta \triangleq \max_i \theta_i$.

This assumption will be required of our analysis in Section 3.3.1. We now state some additional assumptions.

Assumption 9. ***(Function Properties: Lipschitz Outer-Problem Argument).*** *The functions f_i have l_i-Lipschitz gradients and ρ_i-Lipschitz Hessians:*

$$\|\nabla f_i(x, p_i) - \nabla f_i(y, p_i)\| \leq l_i \|x - y\|, \qquad \forall x, y \in \mathbb{R}^d, \forall p_i \in \mathbb{R},$$
$$\|\nabla^2 f_i(x, p_i) - \nabla^2 f_i(y, p_i)\| \leq \rho_i \|x - y\|, \quad \forall x, y \in \mathbb{R}^d, \forall p_i \in \mathbb{R}.$$

We also use the shorthands $l \triangleq \max_i l_i$ and $\rho \triangleq \max_i \rho_i$.

Assumption 10. ***(Function Properties: Bounded Variance Outer-Problem Argument).*** *The*

function F_χ possesses the following bounded variance properties:

$$\mathbb{E}\left[\|\nabla F_\chi(x) - \nabla F(x)\|^2\right] \leq \sigma_1^2, \qquad \mathbb{E}\left[\|\nabla^2 F_\chi(x) - \nabla^2 F(x)\|^2\right] \leq \sigma_2^2,$$
$$\|\nabla F_\chi(x) - \nabla F(x)\|^2 \leq M_1 \text{ almost surely}, \qquad \|\nabla^2 F_\chi(x) - \nabla^2 F(x)\|^2 \leq M_2 \text{ almost surely}.$$

Assumption 11. ***(Function Properties: Lipschitz Interconnection of Variables).*** *The gradient and Hessian of the function f with respect to x are Lipschitz in p; that is, there exists constants $\psi_g, \psi_H > 0$ such that*

$$\|\nabla_x f(x,p) - \nabla_x f(x,\hat{p})\| \leq \psi_g \|p - \hat{p}\|,$$
$$\|\nabla^2_{xx} f(x,p) - \nabla^2_{xx} f(x,\hat{p})\| \leq \psi_H \|p - \hat{p}\|,$$
$$\forall x \in \mathbb{R}^d, p, \hat{p} \in \mathbb{R}^n.$$

Assumption 8 is relatively common in the convex optimization literature, and it lends itself to obtaining approximate solutions to $\mathscr{P}2$ very quickly with stopping criterion guarantees. Assumption 9 is unanimously leveraged in literature on Cubic-Regularized Newton methods, as the constant ρ pertains directly to the cubic submodel, while Assumption 10 is a common assumption in the stochastic optimization literature [94]. We note that, although Assumptions 9 and 10 do not give a direct relationship with the local functions $f_i(x, p_i)$, they do imply an implicit relationship between $x, p,$ and $\mathscr{D}$ in the sense that solutions $p^\star$ to $\mathscr{P}2$ (and therefore the distributions $\mathscr{D}_i$) must be "well-behaved" in some sense. This relationship, along with a broader interpretation of the model $\mathscr{P}1$ and $\mathscr{P}2$, is illustrated more concretely in the following real-world power distribution example.

Example 1. ***(EV Drivers with PV Generators).*** *Consider two EV drivers who each have an EV charging station and a PV generator. The goal of this small grid system is to consume net zero power from the perspective of the tie line to the bulk grid, thus $P_{\text{ref}} = 0$. The distributions $\mathscr{D}_1, \mathscr{D}_2$ represent the power output distributions of the PVs, and we consider two scenarios for these in this example: (1) a "sunny day" scenario, where the realizations $\chi_1, \chi_2 \sim \mathscr{D}_1, \mathscr{D}_2$ of PVs 1 and 2 are deterministic, and (2) a "cloudy day" scenario, where intermittent cloud cover induces*

some uncertainty in the moment-to-moment PV generation.

Let $A \in \{sunny, cloudy\}$ indicate the weather forecast. The model is then fully described as

$$\mathscr{D}_i = \begin{cases} \delta_{1.5}, & A = sunny, \\ \mathscr{U}[0, 1.5], & A = cloudy \end{cases} \quad \text{for both i=1,2,}$$

$$f_1(x, p_1) = (2x + p_1 - 1)^2,\ f_2(x, p_2) = (x + p_2 - 2)^2.$$

For $x = 0$, these quadratic functions[1] have local minima at $p^\star = (p_1^\star, p_2^\star) = (1, 2)$, which is interpreted as drivers 1 and 2 preferring to charge at rates of 1 unit and 2 units, respectively, if there are no external incentives. On a sunny day, both PVs deterministically produce $\hat{\chi}_1, \hat{\chi}_2 = 1.5$, which effectively balances the unconstrained $p^\star$ and both drivers can charge at their preference to maintain $\sum_i p_i = P_{\text{ref}} + \sum_i \hat{\chi}_i$.

However, on cloudy days the generation of the PVs is no longer deterministic. Thus, the variable x comes in to play, which can represent a government credit that the drivers value differently. The role of x is to shift the cost functions such that the unconstrained minima are near lower charging values in consideration of the lower expected generation from PVs 1 and 2. The optimal $x^\star$ to $\mathscr{P}1$ is the value which gives the lowest expected cost of an instance of $\mathscr{P}2$ given $\hat{\chi}_1, \hat{\chi}_2$ realizations from the $A = cloudy$ distributions $\mathscr{D}_1, \mathscr{D}_2$. A more complete model of $\mathscr{P}2$ could include power flow constraints; in this work, we relax these for simplicity.

3.3 Distributed Formulation and Algorithm

In this section, we develop the inner-loop algorithm used to solve $\mathscr{P}2$. We then synthesize inexact solutions to $\mathscr{P}2$ with the DiSCRN algorithm for $\mathscr{P}1$.

[1] See [9] for an example where quadratic costs to EV users are induced by resistive energy losses in the battery model and [5] for a broad reference on modeling generator dispatch.

3.3.1 Inner Loop Gradient Solver

For this section, consider x to be fixed and known by all agents. Further, let $\hat{\chi}_i$ be fixed (presumably from a realization of $\mathscr{D}_i$) and known only to agent i. We adopt the following assumption on the initial condition p^0.

Assumption 12. ***(Feasibility of Inner-Problem Initial Condition).*** *The agents are endowed with an initial condition which is feasible with respect to the constraint of $\mathscr{P}2$; that is, they each possess elements p_i^0 of a p^0 satisfying $\mathbf{1}_n^\top p^0 = P_{\text{ref}} + \hat{\chi}$.*

The assumption is easily satisfied in practice by communicating P_{ref} to one agent i and setting $p_i^0 = P_{\text{ref}} + \hat{\chi}_i$, with all other agents j using $p_j^0 = \hat{\chi}_j$. An alternative to this assumption consists of reformulating $\mathscr{P}2$ with distributed constraints and using a dynamic consensus algorithm as in [34], which would still retain exponential convergence. We impose Assumption 12 for simplicity. Finally, we assume connectedness of the communication graph:

Assumption 13. ***(Graph Properties).*** *The communication graph $\mathscr{G}$ is connected and undirected; that is, a path exists between any pair of nodes and, equivalently, its Laplacian matrix $L = L^\top \succeq 0$ has rank $n-1$ with eigenvalues $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$.*

The discretized Laplacian-flow dynamics are given by:

$$p^+ = p - \eta L \nabla_p f(x, p). \tag{3.1}$$

Note that these dynamics are distributed, as the sparsity of L implies each agent need only know $\nabla_{p_i} f_i(x, p_i)$ and $\nabla_{p_j} f(x, p_j)$ for $j \in \mathscr{N}_i$ to compute p_i^+. We now justify convergence of (3.1) to the solution $p^\star$ of $\mathscr{P}2$:

Proposition 1. ***(Convergence of Discretized Laplacian Flow).*** *Let $p^\star \in \mathbb{R}^n$ be the unique minimizer of $\mathscr{P}2$. Given Assumption 12 on the feasibility of the initial condition, Assumption 13 on connectivity of the communication graph, and Assumption 8 on the Lipschitz gradient*

condition of the function gradients, then, under the dynamics (3.1) *with* $0 < \eta < \frac{2}{\theta\lambda_n}$, *p converges asymptotically to* $p^\star$.

Proof. Using a standard quadratic expansion around the current iterate p (see e.g. §9.3 of [17]) and Lipschitz bounds yields $f(p^+) - f(p) \leq \theta\eta^2/2\|L\nabla_p f(x,p)\|^2$
$-\eta\nabla_p f(x,p)^\top L\nabla_p f(x,p)$. Careful treatment of the eigenspace of L and some algebraic manipulation shows that $f(p^+) - f(p)$ is strictly negative for η as in the statement. A more detailed proof can be found in [7]. □

We now provide an additional result on exponential convergence of the state error with a further-constrained step size as compared to the statement in Proposition 1.

Proposition 2. ***(Exponential Convergence with Bounded Error).*** *Let Assumptions 12, 13, and 8 hold as before. For* $0 < \eta < 2\omega\lambda_2/\theta^2\lambda_n^2$, *the quantity* $\|p - p^\star\|$ *converges exponentially to zero under the dynamics* (3.1). *For* $\eta = \omega\lambda_2/\theta^2\lambda_n^2$, *the rate is* $\|p^+ - p^\star\| \leq \sqrt{1 - \omega^2\lambda_2^2/\lambda_n^2\theta^2}\|p - p^\star\|$, *and* $\|p^K - p^\star\| \leq \Delta$ *for*
$K \geq \log(\Delta/\|p^0 - p^\star\|)/\log(\sqrt{1 - \omega^2\lambda_2^2/\lambda_n^2\theta^2})$.

Proof. Consider $V(p) = \|p - p^\star\|^2$. Substituting (3.1) and applying bounds via eigenvalues of L, using Assumption 8, and ν-strong function convexity, we get $V(p^+) \leq (\eta^2\lambda_n^2\theta^2 - 2\eta\lambda_2\omega + 1)V(p)$, with $0 < \eta < 2\lambda_2\omega/\lambda_n^2\theta^2$. The choice of $\eta = \omega\lambda_2/\lambda_n^2\theta^2$ implies the exponential convergence as in the statement. See [7] for more details. □

We note that the results of Propositions 1 and 2 simply build on a Laplacian-projected version of vanilla gradient descent. However, it lays some basic groundwork and supplements our main results in the next subsection.

With this, we are ready to transition to the discussion on obtaining a DiSCRN update to $\mathscr{P}1$.

3.3.2 Outer-Loop Cubic-Newton Update

We endow each agent with a local copy x_i of the variable x, and we let $\mathbf{x} \in \mathbb{R}^{nd}$ be the stacked vector of these local copies. Thus, a distributed reformulation of $\mathscr{P}1$ is

$$\overline{\mathscr{P}1}: \min_{\mathbf{x}\in\mathbb{R}^{nd}} \quad \bar{F}(\mathbf{x}) = \mathbb{E}_{\chi\sim\mathscr{D}}\left[\bar{F}_\chi(\mathbf{x})\right], \\ \text{subject to} \quad (L\otimes I_d)\,\mathbf{x} = \mathbf{0}_{nd},$$

where $\bar{F}_\chi : \mathbb{R}^{nd} \to \mathbb{R}$ is analagous to $F_\chi : \mathbb{R}^d \to \mathbb{R}$ in the sense that each agent evaluates $f_i(x_i, p_i^\star)$ with its local copy of x_i. Note that the constraint $(L\otimes I_d)\,\mathbf{x} = \mathbf{0}_{nd}$ imposes $x_i = x_j, \forall i, j$ (Assumption 13), so $\bar{F}_\chi$ and F_χ are equivalent in the agreement subspace (and $\overline{\mathscr{P}1}$ is equivalent to $\mathscr{P}1$). Since our problem is nested and stochastic, there is a lack of access to a closed form expression for $\bar{F}$ and $\bar{F}_\chi$. Thus, we introduce an *empirical-risk*, *approximate* objective function. To this end, let $F^S(\mathbf{x}) = 1/S\sum_{s=1}^S F^\Delta_{\chi^s}(\mathbf{x})$ be the approximation of $\bar{F}$ for S samples of $\chi^s \sim \mathscr{D}$, where $F^\Delta_{\chi^s} \equiv \sum f_i(x_i, \tilde{p}_i^s)$ and $\|\tilde{p}^s - p^\star\| \leq \Delta$ for realization χ^s. In this sense, $F^\Delta_{\chi^s}$ implicitly depends on $\tilde{p}^s$, and the Δ superscript is a slight abuse of notation. For now, the reader can consider Δ to be a sufficiently small design parameter describing the inexactness of the obtained solutions to $\mathscr{P}2$; we build on this later. Ultimately, we intend to use batches of F^S rather than the exact $\overline{F}$ to implement DiSCRN. Consider then the cubic regularized submodel of F^S at some $\mathbf{x}^k$:

$$m_S^k(\mathbf{x}) = F^S(\mathbf{x}^k) + (\mathbf{x}-\mathbf{x}^k)^\top g^k + \frac{1}{2}(\mathbf{x}-\mathbf{x}^k)^\top H^k(\mathbf{x}-\mathbf{x}^k) + \sum_{i=1}^n \frac{\rho_i}{6}\|x_i - x_i^k\|^3, \tag{3.2}$$

where $g^k = \nabla F^S(\mathbf{x}^k), H^k = \nabla^2 F^S(\mathbf{x}^k)$. Note that there is a slight difference between (3.2) and the more standard cubic submodel (1.4) in that the regularization terms are *directly separable*; this is crucial for a distributed implementation, and our forthcoming analysis justifies that convergence can still be established. We are interested in finding $\mathbf{x}^+$ which minimizes (3.2) in the agreement subspace:

$$\mathscr{P}3: \min_{\mathbf{x}\in\mathbb{R}^{nd}} \quad m_S^k(\mathbf{x}), \quad \text{subject to } (L\otimes I_d)\,\mathbf{x} = \mathbf{0}_{nd}\,. \tag{3.3}$$

Therefore, we prescribe the Decentralized Gradient Descent dynamics from [113]:

$$\mathbf{x}^{+,t+1} = W\,\mathbf{x}^{+,t} - \alpha_t \nabla_{\mathbf{x}} m_S^k(\mathbf{x}^{+,t}), \tag{3.4}$$

where $W = I_{nd} - (1/\lambda_n L \otimes I_d)$ and $\alpha_t \sim 1/t$. Per Proposition 3 and Theorem 2 of [113], $\mathbf{x}^{+,t}$ under the dynamics (3.2) converges asymptotically to a stationary point of $\mathscr{P}3$ with $O(1/k)$ convergence in the agreement subspace, i.e. $\|x_i - \bar{x}\|$ approaches zero at a rate $O(1/k)$, where $\bar{x} = \text{mean}\,(x_i)$.

We remark that one could formulate the Lagrangian of $\mathscr{P}3$ and use a saddle-point method to obtain a useful update $\mathbf{x}^+$. This is more parallel to the work of [24], which achieves the global solution via gradient descent in the centralized setting. However, even the existence of a Lagrangian saddle-point is in question when the duality gap is nonzero, so further study is required on that approach.

Our aim is to obtain an ε-second-order stationary point of $\mathscr{P}1$, as in Definition 1. The above discussion serves to set up the following condition on $\mathbf{x}^{k+1}$:

Condition 1. ***(Subsolver Output).*** *Let $\mathbf{x}^{k+1}$ be the output of a subsolver for $\mathscr{P}3$. Then,*

(i) $\mathbf{x}^{k+1}$ *satisfies* $(L \otimes I_d)\,\mathbf{x}^{k+1} = \mathbf{0}_{nd}$.

(ii) For an arbitrarily small constant $c > 0$ and some $\varepsilon > 0$, $\mathbf{x}^{k+1}$ satisfies $m_S^k(\mathbf{x}^{k+1}) - m_S^k(\mathbf{x}^k) < -c\varepsilon\|\,\mathbf{x}^{k+1} - \mathbf{x}^k\,\| - c\sqrt{\rho\varepsilon}\|\,\mathbf{x}^{k+1} - \mathbf{x}^k\,\|^2$.

Part (i) is implied in a linear convergence sense by the result of [113] for the subsolver (3.4). The (ii) condition is straightforwardly implied by any subsolver that is guaranteed to strictly decrease m_S^k, e.g. (3.4), because c can be taken arbitrarily small. However, it can be seen in the statement of Theorem 4 that small c implies a direct tradeoff with Δ (becomes small) and/or S (becomes large).

We now give a brief outline of the entire algorithm.

DiSCRN Algorithm

1. Initialize $\mathbf{x}^0$ s.t. $(L \otimes I_d)\,\mathbf{x}^0 = \mathbf{0}_{nd}$
2. Realize χ^s and initialize p^0 per Assumption 12
3. Implement (3.1) until $|p_i^+ - p_i| \leq \Delta\eta\lambda_2\omega/\sqrt{n}, \forall i$
4. Repeat from step 2 S times, storing $\tilde{p}^s \leftarrow p^+$ at each s
5. Compute locally required elements of g^k, H^k
6. Compute an $\mathbf{x}^{k+1}$ satisfying Condition 1, e.g. via (3.4); repeat from step 2

The DiSCRN Algorithm describes a fully distributed algorithm, as each step can be performed with only local information. Ostensibly, $\mathbf{x}^0$ could be initialized arbitrarily, but the first outer-loop would be a "garbage" update until agreement is obtained in step 6. Note that Step 3 relates to a distributed stopping criterion for the subsolver of $\mathscr{P}2$; this condition produces a solution p^+ in finite iterations which is sufficiently close to $p^\star$ for the sake of our analysis. This is detailed more in Theorem 4 and its proof.

Condition 2. ***(Assumptions and Conditions for Theorem 4).*** *Let F satisfy Assumption 9, on Lipschitz gradients and Hessians, and Assumption 10, on variance conditions, and let f satisfy Assumption 11, on Lipschitz interconnection of x and p, and Assumption 8, on the Lipschitz condition of the function gradients with respect to p. Further, let Assumption 12, on the feasibility of the initial condition for $\mathscr{P}2$, and Assumption 13 on connectivity of the communication graph, each hold. Let $\mathbf{x}^{k+1}$ be the output of a subsolver for $\mathscr{P}3$ that satisfies Condition 1 with c, and let $\tilde{p}^s \leftarrow p^+$, where p^+ is the returned value under the dynamics* (3.1) *satisfying* $|p_i^+ - p_i| \leq \Delta\eta\lambda_2\omega/\sqrt{n}, \forall i$.

Theorem 4. ***(Convergence of DiSCRN).*** *Let the circumstances of Condition 2 apply here. For* $S \geq \max\{\frac{M_1}{\bar{c}\varepsilon}, \frac{\sigma_1^2}{\bar{c}^2\varepsilon^2}, \frac{M_2}{\bar{c}\sqrt{\rho\varepsilon}}, \frac{\sigma_2^2}{\bar{c}^2\rho\varepsilon}\} O(\log((\varepsilon^{1.5}\zeta\bar{c})^{-1}))$ *with* $\bar{c}\varepsilon + \psi_g\Delta \leq c\varepsilon$ *and* $\bar{c}\sqrt{\rho\varepsilon} + \psi_H\Delta \leq$

$c\sqrt{\rho\varepsilon}$, then for all $\zeta > 0$ each x_i asymptotically approaches a common ε-second-order stationary point $\tilde{x}$ of F with probability $\geq 1-\zeta$ under the DiSCRN algorithm dynamics.

Proof. First, we aim to obtain the bound $\|\tilde{p}^s - p^\star\| \leq \Delta$ for each instance s of $\mathscr{P}2$. The Lipschitz condition of Assumption 8 implies

$$\begin{aligned}\omega\|p-p^\star\| &\leq \|\nabla_p f(x,p) - \nabla_p f(x,p^\star)\|,\\ \lambda_2\omega\|p-p^\star\| &\leq \|L(\nabla_p f(x,p) - \nabla_p f(x,p^\star))\|\\ &= \|L\nabla_p f(x,p)\| = 1/\eta\|p^+ - p\| \leq \Delta\lambda_2\omega.\end{aligned}$$

Finally, $1/\sqrt{n}$ comes from breaking $p^+ - p$ into components and since, for $v \in \mathbb{R}^n$, if $|v_i| \leq c/\sqrt{n}$ implies $\|v\| \leq c$.

Turning to $\mathscr{P}1$, let $g_\star^k = \frac{1}{S}\sum_{s=1}^S \sum_i \nabla_{x_i} f_i(x_i^k, p_i^\star)$ and $H_\star^k = \frac{1}{S}\sum_{s=1}^S \sum_i \nabla^2_{x_i x_i} f_i(x_i^k, p_i^\star)$. Lemma 4 of [94] justifies that for arbitrary $\bar{c} > 0$, choosing $S \geq \max\{\frac{M_1}{\bar{c}\varepsilon}, \frac{\sigma_1^2}{\bar{c}^2\varepsilon^2}, \frac{M_2}{\bar{c}\sqrt{\rho\varepsilon}}, \frac{\sigma_2^2}{\bar{c}^2\rho\varepsilon}\} O(\log((\varepsilon^{1.5}\zeta\bar{c})^{-1}))$ implies that $\|g_\star^k - \nabla\bar{F}(\mathbf{x}^k)\| \leq \bar{c}\varepsilon$ and $\|(H_\star^k - \nabla^2_{\mathbf{xx}}\bar{F}(\mathbf{x}^k))v\| \leq \bar{c}\varepsilon\sqrt{\rho\varepsilon}\|v\|, \forall v$ with probability $1-\zeta$.

Let $\phi_g^k = g^k - g_\star^k, \phi_H^k = H^k - H_\star^k$, where g^k and H^k use the inexact estimates $\tilde{p}^s$ satisfying $\|\tilde{p}^s - p^\star\| \leq \Delta$. Substitutions and applying Assumption 11 gives:

$$\begin{aligned}&\|g^k - \nabla_{\mathbf{x}}\bar{F}(\mathbf{x}^k)\| \leq \|g_\star^k - \nabla_{\mathbf{x}}\bar{F}(\mathbf{x}^k)\| + \|\phi_g^k\| \leq \bar{c}\varepsilon + \psi_g\Delta,\\ &\|(H^k - \nabla^2_{\mathbf{xx}}\bar{F}(\mathbf{x}^k))v\| \leq \|(H_\star^k - \nabla^2_{\mathbf{xx}}\bar{F}(\mathbf{x}^k))v\| + \|\phi_H^k\| \leq \bar{c}\sqrt{\rho\varepsilon} + \psi_H\Delta, \forall v.\end{aligned}$$

Next, let $\xi^k := \mathbf{x}^{k+1} - \mathbf{x}^k$ for notational convenience. The separable cubic regularized terms of

m_S^k can be used to bound the true function value:

$$\bar{F}(\mathbf{x}^{k+1}) \leq \bar{F}(\mathbf{x}^k) + \nabla\bar{F}(\mathbf{x}^k)^\top \xi^k + \xi^{k\top}\nabla^2\bar{F}(\mathbf{x}^k)\xi^k + \sum_i \rho_i/6\|x_i^{k+1} - x_i^k\|^3 \Rightarrow$$

$$\begin{aligned}
\bar{F}(\mathbf{x}^{k+1}) - \bar{F}(\mathbf{x}^k) &\leq m_S^k(\mathbf{x}^{k+1}) - m_S^k(\mathbf{x}^k) + (\nabla\bar{F}(x^k) - g^k)^\top \xi^k + 1/2\xi^{k\top}(\nabla^2\bar{F}(x^k) - H^k)\xi^k \\
&\leq m_S^k(\mathbf{x}^{k+1}) - m_S^k(\mathbf{x}^k) + (\bar{c}\varepsilon + \psi_g\Delta)\|\xi^k\| + (\bar{c}\sqrt{\rho\varepsilon} + \psi_H\Delta)\|\xi^k\|^2 \\
&\leq m_S^k(\mathbf{x}^{k+1}) - m_S^k(\mathbf{x}^k) + c\varepsilon\|\xi^k\| + c\sqrt{\rho\varepsilon}\|\xi^k\|^2 < 0,
\end{aligned}$$

where the first inequality is implied by breaking up $\bar{F}_\chi(\mathbf{x})$ in to its separable local functions and applying Assumption 9 and noting that the inequality carries through the expectation operator. Subsequent inequalities are directly obtained via substitutions. The lefthand inequality of the final line stems from the Theorem statement, and the righthand inequality of the final line from (ii) of Condition 1. □

3.4 Simulation

We consider a synthetic nonconvex case for our simulation study. The cost functions f_i can be represented as:

$$f_i(x, p_i) = \frac{1}{2}\alpha_i(x)p_i^2 + \beta_i(x)p_i + \gamma_i.$$

Each $\alpha_i : \mathbb{R} \to \mathbb{R}$ is quartic in x and generated according to (3.4), where each a_i^2 is determined such that $\min_x\ \alpha_i(x) = \omega_i > 0$ with $\omega_i \in \mathscr{U}[1,5]$ per Assumption 8. The $\beta_i : \mathbb{R} \to \mathbb{R}$ are (possibly nonconvex) quadratic, and $\gamma_i = 0$.

$$\begin{aligned}
&\alpha_i(x) = a_i^1(x - z_i^1)(x - z_i^2)(x - z_i^3)(x - z_i^4) + a_i^2, \\
&a_i^1 \in \mathscr{U}[0.5, 1.5], z_i^1 \in \mathscr{U}[-2,-1], z_i^2 \in \mathscr{U}[-1,0], z_i^3 \in \mathscr{U}[0,1], z_i^4 \in \mathscr{U}[1,2], \\
&\beta_i(x) = b_i^1(x - z_i^5)(x - z_i^6), \quad b_i^1 \in \mathscr{U}[-1,1], z_i^5 \in \mathscr{U}[-2,0], z_i^6 \in \mathscr{U}[0,2],
\end{aligned}$$

We compare our DiSCRN method with gradient-based and Newton-based updates of the

same batch sizes, where the gradient-like and Newton-like updates are computed via:

$$m_g^k(\mathbf{x}) = F^S(\mathbf{x}^k) + (\mathbf{x}-\mathbf{x}^k)^\top g^k + \sum_i \frac{\eta_g}{2}\|x_i - x_i^k\|^2,$$
$$m_H^k(\mathbf{x}) = F^S(\mathbf{x}^k) + (\mathbf{x}-\mathbf{x}^k)^\top g^k + \frac{1}{2}(\mathbf{x}-\mathbf{x}^k)^\top H^k(\mathbf{x}-\mathbf{x}^k) + \sum_i \frac{\eta_H}{2}\|x_i - x_i^k\|^2,$$

We obtain $\mathbf{x}^{k+1}$ empirically for all three methods by implementing (3.4) until the updates become very small. We found that both η_g and η_H must be sufficiently large to ensure stability, and $\nabla_x^2 F(x) \succ -\eta_H I_d$ to ensure $m_H^k(x)$ bounded. We take $\Delta = 0.1, S = 20, n = 40, |\,\mathscr{E}\,| = 120, P_{\text{ref}} = 40, \mathscr{D}_i = \mathscr{U}[0,1.5]\ \forall i, \rho = 50, \eta_g = 100, \eta_H = 50$.

We note substantially improved performance of DiSCRN over the more traditional gradient-based and Newton-based approaches. In particular, the trajectory finds a minimizer in roughly half and one-third the number of outer-loop iterations required by Newton and gradient, respectively. It is clear that, for $x^{k+1} \approx x^k$, the cubic regularization is less dominant than the squared regularizations, allowing the DiSCRN trajectory to be influenced more by the problem data g^k, H^k. As for the parameters (ρ, η_g, η_H), $\eta_H = 50$ and $\eta_g = 100$ were roughly the lowest possible values without inducing instability. By contrast, reducing ρ to values $\sim 10^{-1}$ was still stable for DiSCRN. We noticed a clear tradeoff between S and Δ, with small $S \sim 10^0$ requiring $\Delta \sim 10^{-1}$ to converge and large $S \sim 10^3$ converging even for large $\Delta \sim 10^2$, which is implied by Theorem 4. Finally, DiSCRN achieves reduced disagreement compared to gradient and Newton; this could be in part due to (3.4) finding a stationary point of $\mathscr{P}3$ faster, allotting more iterations where the consensus terms dominate the update.

Acknowledgements

The material in this chapter, in full, is being revised and prepared for submission to the Systems & Control Letters. It may appear as *Distributed Stochastic Nested Optimization via Cubic Regularization*, T. Anderson and S. Martínez. The dissertation author was the primary

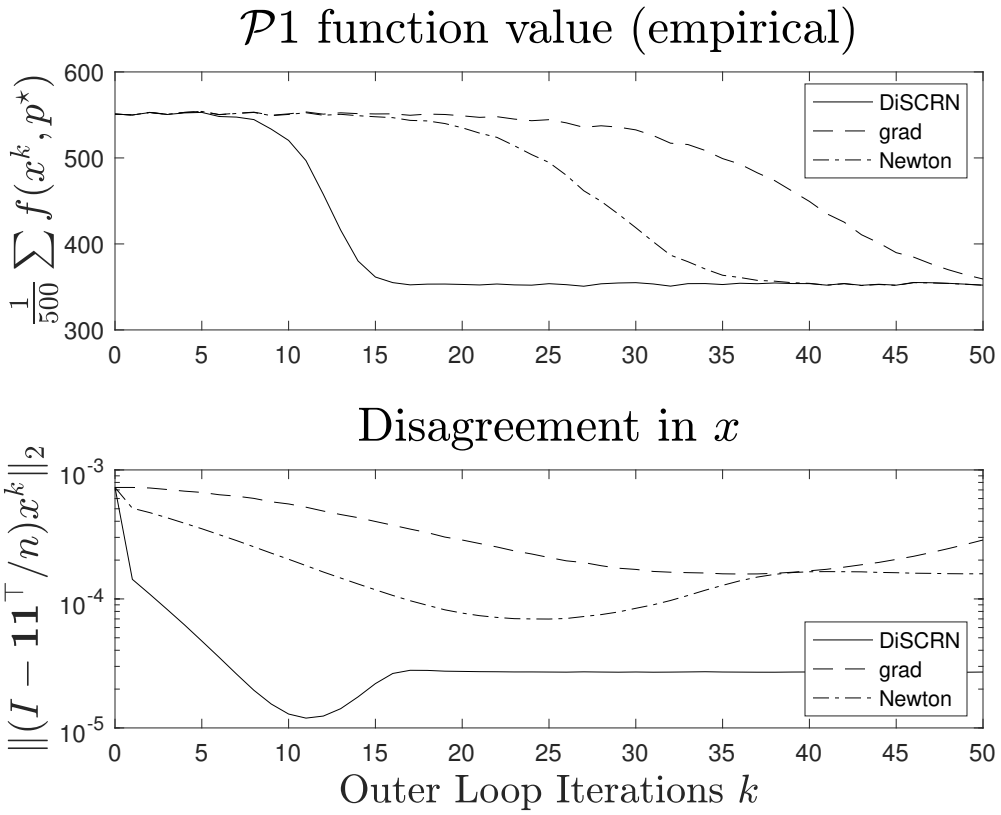


Figure 3.1. Comparison of CRN method with gradient-based and Newton-based approaches. **Top:** empirical approximation of $F(x^k)$, obtained by averaging $f(x^k, p^\star)$ over 500 realizations of $\mathscr{P}2$ at each k. **Bottom:** agents' disagreement on the value of x, quantified by $\|(I - \mathbf{1}\mathbf{1}^\top/n)x^k\|_2$.

investigator and author of this paper.

Chapter 4

Distributed Resource Allocation with Binary Decisions via Newton-like Neural Network Dynamics

This chapter aims to solve a distributed resource allocation problem with binary local constraints. The problem is formulated as a binary program with a cost function defined by the summation of agent costs plus a global mismatch/penalty term. We propose a modification of the Hopfield Neural Network (HNN) dynamics in order to solve this problem while incorporating a novel Newton-like weighting factor. This addition lends itself to fast avoidance of saddle points, which the gradient-like HNN is susceptible to. Turning to a multi-agent setting, we reformulate the problem and develop a distributed implementation of the Newton-like dynamics. We show that if a local solution to the distributed reformulation is obtained, it is also a local solution to the centralized problem. A main contribution of this work is to show that the probability of converging to a saddle point of an appropriately defined energy function in both the centralized and distributed settings is zero under light assumptions. Finally, we enlarge our algorithm with an annealing technique which gradually learns a feasible binary solution. Simulation results demonstrate that the proposed methods are competitive with centralized greedy and SDP relaxation approaches in terms of solution quality, while the main advantage of our approach is a significant improvement in runtime over the SDP relaxation method and the distributed quality of implementation.

4.1 Bibliographical Comments

Quadratic programs with nonconvex binary constraints are known to be NP-hard in general, see [27, 65]. In this chapter, we consider a problem which is quite applicable to the economic dispatch problem in power networks, see [**?**, **?**, 56] for recent examples in microgrid environments and [6] for a distributed Newton-like method in a more abstract setting. However, none of these examples address devices with binary constraint sets. The binary problem is, however, desirable to approach in a distributed context [109, 110]. Greedy algorithms [35] have been proposed for binary programs, such as the well-known Traveling Salesman Problem (TSP), but it is well documented that these methods can greatly suffer in performance [52] except in cases where the cost function is submodular [76, 89]. A more modern approach to solving optimization problems with a binary feasibility set is to cast them as a semidefinite program (SDP) with a nonlinear rank constraint, see [16, 82, 100] for some classical references or [69, 102] for more recent work on the topic. By relaxing the rank constraint, a convex problem is obtained whose solution can be shown to be equal to the optimal dual value of the original problem, see e.g. [81]. However, it is necessary in these approaches to either impose a single centralized coordinator to compute the solution and broadcast it to the actuators or agents, or schedule computations, which suffers from scalability issues, privacy concerns, and does not enjoy the simpler and more robust implementation of a distributed architecture in a large network.

Neuro-dynamic programming is a different paradigm for addressing nonconvex problems with computational tractability, see [11] for a broad reference. A neural-network based method for binary programs was first developed by Hopfield in [57], which was originally proposed in order to address TSPs. We refer to this method from here on as a Hopfield Neural Network (HNN). This method provided a completely different avenue for approaching binary optimizations, and followup works are found in [10, 60, 71, 90]. These works formalize and expand the framework in which the HNN method is applicable. However, these algorithms essentially implement a gradient-descent on an applicable nonconvex energy function, which is susceptible to being

slowed down by convergence to saddle-points. There are avenues for Newton-like algorithms in nonconvex environments to address this issue, which incorporate some treatment of the negative Hessian eigenvalues in order to maintain a monotonic descent of the cost function, see e.g. [36, 46]. A recently developed method employs a Positive-definite Truncated inverse (PT-inverse) operation on the Hessian of a nonconvex energy or cost function in order to define a nonconvex Newton-descent direction [85], although the technique does not presently address binary settings. Perhaps more importantly, all variants of existing HNN methods and the aforementioned works for nonconvex Newton-like algorithms are framed for centralized environments in which each agent knows global information about the state of all other agents, which is not scalable.

Statement of Contributions

The contributions of this chapter are threefold. We start by considering a binary programming problem formulated as a summation of local costs plus a squared global term. By leveraging a specific choice for the cost functions, we adapt the setting to an HNN framework. Then, we propose a novel modification of the dynamics with a PT-inverse of the Hessian of an appropriate energy function to define centralized NEWTON-LIKE NEURAL NETWORK (NNN-C). We prove a rigorous convergence result to a local minimizer, thus excluding saddle-points, with probability one, given some mild assumptions on the algorithm parameters and initial condition. Thirdly, we reformulate the problem so that it is solvable via a distributed algorithm by means of an auxiliary variable. We show that local solutions of the distributed reformulation are equivalent to local solutions of the centralized one, and we define a corresponding energy function and distributed algorithm for which we show convergence to a local minimizer with probability one. Simulations validate that our method is superior to SDP relaxation approaches in terms of runtime and scalability and outperforms greedy methods in terms of scalability.

4.2 Problem Statement and Dual Problem

Here, we formally state the nonconvex optimization problem we wish to solve and formulate its dual for the sake of deriving a lower bound to the optimal cost.

We aim to find an adequate solution to a resource allocation problem where the optimization variables take the form of binary decisions over a population of n agents. We note that the problem we consider is applicable to generator dispatch and active device response in an economic dispatch power systems setting [1], but the remainder of the chapter will frame it primarily as resource allocation. Let each agent $i \in \{1,\dots,n\}$ be endowed with a decision variable x_i and a cost $c_i \in \mathbb{R}$, a value which indicates the incremental cost of operating in the $x_i = 1$ state versus the $x_i = 0$ state. We do not impose a sign restriction on c_i, but this may be a common choice in the power systems setting where $x_i = 1$ represents an "on" device state and $x_i = 0$ represents "off." Additionally, each agent is endowed with a parameter p_i which represents some incremental consumption or generation quantity when operating in the $x_i = 1$ state versus $x_i = 0$ and also a passive cost d_i.

We are afforded some design choice in the cost function models for $x_i \notin \{0,1\}$, and for each $i \in \{1,\dots,n\}$, so we design abstracted cost functions $f_i : [0,1] \to \mathbb{R}$ that satisfy $f_i(0) = d_i$ and $f_i(1) = c_i + d_i, \forall i$. This design choice is intrinsic to a cost model for any separable binary decision optimization context. In particular, the value of $f_i(x_i)$ for any $x_i \notin \{0,1\}$ is only relevant to the algorithm design, but need not have a physical interpretation or pertain to the optimization model since these points are infeasible. With this in mind, we enlarge the cost model by adopting the following:

Assumption 14. ***(Quadratic Cost Functions).*** *The local cost functions f_i take the form*

$$f_i(x_i) = \frac{a_i}{2}(x_i - b_i)^2 - \frac{a_i b_i^2}{2} + d_i,$$

with $a_i, b_i, d_i \in \mathbb{R}$.

Note that, for any value $c_i = f_i(1) - f_i(0)$, there exists a family of coefficients a_i, b_i such that $(a_i/2)(1-b_i)^2 - (a_i/2)b_i^2 = c_i$. Further, the constant terms ensure $f_i(0) = d_i$ and $f_i(1) = c_i + d_i$. The design of a_i, b_i will be discussed in Section 4.3.

The problem we aim to solve can now be formulated as:

$$\mathscr{P}1: \min_{x\in\{0,1\}^n} f(x) = \sum_i^n f_i(x_i) + \frac{\gamma}{2}\left(p^\top x - P_\mathrm{r}\right)^2.$$

Here, $P_\mathrm{r} \in \mathbb{R}$ is a given reference value to be matched by the total output $p^\top x$ of the devices, with $p \in \mathbb{R}^n$ having entries p_i. This matching is enforced by means of a penalty term with coefficient $\gamma > 0$ in $\mathscr{P}1$. In the power systems setting, P_r can represent a real-power quantity to be approximately matched by the collective device-response. The coefficient γ and the signal P_r are determined by an Independent System Operator (ISO) and communicated to a Distributed Energy Resource Provider (DERP) that solves $\mathscr{P}1$ to obtain a real-time dispatch solution, see [1] for additional information.

The primal $\mathscr{P}1$ has an associated dual $\mathscr{D}1$ which takes the form of a semidefinite program (SDP) whose optimal value lower bounds the cost of $\mathscr{P}1$. This SDP is

$$\mathscr{D}1: \max_{\mu\in\mathbb{R}^n, \Delta\in\mathbb{R}} \quad \Delta, \tag{4.1a}$$

$$\text{subject to} \quad \begin{bmatrix} \frac{1}{2}Q(\mu) & \xi(\mu) \\ \xi(\mu)^\top & \zeta - \Delta \end{bmatrix} \succeq 0. \tag{4.1b}$$

In $\mathscr{D}1$, $Q: \mathbb{R}^n \to \mathbb{R}^{n\times n}$ and $\xi: \mathbb{R}^n \to \mathbb{R}^n$ are real-affine functions of μ and ζ is a constant. These definitions are $Q(\mu) = \left(\mathrm{diag}(a/2+\mu) + \frac{\gamma}{2}pp^\top\right), \xi(\mu) = ((a_ib_i)_i + \mu + \gamma P_\mathrm{r} p)$, and $\zeta = \sum_{i=1}^n \frac{a_i b_i^2}{2} + \frac{\gamma}{2}P_\mathrm{r}^2$. See [17] for more detail on the derivation of $\mathscr{D}1$.

4.3 Centralized Newton-like Neural Network

In this section, we develop the Centralized NEWTON-LIKE NEURAL NETWORK, or NNN-C, which is well suited for solving ¶1 in a centralized setting.

To draw analogy with the classic Hopfield Neural Network approach we will briefly introduce an auxiliary variable u_i whose relation to x_i is given by the logistic function g for each i:

$$\begin{aligned} x_i = g(u_i) &= \frac{1}{1+e^{-u_i/T}}, && u_i \in \mathbb{R}, \\ u_i = g^{-1}(x_i) &= -T\ \log\left(\frac{1}{x_i}-1\right), && x_i \in (0,1)\,, \end{aligned} \tag{4.2}$$

with temperature parameter $T > 0$.

Let $x \in (0,1)^n, u \in \mathbb{R}^n$ be vectors with entries given by x_i, u_i. To establish our algorithm, it is appropriate to first define an energy function related to $\mathscr{P}1$. Consider

$$E(x) = f(x) + \frac{1}{\tau}\sum_i \int_0^{x_i} g^{-1}(v)dv, \tag{4.3}$$

where $\tau > 0$ is a time-constant and for $z \in [0,1]$,

$$\int_0^z g^{-1}(v)dv = \begin{cases} T\left(\log(1-z) - z\log(\frac{1}{z}-1)\right), & z \in (0,1), \\ 0, & z \in \{0,1\}. \end{cases}$$

The classic HNN implements dynamics of the form $\dot{u} = -\nabla_x E(x)$, where the equivalent dynamics in x can be computed as $\dot{x} = -\nabla_x E(x)dx/du$. These dynamics can be thought of to model the interactions between neurons in a neural network or the interconnection of amplifiers in an electronic circuit, where in both cases the physical system tends toward low energy states, see [57,90]. In an optimization setting, low energy states draw analogy to low cost solutions. We now describe our modification to the classical HNN dynamics.

Recall that the domain of x is $(0,1)^n$ and our elementwise notation for log and division. We have the expressions $\nabla_x E(x) = -Wx - v - (T/\tau)\log(1/x_i - 1)_i$ and $dx/du = (x - (x_i^2)_i)/T$, where $W = -\operatorname{diag}(a) - \gamma pp^\top \in \mathbb{R}^{n\times n}$ and $v = (a_i b_i)_i + \gamma P_r p \in \mathbb{R}^n$ are defined via f. From this point forward, we work mostly in terms of x for the sake of consistency. Consider modifying the classic HNN dynamics with a PT-inverse $(|H(x)|_m)^{-1} \succ 0$ as in [85], where $H(x) = \nabla_{xx} E(x)$. The NNN-C dynamics are then given by:

$$\begin{aligned}\dot{x} &= -(|H(x)|_m)^{-1}\operatorname{diag}(\frac{dx}{du})\nabla_x E(x) \\ &= (|H(x)|_m)^{-1}\operatorname{diag}(\frac{(x_i - x_i^2)_i}{T})\left(Wx + v + \frac{T}{\tau}\log(1/x_i - 1)_i\right).\end{aligned} \tag{4.4}$$

These dynamics lend to the avoidance of saddle points of E via inclusion of the PT-inverse weighting $(|H(x)|_m)^{-1}$, in contrast to the more first-order flavor of the classic HNN dynamics. To see this, consider the eigendecomposition $H(\tilde{x}) = Q^\top \Lambda Q$ at some $\tilde{x}$ near a saddle point, i.e. $\nabla_x E(\tilde{x}) \approx 0$. If many entries of Λ are small in magnitude and remain small in the proximity of $\tilde{x}$, then the gradient is changing slowly along the "slow" manifolds associated with the eigenspace of the small eigenvalues. This is precisely what the PT-inverse is designed to combat: the weighting of the dynamics is increased along these manifolds by a factor that is inversely proportional to the magnitude of the eigenvalues. Additionally, negative eigenvalues of the Hessian are flipped in sign, which causes attractive manifolds around saddle points to become repellent.

It is desirable for E to be concave on most of its domain so the trajectories are pushed towards the feasible points of $\mathscr{P}1$; namely, the corners of the unit hypercube. To examine this, the Hessian of E can be computed as $H(x) = \frac{d^2 f}{dx^2} + \frac{1}{\tau}\operatorname{diag}(\frac{dg^{-1}(x)}{dx}) = -W + \frac{T}{\tau}\operatorname{diag}(\frac{1}{(x_i - x_i^2)_i})$. Notice that the second term is positive definite on $x \in (0,1)^n$ and promotes the convexity of E, particularly for elements x_i close to 0 or 1. For a fixed T, τ, choosing $a_i < -\gamma\|p\|^2 - 4T/\tau, \forall i$ guarantees $E(x) \prec 0$ at $x = (0.5)\mathbf{1}_n$. Generally speaking, choosing a_i to be negative and large in magnitude lends itself to concavity of E over a larger subset of its domain and to trajectories

converging closer to the set $\{0,1\}^n$. However, this comes at the expense of not exploring a rich subset of the domain. At the end of this section, we develop a Deterministic Annealing (DA) approach inspired by [84] for the online adjustment of T, τ to obtain an effective compromise between exploration of the state space and convergence to a feasible point of $\mathscr{P}1$.

We now characterize the equilibria of (4.4) for $x \in [0,1]^n$. It would appear that x with some components $x_i \in \{0,1\}$ are candidate equilibria due to the $x_i - x_i^2$ factor vanishing. However, the dynamics are not well defined here due to the log term. Additionally, note that $\lim_{x_i \to \delta} e_i^\top H(x) e_i = \infty,\ \delta \in \{0,1\}, \forall i$, where e_i is the i^{th} canonical basis vector. Due to the $\frac{T}{\tau(x_i - x_i^2)}$ term dominating W in the expression for H when x_i values are close to $\{0,1\}$, it follows that an eigenvalue of $(|H(x)|_m)^{-1}$ approaches zero as $x_i \to 0$ or 1 with corresponding eigenvector approaching $v_i = e_i$:

$$\lim_{x_i \to \delta} = v_i^\top (|H(x)|_m)^{-1} v_i = \frac{T}{\tau}(x_i - x_i^2) = 0, \quad \delta \in \{0,1\}, \forall i.$$

Using this fact, and ignoring $T, \tau > 0$, we can compute the undetermined limits in the components of $\dot{x}$ as $x_i \to \delta \in \{0,1\}$ by repeated applications of L'Hospital's rule:

$$\lim_{x_i \to \delta} \log\left(\frac{1}{x_i} - 1\right)(x_i - x_i^2)^2 = \begin{cases} 0, & \delta = 0^+, \\ 0, & \delta = 1^-. \end{cases} \tag{4.5}$$

Thus, components $x_i \in \{0,1\}$ constitute candidate equilibria. We will, however, return to the first line of (4.5) in the proof of Lemma 3 to show that they are unstable. As for components of x in the interior of the hypercube, the expression $\dot{x} = 0$ can not be solved for in closed form. However, we provide the following Lemma which shows that the set of equilibria is finite.

Lemma 2. ***(Finite Equilibria).*** *Let $\mathscr{X}$ be the set of equilibria of* (4.4) *satisfying $\dot{x} = 0$ on $x \in [0,1]^n$. The set $\mathscr{X}$ is finite.*

Proof. First consider only $\mathscr{X} \cap (0,1)^n$. Note that $(|H(x)|_m)^{-1} \succ 0$ (by construction) and

$\mathrm{diag}((x_i - x_i^2)_i/T) \succ 0$ on $x \in (0,1)^n$, so we focus on

$$Wx + \frac{T}{\tau}\log\left(1/x_i - 1\right)_i + v = \mathbf{0}_n\,. \tag{4.6}$$

Examining the above expression elementwise, it is nonconstant, continuous, and its derivative changes sign only a finite number of times. Therefore, the total number of zeros on $(0,1)^n$ must be finite.

Now consider the i^{th} element of (4.6) for $x_j \to 0$ or 1 for all j in an arbitrary permutation of $\{1,\ldots,n\} \setminus \{i\}$. Since the number of these permutations is finite, and each permutation still gives rise to a finite number of solutions to (4.6) in the i^{th} component, it follows that $\mathscr{X}$ is finite. □

To demonstrate the qualitative behavior of equilibria in a simple case, consider a one-dimensional example with $a > -\gamma p^2 - 4T/\tau$ and recall that, for $x \in (0,1)$, the sign of $-\nabla_x E(x)$ is the same as $\dot{x}$. In Figure 4.1, we observe that $-\nabla_x E(x)$ monotonically decreases in x, and a globally stable equilibrium exists in the interior $x \in (0,1)$ near $x = 0.5$. On the other hand, $a < -\gamma p^2 - 4T/\tau$ gives way to 3 isolated equilibria in the interior (one locally unstable near $x = 0.5$ and two locally stable near $x \in \{0,1\}$). This behavior extends in some sense to the higher-dimensional case. Therefore, for a scheme in which T and τ are held fixed, we prescribe $a < -\gamma\|p\|^2 - 4T/\tau$. We provide a Deterministic Annealing (DA) approach inspired by [84] for the online adjustment of T, τ in the following subsection which compromises with this strict design of a.

Finally, we establish a Lemma about the domain of the trajectories of (4.4).

Lemma 3. ***(Forward Invariance of the Open Hypercube).*** *The open hypercube* $(0,1)^n$ *is a forward-invariant set under the* NNN-C *dynamics* (4.4).

Proof. Consider again the terms of $\dot{x}$ elementwise. There are two cases to consider for evaluating x_i: $x_i = \varepsilon$ and $x_i = 1 - \varepsilon$ for some $0 < \varepsilon \ll 1$ sufficiently small such that the terms of $(|H(x)|_m)^{-1}$

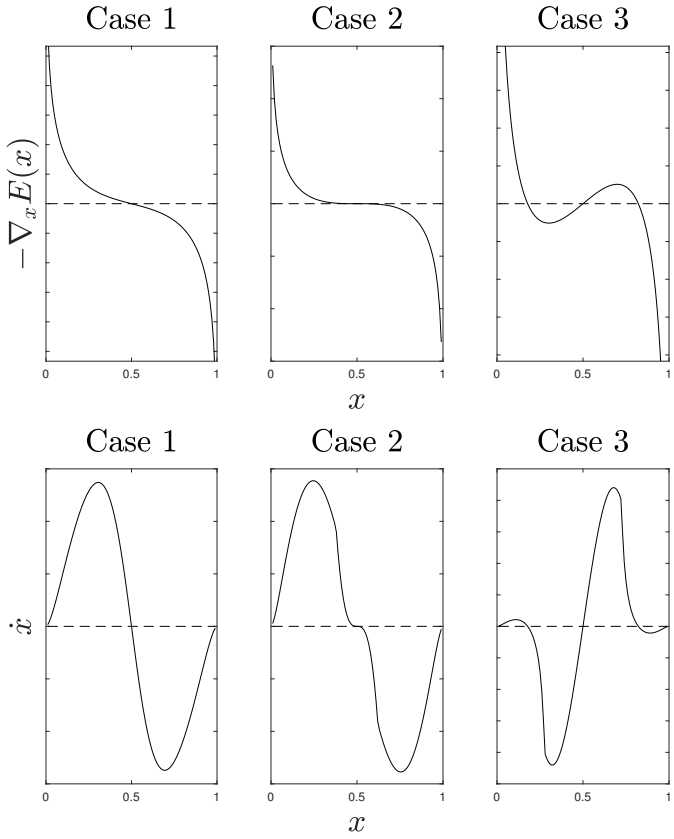


Figure 4.1. Illustration of $-\nabla_x E(x)$ (top) and $\dot{x}$ (bottom) for three instances of a. Case 1: $a > -\gamma\|p\|^2 - 4T/\tau$, Case 2: $a = -\gamma\|p\|^2 - 4T/\tau$, Case 3: $a < -\gamma\|p\|^2 - 4T/\tau$.

are still dominated by $(1/x_i - x_i^2)$ and the $Wx + v$ are still dominated by the log term. Then, consider the expression

$$\log(1/x_i - 1)(x_i - x_i^2)^2. \tag{4.7}$$

For $x_i = \varepsilon \approx 0$, (4.7) evaluates to a small positive value, and for $x_i = 1 - \varepsilon \approx 1$, (4.7) evaluates to a small negative value. We have argued that these are the dominating terms regardless of values of the remaining components of x, and so we conclude that $x_i \in \{0, 1\}$ are componentwise anti-stable and that elements of x will never approach 0 or 1. Thus, the open hypercube is forward invariant. □

Knowing that $\mathscr{P}1$ is generally NP-hard, it is unlikely that a non-brute-force algorithm exists that can converge to a global minimizer. For this reason, we aim to establish asymptotic

stability to a local minimizer of E. We first establish some assumptions.

Assumption 15. ***(Random Initial Condition).*** *The initial condition $x(0)$ is chosen randomly according to a distribution $\mathbb{P}$ that is nonzero on sets that have nonzero volume in $[0,1]^n$.*

An appropriately unbiased initial condition for our algorithm is $x(0) \approx (0.5)\,\mathbf{1}_n$, which is adequately far from the local minima located near corners of the unit cube. So, we suggest choosing a uniformly random $x(0) \in \mathscr{B}((0.5)\,\mathbf{1}_n, \varepsilon)$, where $0 < \varepsilon \ll 1$.

Assumption 16. ***(Choice of T, τ).*** *The constants $T, \tau > 0$ are each chosen randomly according to a distribution $\bar{\mathbb{P}}$ that is nonzero on sets that have nonzero volume on $\mathbb{R}_+$.*

Similarly to $x(0)$, we suggest choosing these constants uniformly randomly in a ball around some nominal T_0, τ_0, i.e. $T \in \mathscr{B}(T_0, \varepsilon), \tau \in \mathscr{B}(\tau_0, \varepsilon), 0 < \varepsilon \ll 1$. The T_0, τ_0 themselves are design parameters stemming from the neural network model, and we provide some intuition for selecting these in the simulation Section.

Now we state the main convergence result of NNN-C in Theorem 5, which states that for a random choice of T, τ, an initial condition chosen randomly from $(0,1)^n$ converges asymptotically to a local minimizer of E with probability one.

Theorem 5. ***(Convergence of* NNN-C*).*** *Given an initial condition $x(0) \in (0,1)^n$, the trajectory $x(t)$ under* NNN-C *converges asymptotically to a critical point $x^\star$ of E. In addition, under Assumption 15, on the random choice of initial conditions, and Assumption 16, on the random choice of T, τ, the probability that $x(0)$ is in the set $\bigcup_{\hat{x}} \mathscr{W}^s(\hat{x})$, where $\hat{x}$ is a saddle-point or local maximum of E, is zero.*

Proof. Let $\mathscr{X}$ be the set of all critical points of E. We first establish that E decreases along the trajectories of NNN-C and that $x(t)$ converges asymptotically to $\mathscr{X}$. Differentiating E in time,

we obtain:

$$\begin{aligned}\frac{dE}{dt} &= \dot{x}^\top \nabla_x E(x) = \dot{x}^\top \left(-Wx - v + g^{-1}(x)/\tau\right) \\ &= -\dot{x}^\top \operatorname{diag}\left(\frac{T}{(x_i - x_i^2)_i}\right) |H(x)|_m \dot{x} < 0, \quad \text{for } \dot{x} \neq 0,\ x \in (0,1)^n.\end{aligned} \tag{4.8}$$

Recall that $x(t) \in (0,1)^n$ for all $t \geq 0$ due to Lemma 3. From (4.4) and the discussion that followed on equilibria, $\dot{x} = 0$ implies $\nabla_x E(x) = 0$ due to $(|H(x)|_m)^{-1} \succ 0$ and $\operatorname{diag}((x_i - x_i^2)_i/T) \succ 0$ on $x \in (0,1)^n$. The domain of E is the compact set $[0,1]^n$ (per the definition of the integral terms), and E is continuous and bounded from below on this domain, so at least one critical point exists. Combining this basic fact with (4.8) shows that the NNN-C dynamics monotonically decrease E until reaching a critical point. More formally, applying the LaSalle Invariance Principle [62] tells us that the trajectories converge to the largest invariant set contained in the set $dE/dt = 0$. This set is $\mathscr{X}$, which is finite per Lemma 2. In this case, the LaSalle Invariance Principle additionally establishes that we converge to a single $x^\star \in \mathscr{X}$.

The proof of the second statement of the theorem relies on an application of the Stable Manifold Theorem (see [51]) as well as Lemma 2. Let $\dot{x} = \varphi_{T,\tau}(x)$ for a particular T, τ. We aim to show that $\mathbb{P}[\cup_{\hat{x}}\{\mathscr{W}_s(\hat{x}) \mid \hat{x} \text{ is a saddle or local maximum}\}] = 0$ under Assumptions 15-16. It is sufficient to show that, for each critical point $x^\star$ such that $\varphi_{T,\tau}(x^\star) = 0$, and almost all T, τ, $D\varphi_{T,\tau}(x^\star)$ is full rank and its eigenvalues have non-zero real parts. The reason for this argument is the following: let $x^\star$ be a critical point with $D\varphi_{T,\tau}(x^\star)$ full rank and eigenvalues with non-zero real parts. If the eigenvalues do *not* all have positive real parts, then some have negative real parts, which indicates that $x^\star$ is a saddle or local maximum of E. These negative real-part eigenvalues induce an unstable manifold of dimension $n_u \geq 1$. As such, the globally stable set $\mathscr{W}_s(x^\star)$ is a manifold with dimension $n - n_u < n$, and $\mathbb{P}[x(0) \in \mathscr{W}_s(x^\star)] = 0$ per Assumption 15.

To argue this case, define $h : (0,1)^n \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ as

$$h(x, T, \tau) = \det D\varphi_{T,\tau}(x).$$

We now leverage Assumption 16 and [73] to claim first that $\bar{\mathbb{P}}[h(x^\star, T, \tau) = 0] = 0$ for each $x^\star \in \mathscr{X}$, i.e. $D\varphi_{T,\tau}(x^\star)$ is full rank for each $x^\star$ with probability one w.r.t. $\bar{\mathbb{P}}$. We first address the points x for which the function h is discontinuous. Define $\hat{\mathscr{X}}$ as the set of x for which the truncation of the eigenvalues of $H(x)$ becomes active, i.e. the discontinuous points of h. Although we do not write it as such, note that H is implicitly a function of T, τ and that the eigenvalues of H can be expressed as nonconstant real-analytic functions of T, τ. Considering this fact and an arbitrary x, the set of T, τ which give $x \in \hat{\mathscr{X}}$ has measure zero with respect to $\mathbb{R}^2$ [73]. Thus, for particular T, τ, h is C^∞ almost everywhere. Applying once more the argument in [73] and Assumption 16 with the fact that h is a nonconstant real analytic function of T, τ we have that

$$\bar{\mathbb{P}}[\mathscr{T}(\hat{x}) \triangleq \{(T, \tau) \,|\, h(\hat{x}, T, \tau) = 0\}] = 0, \quad \forall \hat{x} \notin \hat{\mathscr{X}}.$$

Now consider the set of critical points as an explicit function of T, τ and write this set as $\mathscr{X}(T, \tau)$. Recalling Lemma 2, the set of $\hat{x}$ that we are interested in reduces to a finite set of critical points $x^\star \in \mathscr{X}(T, \tau)$. Thus, we can conclude that $\bar{\mathbb{P}}(\cup_{x^\star \in \mathscr{X}(T,\tau)} \mathscr{T}(x^\star)) \leq \sum_{x^\star \in \mathscr{X}(T,\tau)} \bar{\mathbb{P}}(\mathscr{T}(x^\star)) = 0$.

There is an additional case which must be considered, which is that $h(x^\star, T, \tau) \neq 0$, but some eigenvalues of $D\varphi_{T,\tau}(x^\star)$ are purely imaginary and induce stable center manifolds, which could accommodate the case of a globally stable set which is an n-dimensional manifold (i.e. the "degenerate saddle" case). We consider the function h mostly out of convenience, but the argument can be extended to a function $\mathbf{h} : (0,1)^n \times \mathbb{R} \times \mathbb{R} \to \mathbb{C}^n$ which is a map to the roots of the characteristic equation of $D\varphi_{T,\tau}(x)$. We are concerned that each element of $\mathbf{h}(x, T, \tau)$ should have a nonzero real part almost everywhere. To extend the previous case to this, consider the identification $\mathbb{C} \equiv \mathbb{R}^2$ and compose $\mathbf{h}$ with the nonconstant real analytic function $\zeta(w, z) = w$, for which the zero set is $w \equiv 0$, corresponding to the imaginary axis in our identification. From this, we obtain a nonconstant real-analytic as before whose zero set is the imaginary axis. Applying the argument in [73] in a similar way as above, $\mathbf{h}(x, T, \tau)$ has nonzero real parts for almost all

(T, τ) for each x. Therefore, the probability of a particular saddle point or local maximum $x^\star$ having a nonempty stable center manifold is zero for arbitrary $x(0)$ satisfying Assumption 15 and T, τ satisfying Assumption 16. □

We now define a Deterministic Annealing (DA) variant inspired by [84] to augment the NNN-C dynamics and provide a method for gradually learning a justifiably good feasible point of $\mathscr{P}1$. In [84], the author justifies the deterministic online tuning of a temperature parameter in the context of data clustering and shows that this avoids poor local optima by more thoroughly exploring the state space. Similarly, we aim to learn a sufficiently good solution trajectory by allowing the dynamics to explore the interior of the unit hypercube in the early stages of the algorithm, and then to force the trajectory outward to a feasible binary solution by gradually adjusting T or τ online.

Consider either reducing the temperature T or increasing the time constant τ during the execution of NNN-C. This reduces the terms in E which promote convexity, particularly near the boundaries of the unit hypercube. As T, τ are adjusted, for $a \prec -\gamma\|p\|^2$, the domain of E becomes gradually more concave away from the corners of the unit hypercube. Thus, starting with T_0/τ_0 sufficiently large, the early stages of the algorithm promote exploration of the interior of the state space. As T/τ is reduced at a rate dictated by β, the equilibria of E are pushed closer to (and eventually converge to) the feasible points of $\mathscr{P}1$. The update policy we propose is described formally in Algorithm 2, and we further explore its performance in simulation.

Algorithm 2. Determinisitc Annealing

```
1: procedure DET-ANNEAL(β > 1, T_0, τ_0, t_d)
2:     Initialize x(0)
3:     T ← T_0, τ ← τ_0
4:     while true do
5:         Implement NNN-C for t_d seconds
6:         τ ← βτ   or   T ← (1/β)T
```

Note that Algorithm 2 leads to a hybrid dynamic system with discrete jumps in an enlarged state $\phi = (x, T, \tau)$, which can cast some doubt on basic existence and uniqueness of

solutions. We refer the reader to Propositions 2.10 and 2.11 of [48] to justify existence and uniqueness of solutions in the case of $t_d > 0$ fixed.

Corollary 1. ***(Convergence to Feasible Points).*** *Under Assumptions 15-16 and $a \prec -\gamma\|p\|^2$, the* NNN-C *dynamics augmented with Algorithm 2 converge asymptotically to feasible points of $\mathscr{P}1$.*

The result of the Corollary is quickly verified by inspecting the terms of $H(x)$. The function E is smooth, strictly concave near $x = (0.5)\mathbf{1}_n$ for small T/τ due to the design of a_i, and becomes strictly convex as the elements of x approach 0 or 1, corresponding to isolated local minima of E, due to the T/τ term dominating $H(x)$. As the quantity T/τ is reduced under Algorithm 2, these local minima are shifted asymptotically closer to corners of the unit hypercube, i.e. feasible points of $\mathscr{P}1$.

4.4 Distributed Hopfield Neural Network

With the framework of the previous section we formulate a problem ¶2 which is closely related to $\mathscr{P}1$, but for which the global penalty term can be encoded by means of an auxiliary decision variable. This formulation leads to the Distributed NEWTON-LIKE NEURAL NETWORK, or NNN-D, which we rigorously analyze for its convergence properties.

It is clear from the PT-inverse operation and W being nonsparse that NNN-C is indeed centralized. In this section, we design a distributed algorithm in which each agent i must only know $p_j, j \in \mathscr{N}_i$ and the value of an auxiliary variable $y_j, j \in \mathscr{N}_i \cup \mathscr{N}_i^2$, i.e. it must have communication with its two-hop neighbor set. If two-hop communications are not directly available, the algorithm can be implemented with two communication rounds per algorithm step. We provide comments on a one-hop algorithm in Remark 6.

Assumption 17. ***(Graph Properties and Connectivity).*** *The graph $\mathscr{G} = (\mathscr{N}, \mathscr{E})$ is undirected and connected; that is, a path exists between any two pair of nodes and, equivalently, its associated Laplacian matrix $L = L^\top$ has rank $n-1$.*

Now consider the n linear equations $(p_i x_i)_i + Ly = (P_{\mathrm{r}}/n)\,\mathbf{1}_n$. Notice that, by multiplying from the left by $\mathbf{1}_n^\top$ and applying $\mathbf{1}_n^\top L = \mathbf{0}_n^\top$, we recover $p^\top x = P_{\mathrm{r}}$. Thus, by augmenting the state with an additional variable $y \in \mathbb{R}^n$, we can impose a distributed penalty term. We now formally state the distributed reformulation of $\mathscr{P}1$:

$$\mathscr{P}2: \min_{x\in\{0,1\}^n, y\in\mathbb{R}^n} \tilde{f}(x,y) = \sum_i^n f_i(x_i) + \frac{\gamma}{2}\sigma^\top\sigma,$$

where the costs f_i again satisfy $f_i(1) - f_i(0) = c_i$ and we have defined $\sigma = (p_i x_i)_i + Ly - (P_{\mathrm{r}}/n)\,\mathbf{1}_n$ for notational simplicity. Before proceeding, we provide some context on the relationship between $\mathscr{P}1$ and $\mathscr{P}2$.

Lemma 4. ***(Equivalence of P1 and P2).*** *Let Assumption 17, on graph connectivity, hold, and let $(x^\star, y^\star)$ be a solution to $\mathscr{P}2$. Then, $x^\star$ is a solution to $\mathscr{P}1$ and $f(x^\star) = \tilde{f}(x^\star, y^\star)$.*

Proof. The equivalence stems from the global term and the flexibility in the unconstrained y variable. Notice

$$\begin{aligned}\frac{\gamma}{2}\sigma^\top\sigma &= \frac{\gamma}{2}\sigma^\top(I_n - \mathbf{1}_n\mathbf{1}_n^\top/n)\sigma + \frac{\gamma}{2}\sigma(\mathbf{1}_n\mathbf{1}_n^\top/n)\sigma \\ &= \frac{\gamma}{2}\sigma^\top(I_n - \mathbf{1}_n\mathbf{1}_n^\top/n)\sigma + \frac{\gamma}{2}(p^\top x - P_{\mathrm{r}})^2.\end{aligned}$$

We have recovered the original global term of $\mathscr{P}1$ in the bottom line, so now we deal with the remaining term. The matrix $I_n - \mathbf{1}_n\mathbf{1}_n^\top/n \succeq 0$ has $\operatorname{image} I_n - \mathbf{1}_n\mathbf{1}_n^\top/n = \operatorname{span}\{\mathbf{1}_n\}^\perp = \operatorname{image} L$, given that L is connected. Thus, because y is unconstrained and does not enter the cost anywhere else, we can compute the set of possible minimizers of $\tilde{f}$ in closed form with respect to any x as

$$\begin{aligned}y^\star &\in \{-L^\dagger\big((p_i x_i)_i - (P_{\mathrm{r}}/n)\,\mathbf{1}_n\big) + \theta\,\mathbf{1}_n \mid \theta \in \mathbb{R}\} \\ &= \{-L^\dagger(p_i x_i)_i + \theta\,\mathbf{1}_n \mid \theta \in \mathbb{R}\}.\end{aligned}$$

Moreover, substituting a $y^\star$ gives $\sigma \in \operatorname{span}\{\mathbf{1}_n\}$, and it follows that the problem $\mathscr{P}2$ reduces precisely to $\mathscr{P}1$. □

To define NNN-D, we augment the centralized NNN-C with gradient-descent dynamics

in y on a newly obtained energy function $\widetilde{E}$ of $\mathscr{P}2$. Define $\widetilde{E}$ as

$$\widetilde{E}(x,y) = \tilde{f}(x,y) + \frac{1}{\tau}\sum_i \int_0^{x_i} g^{-1}(\nu)d\nu. \tag{4.9}$$

In Section 4.3, we obtained a matrix W which was nonsparse. Define $\widetilde{W}, \tilde{v}$ for $\widetilde{E}$ via $\tilde{f}$ as $\widetilde{W} = -\operatorname{diag}(a+\gamma(p_i^2)_i), \tilde{v} = (a_i b_i)_i + \gamma \operatorname{diag}(p)\left((P_{\mathrm{r}}/n)\,\mathbf{1}_n - Ly\right)$. Compute the Hessian of $\widetilde{E}$ with respect to only x as $\widetilde{H}(x) = \nabla_{xx}\widetilde{E}(x,y) = -\widetilde{W} + (T/\tau)\operatorname{diag}(1/x - (x_i^2)_i)$. Since $\widetilde{H}(x)$ is diagonal, the ii^{th} element of the PT-inverse of $\widetilde{H}(x)$ can be computed locally by each agent i as:

$$(|\widetilde{H}(x)|_m)^{-1}_{ii} = \begin{cases} |\widetilde{H}(x)_{ii}|^{-1}, & |\widetilde{H}(x)_{ii}| \geq m, \\ 1/m, & \text{o.w.} \end{cases}$$

where $\widetilde{H}(x)_{ii} = a_i + \gamma p_i^2 + T/\tau(x_i - x_i^2)^{-1}$. The NNN-D dynamics, which are PT-Newton descent in x and gradient descent in y on $\widetilde{E}$, are then stated as:

$$\begin{aligned} \dot{x} &= (|\widetilde{H}(x)|_m)^{-1} \operatorname{diag}\left(\frac{(x_i - x_i^2)_i}{T}\right)\left(\widetilde{W}x + \frac{T}{\tau}\log\left(1/x_i - 1\right)_i + \tilde{v}\right), \\ \dot{y} &= -\alpha\gamma L\left((p_i x_i)_i + Ly\right), \end{aligned} \tag{4.10}$$

where $\alpha = \operatorname{diag}(\alpha_i)$ is a diagonal matrix of arbitrary positive gains $\alpha_i > 0$. Due to the new matrices $\widetilde{W}, \tilde{v}$ and the sparsity of L, $\dot{x}$ can be computed with one-hop information and $\dot{y}$ with two-hop information (note the L^2 term); thus, (4.10) defines a distributed algorithm. Additionally, recalling the discussion on parameter design, the problem data a and b can now be locally designed.

Before proceeding, we establish a property of the domain of y and some distributed extensions of Lemmas 2 and 3.

Lemma 5. ***(Domain of Auxiliary Variable).*** *Given an initial condition $y(0)$ with $\mathbf{1}_n^\top y(0) = \kappa$,*

the trjaectory $y(t)$ is contained in the set

$$\mathscr{Y} = \{\omega + (\kappa/n)\,\mathbf{1}_n \mid \mathbf{1}_n^\top \omega = 0\}. \tag{4.11}$$

Proof. The proof is trivially seen by multiplying $\dot{y}$ in (4.10) from the left by $\mathbf{1}_n$ and applying the null space of L. □

Lemma 6. ***(Closed Form Auxiliary Solution).*** *For an arbitrary fixed $x \in [0,1]^n$, the unique minimizer $y^\star$ contained in $\mathscr{Y}$ of both $\tilde{f}$ and $\widetilde{E}$ is given by*

$$y^\star = -L^\dagger \left(p_i \tilde{x}_i\right)_i + \frac{\kappa}{n}\mathbf{1}_n\,. \tag{4.12}$$

This is also the unique equilibrium of (4.10) *in $\mathscr{Y}$.*

Proof. The first term is computed by setting $\nabla_y \tilde{f}(x,y^\star) = \mathbf{0}_n$ (resp. $\nabla_y \widetilde{E}(x,y^\star) = \mathbf{0}_n$) and solving for $y^\star$. There is a hyperplane of possible solutions due to the rank deficiency of L, but we are looking for the unique solution in $\mathscr{Y}$. The second term therefore follows from (4.11). The fact that this point is also the unique equilibrium in $\mathscr{Y}$ follows from the fact that $\dot{y} = -\alpha \nabla_y \widetilde{E}(x,y^\star)$. □

Lemma 7. ***(Finite Equilibria (Distributed)).*** *Let $\widetilde{\mathscr{X}} \times \widetilde{\mathscr{Y}}$ be the set of equilibria of* (4.10) *satisfying $(\dot{x},\dot{y}) = 0$ on $(x,y) \in [0,1]^n \times \mathscr{Y}$. The set $\widetilde{\mathscr{X}} \times \widetilde{\mathscr{Y}}$ is finite.*

Proof. The proof follows closely to the proof of Lemma 2 with the variation that $\tilde{v}$ in the expression for $\dot{x}$ is now a function of y. Given the result of Lemma 6, we may directly substitute the unique $y^\star$ (4.12) for any x. Because $y^\star$ is simply a linear expression in x, the same argument as in Lemma 2 that $\widetilde{\mathscr{X}}$ is finite follows. □

We now extend the results of Theorem 5 to the distributed case of solving $\mathscr{P}2$ via NNN-D. We have the following theorem on the trajectories of $(x(t),y(t))$ under (4.10), which can be interpretted as establishing convergence to a local minimizer with probability one.

Theorem 6. ***(Convergence of* NNN-D*).*** *Given an initial condition* $(x(0), y(0)) \in (0,1)^n \times \mathbb{R}^n$*, the trajectory* $(x(t), y(t))$ *under* NNN-D *converges asymptotically to a critical point* $(x^\star, y^\star)$ *of* $\widetilde{E}$*. In addition, under Assumption 15, on the random choice of initial condition* $x(0)$*, and Assumption 16, on the random choice of* T, τ*, the probability that* $(x(0), y(0))$ *is in the set* $\bigcup_{\hat{x},\hat{y}} \mathscr{W}^s(\hat{x},\hat{y})$*, where* $(\hat{x},\hat{y})$ *is a saddle-point or local maximum of* $\widetilde{E}$*, is zero. Lastly, all local minima* $(x^\star, y^\star)$ *of* $\widetilde{E}$ *are globally optimal in* y*:* $\widetilde{E}(x^\star, y) \geq \widetilde{E}(x^\star, y^\star), \forall y \in \mathbb{R}^n$*.*

Proof. The first part of the proof to establish convergence to a critical point follows from a similar argument to the proof of Theorem 5. Differentiating $\widetilde{E}$ with respect to time gives:

$$\begin{aligned} \frac{d\widetilde{E}}{dt} &= \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}^\top \begin{bmatrix} \nabla_x \widetilde{E}(x,y) \\ \nabla_y \widetilde{E}(x,y) \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}^\top \begin{bmatrix} -\widetilde{W}x - \tilde{v} + g^{-1}(x)/\tau \\ -\alpha^{-1}\dot{y} \end{bmatrix} \\ &= -\dot{x}^\top \operatorname{diag}(T/(x_i - x_i^2)_i)|\tilde{H}(x)|_m \dot{x} - \alpha^{-1}\dot{y}^\top \dot{y} < 0, \\ &\qquad \dot{x} \neq 0 \text{ or } \dot{y} \neq 0, \quad (x,y) \in (0,1)^n \times \mathscr{Y}. \end{aligned} \tag{4.13}$$

Thus, $\widetilde{E}$ monotonically decreases along the trajectories of NNN-D. Given (4.13), we call again on the forward invariance property of the open hypercube for the distributed case via Lemma 8, stated below, which verifies that $(x,y) \in (0,1)^n \times \mathscr{Y}$ at all times.

Due to the deficiency induced by L, $\widetilde{E}$ is not radially unbounded in y over all of $\mathbb{R}^n$, so we must be careful before applying the LaSalle Invariance Principle. Instead, define $\widetilde{E}$ only on $[0,1]^n \times \mathscr{Y}$ in consideration of Lemma 5. Radial unboundedness in $\widetilde{E}$ is then obtained given any $y(0)$, and it follows that the trajectories converge to largest invariant set contained in $d\widetilde{E}/dt = 0$ per the LaSalle Invariance Principle [62]. This is the finite set of critical points of $\widetilde{E}$ per Lemma 7, and so it additionally follows that we converge to a single critical point $(x^\star, y^\star)$.

Because $\widetilde{E}$ is convex in y, it follows that for any fixed x there exist only local minima of $\widetilde{E}$ with respect to y. In consideration of this, we need only apply the Stable Manifold Theorem [51] to x. The argument for this develops similarly to the proof of Theorem 5, and we conclude that the trajectories of NNN-D converge to a local minimizer $(x^\star, y^\star)$ of $\widetilde{E}$ with probability one.

The final part of the Theorem statement that $\widetilde{E}(x^\star, y) \geq \widetilde{E}(x^\star, y^\star), \forall y \in \mathbb{R}^n$ can also be seen from the convexity of $\widetilde{E}$ in y and applying the first-order condition of convexity:

$$\widetilde{E}(x^\star, y) \geq \widetilde{E}(x^\star, y^\star) + (y - y^\star)^\top \nabla_y \widetilde{E}(x^\star, y^\star)$$

along with $\nabla_y \widetilde{E}(x^\star, y^\star) = \mathbf{0}_n$. □

Lemma 8. ***(Forward Invariance of the Open Hypercube (Distributed)).*** *The set* $(0,1)^n \times \mathscr{Y}$ *is a forward-invariant set under the* NNN-D *dynamics* (4.10)*.*

Proof. The forward invariance of $\mathscr{Y}$ is already established per its definition and Lemma 5, but we must establish that the trajectories $y(t)$ remain bounded in order to apply the argument in Lemma 3 to the proof of Theorem 6. Compute the Hessian of $\widetilde{E}$ with respect to y as:

$$\nabla_{yy} \widetilde{E} = \gamma L^2 \succeq 0.$$

Due to the connectedness of L, the eigenspace associated with the $n-1$ strictly positive eigenvalues of γL^2 is parallel to $\mathscr{Y}$. Therefore, $\widetilde{E}$ is strictly convex in y on this subspace, and it follows that $\widetilde{E}$ is bounded from below on $\mathscr{Y}$. Due to $d\widetilde{E}/dt \leq 0$ (4.13) and the continuity of $\widetilde{E}$ in y, it follows that $y(t)$ is bounded for all t. Given this, the argument from Lemma 3 applies to the trajectories $x(t)$, and the set $(0,1)^n \times \mathscr{Y}$ is forward invariant under NNN-D (4.10). □

Remark 6. ***(One-Hop Distributed Algorithm).*** The proposed distributed algorithm requires two-hop neighbor information, which may be intractable in some settings. The source of the two-hop term stems from the quadratic γ penalty term. However, it is possible to define a one-hop distributed algorithm via a Lagrangian-relaxation route.

Consider posing $\mathscr{P}2$ with the γ term instead as a linear constraint: $\sqrt{\gamma/2}((p_i x_i)_i + Ly) = \sqrt{\gamma/2}(P_{\mathrm{r}}/n)\,\mathbf{1}_n$. Applying Lagrangian relaxation to this problem introduces a Lagrange multiplier on the linear terms, and from there it would be appropriate to define a saddle-point-like algorithm

along the lines of [34] in which gradient-ascent in the dual variable is performed. This changes the nature of the penalty from squared to linear, so the underlying optimization model is different in that sense, but it follows that this approach could be implemented with one-hop information.

We note that, in some distributed contexts, penalty terms or constraints can be imposed via $\sqrt{L}$ which then appears as L in the associated squared terms of the dynamics (in place of L^2). However, the linear L also appears in our algorithm, and substituting $\sqrt{L}$ would not inherit the sparsity of the communication graph. Therefore we leave the design of a fully one-hop mixed first-order/second-order algorithm as an open problem.

4.5 Simulations

Our simulation study is split in to two parts; the first focuses on numerical comparisons related to runtime and solution quality, and the second is a 2D visualization of the trajectories of the Distributed Annealing (DA) variants for both the centralized and distributed NNN methods.

4.5.1 Runtime and Solution Quality Comparison

In this section, we compare to a greedy method stated as Algorithm 3 and a semidefinite programming (SDP) relaxation method stated as Algorithm 4. In short, the greedy method initializes the state as $x = \mathbf{0}_n$ and iteratively sets the element x_i to one which decreases the cost function the most. This is repeated until no element remains for which the updated state has lower cost than the current state. For the SDP method, a convex SDP is obtained as the relaxation of $\mathscr{P}1$, see e.g. [100]. We use the shorthand $\texttt{SDPrlx}(\bullet)$ to indicate this in the statement of Algorithm 4. This SDP is solved using CVX software in MATLAB [50] and a lowest-cost partition is computed to construct a feasible solution. For the sake of convenience in stating both algorithms, we have defined $f' : 2^n \to \mathbb{R}$ to be the set function equivalent of f, i.e. the cost of $\mathscr{P}1$. That is, $f'(\mathscr{S}) = f(x)$, where $i \in \mathscr{S}$ indicates $x_i = 1$ and $i \notin \mathscr{S}$ indicates $x_i = 0$. Finally, we additionally compare to a brute force method which we have manually programmed as an

exhaustive search over the entire (finite) feasibility set.

Algorithm 3. Greedy Method

procedure GREEDY(f')
  $\mathscr{S} \leftarrow \emptyset$
  `done` $\leftarrow$ `false`
  while `done` $=$ `false` **do**
    $i^\star \leftarrow \underset{i \notin \mathscr{S}}{\operatorname{argmin}} f'(\mathscr{S} \cup \{i\})$
    if $f'(\mathscr{S} \cup \{i^\star\}) < f'(\mathscr{S})$ **then**
      $\mathscr{S} \leftarrow \mathscr{S} \cup \{i^\star\}$
    else
      `done` $\leftarrow$ `true`
  $x_i \leftarrow \begin{cases} 0, & i \notin \mathscr{S}, \\ 1, & i \in \mathscr{S}. \end{cases}$
  return x

Algorithm 4. SDP Relaxation Method

procedure SDP(f')
  $¶_{\text{SDP}} \leftarrow \texttt{SDPrlx}(\mathscr{P}1)$
  $x^\star \leftarrow \underset{x}{\operatorname{argmin}} ¶_{\text{SDP}}$
  $\mathscr{S} \leftarrow \emptyset$
  `done` $\leftarrow$ `false`
  while `done` $=$ `false` **do**
    $i^\star \leftarrow \underset{i \notin \mathscr{S}}{\operatorname{argmax}} x_i$
    if $f'(\mathscr{S} \cup \{i^\star\}) < f'(\mathscr{S})$ **then**
      $\mathscr{S} \leftarrow \mathscr{S} \cup \{i^\star\}$
    else
      `done` $\leftarrow$ `true`
  $x_i \leftarrow \begin{cases} 0, & i \notin \mathscr{S}, \\ 1, & i \in \mathscr{S}. \end{cases}$
  return x

In Figure 4.2 we plot the runtime in MATLAB on a 3.5GHz Intel Xeon E3-1245 processor over increasing problem size n for each of six methods: a brute force search, the aforementioned greedy and SDP methods, the HNN first proposed in [57] (i.e. the gradient-like version of NNN-C), and the NNN-C and NNN-D methods we developed in Sections 4.3 and 4.4. The first obvious observation to make is that the runtime of brute force method increases at a steep

exponential rate with increasing n and exceeds 120 seconds at $n = 22$, making it intractable for even medium sized problems. Next, we note that there are some spikes associated with the HNN method around $n = 25$ to $n = 40$. These are reproducible, and we suspect that this is due to the emergence of saddle-points and increasing likelihood of encountering these along the trajectory as n increases. This is a well-documented problem observed in literature, see e.g. [36], and we also confirm it empirically in this setting by observing that share of iterations for which the Hessian is indefinite (as opposed to positive definite) tends to grow as n increases. We also note that NNN-C scales relatively poorly, which can be attributed to a matrix eigendecomposition being performed at each discretized iteration of the continuous-time algorithm. For NNN-D, the matrix being eigendecomposed is diagonal, which makes it a trivial operation and allows NNN-D to scale well. We note that the SDP method scales the worst amongst the non brute-force methods. Unsurprisingly, the greedy method remains the fastest at large scale, although recall that the motivation of developing our method is for it to be distributed and that a greedy approach can not be distributed due to the global penalty term.

As for algorithm performance as it pertains to the cost of the obtained solution, we fix $n = 50$ and additionally include DA variants of both NNN-C and NNN-D. We also omit the brute force method due to intractability. For the sake of comparison, we compute a performance metric Q and provide it for each method in Table 4.1. The metric Q is computed as follows: for each trial, sort the methods by solution cost. Assign a value of 6 for the best method, 5 for the second-best, and so on, down to the seventh-best (worst) receiving zero. Add up these scores for all 100 trials, and then normalize by a factor of 600 (the maximum possible score) to obtain Q. Note that Q does not account for runtime in any way.

It should be unsurprising that the tried-and-true centralized greedy and SDP methods perform the best. However, we note that they were beaten by our methods in a significant number of trials, which can be seen by noting that a Q score for two methods which perform best or second-best in all trials would sum to $1100/600 = 1.83$, while $Q(\text{greedy}) + Q(\text{SDP}) = 1.75$, or a cumulative pre-scaled score of 1050, indicating that our methods outperformed these methods

in net 50 "placement spots" over the 100 trials. In general, we find that the DA version of the NNN algorithms obtains better solutions than the non-DA version, confirming the benefit of this approach. We also find that NNN-D generally outperforms NNN-C. It's possible that an initially "selfish" trajectory in x is beneficial, which would neglect the global penalty until y adequately converges, although this is speculative. Lastly, we note that the HNN method never performs better than worst, which we attribute to the steepest-descent nature of gradient algorithms which do not use curveature information of the energy function. It might be possible that the stopping criterion forces HNN to terminate near saddle-points, although we do not suspect this since we observe the Hessian is positive-definite in the majority of termination instances.

As for parameter selection, we find that choosing $m \ll 1$ is generally best, since $m \geq 1$ would always produce a PT-inverse Hessian with eigenvalues contained in $(0, 1]$. This effectively scales down $\dot{x}$ in the eigenspace associated with Hessian eigenvalue magnitudes greater than 1, but does not correspondingly scale up $\dot{x}$ in the complementary eigenspace associated with small eigenvalues. Additionally, choosing T/τ greater than 1 in the fixed case tended to be effective. This may be related to selecting $a_i < -\gamma\|p\|^2 - 4T/\tau$ to guarantee anti-stability from $(0.5)\,\mathbf{1}_n$, and would explain why a high T_0/τ_0 that decreases in the DA learning variant performs so well. In general, for the DA learning variant, we recommend choosing T_0, τ_0 so that $T_0/\tau_0 \gg 1$ and also $\beta > 1$ sufficiently large so that $T/\tau \ll 1$ by algorithm termination, which gives rise to a robust exploration/exploitation tradeoff. Finally, all $\alpha \approx 1$ seem to behave roughly the same, with only $\alpha \ll 1$ and $\alpha \gg 1$ behaving poorly (the former leading to slow convergence in y and "selfish" behavior in x, and the latter being destabilizing in the discretization of $\dot{y}$).

4.5.2 Learning Steps and 2-D Trajectories

Next, for the sake of understanding how the learning rate T/τ affects the trajectories of the solutions, we have provided Figure 4.3 which plots the 2-D trajectories of NNN-C and NNN-D with T/τ being gradually reduced over 15 learning steps. The contours of the energy

Table 4.1. Comparison of performance metric Q for 100 randomized trials with $n = 50$.

Method	Q
NNN-C	0.2161
NNN-C-DA	0.2891
NNN-D	0.5443
NNN-D-DA	0.7005
HNN	0
Greedy	0.8411
SDP	0.9089

Table 4.2. Problem data and parameter choices (where relevant) for performance comparison. Problem data p_i, c_i is generated randomly from given distributions for each of 100 trials.

Data or parameter	Value
n	50
p_i	$\mathscr{U}[1,50]$
c_i	p_i^e, $e \sim \mathscr{U}[2,3]$
P_r	1500
γ	1
T_0	1
τ_0	0.1
m	0.1
α	1
Learning steps	10
β	1.4
n	50

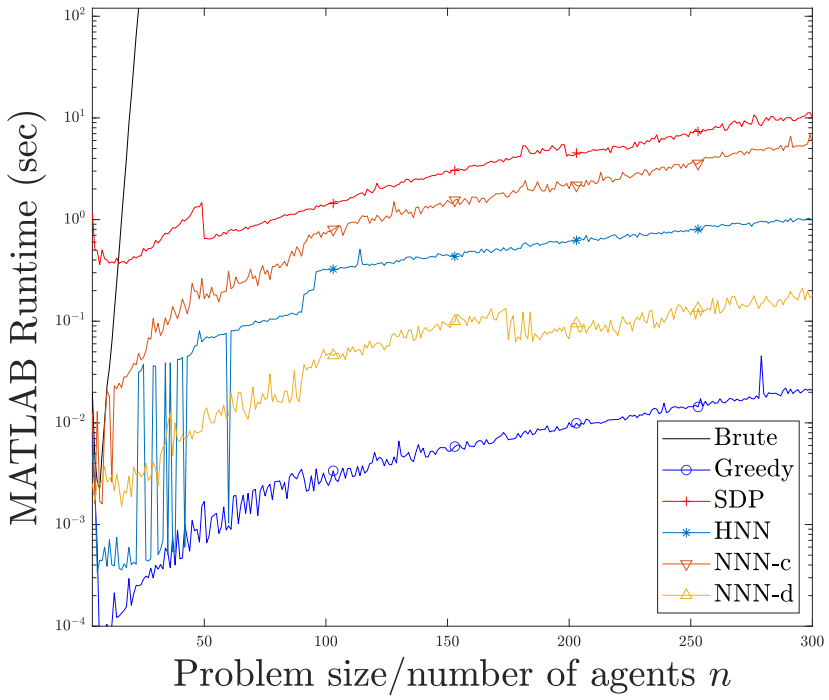


Figure 4.2. Runtime of each method for increasing problem sizes.

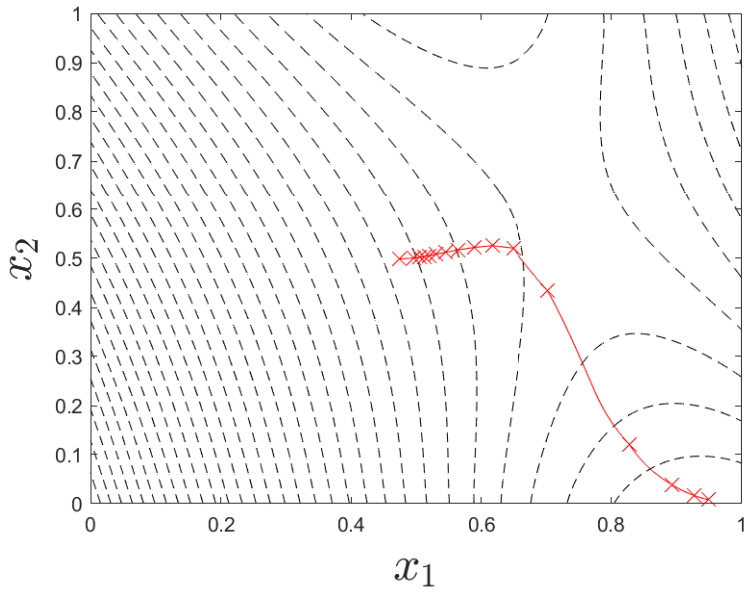
function for the final step are also plotted. The problem data and choice for a is:

$$c = (2,1)^\top, \quad p = (3,1)^\top, \quad P_\mathrm{r} = 2.8, \quad \gamma = 4, \quad a = -(10,10)^\top.$$

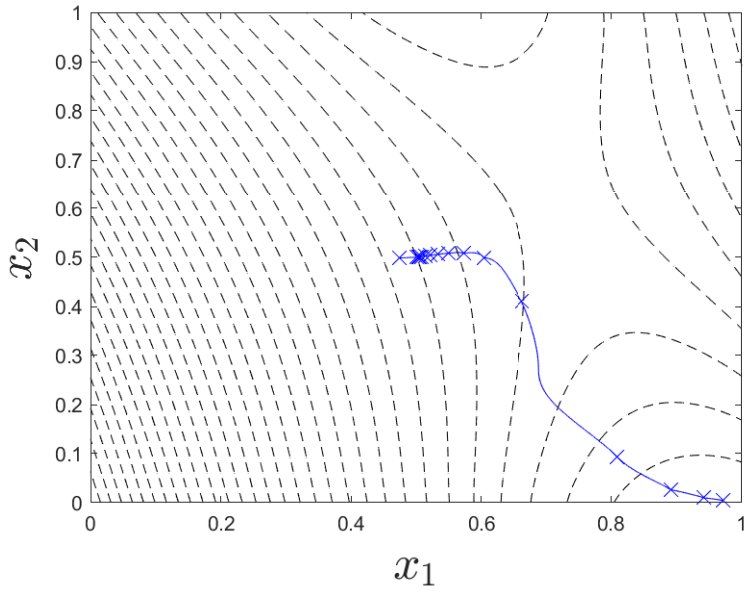
Note that, in each case, the trajectory approaches the optimal solution $x^\star = (1,0)^\top$. However, it is worth noting that a steep saddle point occurs around $x = (0.75,0.6)^\top$. Intuitively, this corresponds to a high risk of the trajectory veering away from the optimal solution had the DA not been implemented. With the opportunity to gradually learn the curveature of the energy function, as shown by stabilization to successive equilibria marked by $\times$, each algorithm is given the opportunity to richly explore the state space before stabilizing to the optimal solution $(1,0)^\top$. Further studying the learning-rate T/τ and a more complete analysis of Algorithm 2 and the parameter β are subjects of future work.

Acknowledgements

The material in this chapter, in full, is provisionally accepted in Automatica. It is expected to appear as *Distributed Resource Allocation with Binary Decisions via Newton-like Neural Network Dynamics*, T. Anderson and S. Martínez. The dissertation author was the primary investigator and author of this paper.



(a)



(b)

Figure 4.3. Centralized NNN-C (a) and distributed NNN-D (b) trajectories in 2D with 15 learning steps. Stable equilibrium points between learning steps indicated by $\times$, contours of E and $\widetilde{E}$ in final step indicated by dashed lines.

Chapter 5

Maximizing Algebraic Connectivity of Constrained Graphs in Adversarial Environments

This chapter aims to maximize algebraic connectivity of networks via topology design under the presence of constraints and an adversary. We are concerned with three problems. First, we formulate the concave-maximization topology design problem of adding edges to an initial graph, which introduces a nonconvex binary decision variable, in addition to subjugation to general convex constraints on the feasible edge set. Unlike previous methods, our method is justifiably not greedy and capable of accommodating these additional constraints. We also study a scenario in which a coordinator must selectively protect edges of the network from a chance of failure due to a physical disturbance or adversarial attack. The coordinator needs to strategically respond to the adversary's action without presupposed knowledge of adversary's feasible attack actions. We propose three heuristic algorithms for the coordinator to accomplish the objective and identify worst-case preventive solutions. Each algorithm is shown to be effective in simulation and we provide some discussion on their compared performance.

5.1 Bibliographical Comments

The classic paper [42] by Miroslav Fiedler proposes a scalar metric for the *algebraic connectivity* of undirected graphs, which is given by the second-smallest eigenvalue of the graph

Laplacian and is also referred to as the *Fiedler* eigenvalue. One of the main problems we are interested in studying is posed in [45], where the authors develop a heuristic for strategically adding edges to an initial topology to maximize this eigenvalue. Lower and upper bounds for the Fiedler eigenvalue with respect to adding a particular edge are found, however, the work is limited in that their approach is greedy and may not perform well in some cases. In addition, the proposed strategy does not address how to handle additional constraints that may be imposed on the network, such as limits on nodal degree or restricting costlier edges. The authors of [12] aim to solve the problem of maximizing connectivity for a particular robotic network scenario in the presence of an adversarial jammer, although the work does not sufficiently address scenarios with a more general adversary who may not be subject to dynamical constraints. The Fiedler eigenvector, which has a close relationship to the topology design problem, is studied in [72]. Many methods to compute this eigenvector exist, such as the cascadic method in [98]. However, these papers do not fully characterize how this eigenvector evolves from adding or removing edges from the network, which is largely unanswered by the literature. The authors of [38] study the spectra of randomized graph Laplacians, and [108] gives a means to estimate and maximize the Fiedler eigenvalue in a mobile sensor network setting. However, neither of these works consider the problem from a design perspective. In the celebrated paper by Goemans and Williamson [49], the authors develop a relaxation and performance guarantee on solving the MAXCUT problem, which has not yet been adapted for solving the topology design problem. Each of [15, 37] survey existing results related to the Fiedler eigenvalue and contain useful references.

Statement of Contributions

This chapter considers three optimization problems and has two main contributions. First, we formulate the concave-maximization topology design problem from the perspective of adding edges to an initial network, subject to general convex constraints plus an intrinsic binary

constraint. We then pose a scenario where a coordinator must strategically select links to protect from random failures due to a physical disturbance or malicious attack by a strategic adversary. In addition, we formulate this problem from the adversary's perspective. Our first main contribution is a method to solve the topology design problem (and, by extension, the protected links problem). We develop a novel MAXCUT-inspired SDP relaxation to handle the binary constraint, which elegantly considers the whole problem in a manner where previous greedy methods fall short. Our next main contribution returns to the coordinator-adversary scenario. We first discuss the nonexistence of a Nash equilibrium in general. This motivates the development of an optimal *preventive* strategy in which the coordinator makes an optimal play with respect to any possible response by the adversary. We rigorously prove several auxiliary results about the solutions of the adversary's computationally hard concave-minimization problem in order to justify heuristic algorithms which may be used by the coordinator to search for the optimal preventive strategy. A desirable quality of these algorithms is they do not presuppose the knowledge of the adversary's feasibility set, nor the capability of solving her problem. Rather, the latter two algorithms observe her plays over time and use these against her construct an effective preventive solution. Simulation studies demonstrate the effectiveness of our SDP relaxation for topology design and the performance of the preventive-solution seeking algorithms when applied to the related adversarial link-protection problem.

5.2 Problem Statements

This section formulates the three optimization problems that we study. The first is the problem of adding edges to an initial topology to maximize algebraic connectivity of the final graph. The second problem introduces a coordinator who is charged with protecting some links in a network that are subject to an external disturbance or attack from the perspective of maximizing connectivity. The third problem takes the opposite approach of the second problem by aiming to minimize connectivity from an adversarial point of view.

5.2.1 Topology Design for Adding Edges

Consider a network of agents with some initial (possibly disconnected) graph topology characterized by an edge set $\mathscr{E}_0$ and Laplacian L_0. We would like to add k edges to $\mathscr{E}_0$, possibly subject to some additional convex constraints, so as to maximize the Fiedler eigenvalue of the final Laplacian $L^\star$. This problem is well motivated: the Fiedler eigenvalue dictates convergence rate of many first-order distributed algorithms, such as consensus and gradient descent. First, let $\mathscr{E}$ be the complete edge set (not including redundant edges or self loops) with $m = |\mathscr{E}|$. Consider the incidence matrix E associated with $\mathscr{E}$ and the vector of edge connectivities x, as described in Section **??**. The constrained topology design problem is then formulated as

$$\begin{aligned}
\mathscr{P}1 : \max_{x,\alpha} \quad & \alpha, && \text{(5.1a)}\\
\text{subject to} \quad & E(\operatorname{diag}(x))E^\top \succeq \alpha \mathbb{I}_n, && \text{(5.1b)}\\
& \sum_{l=1}^{m} x_l \leq k + |\mathscr{E}_0|, && \text{(5.1c)}\\
& x \in \mathscr{X}, && \text{(5.1d)}\\
& x_l = 1,\ l \sim (i,j) \in \mathscr{E}_0, && \text{(5.1e)}\\
& x_l \in \{0,1\},\ l \in \{1,\ldots,m\}. && \text{(5.1f)}
\end{aligned}$$

In $\mathscr{P}1$, the solution $\alpha^\star$ is precisely the value for λ_2 of the final Laplacian solution given by $L(x^\star) = L_0 + E(\operatorname{diag}(x^\star))E^\top$. This is encoded in the constraint (5.1b), where the pseudo-identity matrix $\mathbb{I}_n$ has the effect of filtering out the fixed zero-eigenvalue of the Laplacian. A useful relation is $\lambda_2 = \inf_z \{z^\top L(x) z \mid z \perp \mathbf{1}_n, \|z\|_2 = 1\}$, which shows that λ_2 as a function of x is a pointwise infimum of linear functions of x and is therefore concave. By extension, $\mathscr{P}1$ is a concave-maximization problem in x. The set $\mathscr{X}$ is assumed compact and convex and may be chosen by the designer in accordance with problem constraints such as bandwidth/memory limitations, restrictions on nodal degrees, or restricting certain edges from being chosen. These constraints

may manifest in applications such as communication bandwidth limitations amongst Distributed Energy Resource Providers for Real-Time Optimization in renewable energy dispatch [20]. The binary constraint (5.1f) is nonconvex and makes the problem a NP-hard. Handling this constraint is one of the main objectives of this chapter and will be addressed in Section 5.3.

As for existing methods of solving $\mathscr{P}1$, one option is solve it over the convex hull of the constraint set, which is given by $[0,1]^m \cap \mathscr{X}$. Then, the problem may be solved in k steps by iteratively adding the edge $l \sim (i,j)$ for which x_l is maximized in each step. This is discussed in [45] and the references therein. Although this method allows the designer to easily capture $\mathscr{X}$, it is not a satisfying relaxation because the underlying characteristics of the connectivity are not well captured. The authors of [45] propose an alternate method which chooses the edge l for which $\dfrac{\partial \lambda_2}{\partial x_l} = v_2^\top \dfrac{\partial L(x)}{\partial x_l} v_2 = v_2^\top e_l e_l^\top v_2 = (v_{2,i} - v_{2,j})^2$ is maximal. This method is limited in that it is (a) greedy and (b) cannot account for $\mathscr{X}$. We are motivated to develop a relaxation for $\mathscr{P}1$ which improves on existing techniques in both performance and the capability of handling constraints.

5.2.2 Topology Design for Protecting Edges

We now formulate a problem which is closely related to $\mathscr{P}1$, but which is interesting to study in its own right for a variety of reasons. Motivated by the possibility of guarding against disruptive physical disturbances or adversarial attacks, consider a coordinator who may protect up to k_s links from failing in a network. The failure of the links are assumed to be independent Bernoulli random variables whose probabilities are encoded by the vector $p \in [0,1]^m$. Then, consider the coordinator's decision vector $s \in \mathsf{S} = \{0,1\} \cap \mathsf{S}'$, where S' is assumed convex. We write out the Laplacian as before, $L(x) = E \operatorname{diag}(x) E^\top$. Following a disturbance or attack, the probability that an edge l is (dis)connected is given by $\mathbb{P}(x_l \equiv 1) = (s_l - 1)p_l + 1$ (resp. $\mathbb{P}(x_l \equiv 0) = (1 - s_l)p_l$). The interpretation for the vector s here is that, if a particular element $s_l = 1$, it is deterministically connected and considered *immune* to the disturbance or attack. The

coordinator's problem is formulated as

$$\begin{aligned}
\mathscr{P}2 : \max_{s,\alpha} \quad & \alpha, && (5.2a)\\
\text{subject to} \quad & \mathbb{E}\left[E(\operatorname{diag}(x))E^\top\right] \succeq \alpha \mathbb{I}_n, && (5.2b)\\
& \mathbb{P}(x_l \equiv 1) = (s_l - 1)p_l + 1, && (5.2c)\\
& \mathbb{P}(x_l \equiv 0) = (1 - s_l)p_l, && (5.2d)\\
& \sum_{l=1}^{m} s_l \leq k_s, && (5.2e)\\
& s \in \mathsf{S}, \quad s \in \{0,1\}^m. && (5.2f)
\end{aligned}$$

Due to the linearity of the expectation operator, (5.2b) is equivalent to $E(\operatorname{diag}((s - \mathbf{1}_m) \diamond p + \mathbf{1}_m))E^\top \succeq \alpha \mathbb{I}_n$, which is an LMI (linear matrix inequality) in s. We note that, as in $\mathscr{P}1$, the objective of $\mathscr{P}2$ may be thought of as a pointwise infimum of linear functions and, as such, is a binary concave-maximization problem in s.

The formulation in $\mathscr{P}2$ presupposes a fixed vector p. However, it may be the case that a strategic attacker detects preventive action taken by the coordinator and adjusts her strategy to improve the likelihood of disconnecting the network. We now formulate the attacker's problem for some known, fixed coordinator strategy s:

$$\begin{aligned}
\mathscr{P}3 : \min_{p,\alpha} \quad & \alpha, && (5.3a)\\
\text{subject to} \quad & \lambda_2(\mathbb{E}\left[E(\operatorname{diag}(x))E^\top\right]) = \alpha, && (5.3b)\\
& \mathbb{P}(x_l \equiv 1) = (s_l - 1)p_l + 1, && (5.3c)\\
& \mathbb{P}(x_l \equiv 0) = (1 - s_l)p_l, && (5.3d)\\
& p \in \mathsf{P}, \quad p \in [0,1]^m. && (5.3e)
\end{aligned}$$

Notice here that the optimization is instead over p, and is now a minimization of α. The

constraint (5.3b) is now a nonlinear equality rather than an LMI, which manifests itself from this becoming a concave-*minimization* problem. This equality is not a convex constraint and will be addressed in Section 5.4. We assume P is convex. In addition, we assume it is compact, which has the interpretation of assuming the attacker's resources are finite, in some sense.

5.3 An SDP Relaxation for Topology Design

This section aims to develop a relaxed approach to solve $\mathscr{P}1$ in a computationally efficient manner. Ideally, such an approach may also be straightforwardly extended to problems of the form $\mathscr{P}2$. To do this, we draw on intuition from the randomized hyperplane strategy given in [49] for solving the well-studied MAXCUT problem.

We now provide a novel reformulation of $\mathscr{P}1$ which is well suited for SDP relaxation. There are two notable differences between $\mathscr{P}1$ and MAXCUT: the entries of the decision vector in $\mathscr{P}1$ take values in $\{0,1\}$, whereas in MAXCUT, the decision (let's say z) takes values $z_i \in \{-1,1\}$. The latter is convenient because it is equivalent to $z_i^2 = 1$. Additionally, the enumeration in MAXCUT is *symmetric* in the sense that, if $z^\star$ is a solution, then so is $-z^\star$. However, $\mathscr{P}1$ is *assymmetric* in the sense that, if $x^\star$ is a solution, it *cannot* be said that $-2x^\star + \mathbf{1}_m$ is a solution (effectively swapping the zeros and ones in the elements of $x^\star$). We rectify these issues with a transformation and variable lift, respectively. Introduce a vector

$$y = 2x - \mathbf{1}_m \tag{5.4}$$

and notice $x \in \{0,1\}^m$ maps to $y \in \{-1,1\}^m$. Then, define $Y = yy^\top$ so that $y_l^2 = 1$ may be enforced via $Y_{ll} = 1$, $l \in \{1,\dots,m\}$. In addition, define $\widetilde{Y} = \begin{bmatrix} y \\ 1 \end{bmatrix} \begin{bmatrix} y \\ 1 \end{bmatrix}^\top = \begin{bmatrix} Y & y \\ y^\top & 1 \end{bmatrix}$ to capture the asymmetry in the original variable x. Now, we are ready to reformulate $\mathscr{P}1$ as an SDP in the

variable y:

$$\mathscr{P}4: \max_{Y,y,\alpha} \quad \alpha, \tag{5.5a}$$

$$\text{subject to} \quad \frac{1}{2}E(\text{diag}(y)+I_m)E^\top \succeq \alpha \mathbb{I}_n, \tag{5.5b}$$

$$\widetilde{Y} = \begin{bmatrix} Y & y \\ y^\top & 1 \end{bmatrix} \succeq 0, \tag{5.5c}$$

$$\text{rank}\,(\widetilde{Y}) = 1, \tag{5.5d}$$

$$\widetilde{Y}_{ll} = 1,\ l \in \{1,\dots,m\}, \tag{5.5e}$$

$$y \in \mathscr{Y}, \tag{5.5f}$$

$$y_l = 1,\ l \sim (i,j) \in \mathscr{E}_0, \tag{5.5g}$$

$$\frac{1}{2}\sum_i (y_i+1) \leq k + |\,\mathscr{E}_0\,|. \tag{5.5h}$$

where $\mathscr{Y}$ is an affine transformation on the set $\mathscr{X}$ in (5.1d); we have simply used (5.4) and the variable lift to rewrite the constraints. The problem $\mathscr{P}4$ is equivalent to $\mathscr{P}1$: the NP-hardness now manifests itself in the nonlinear constraint (5.5d). Dropping this constraint produces a *relaxed* solution $\widetilde{Y}^\star$ with the rank of $\widetilde{Y}^\star$ not necessary one.

This also produces a solution $y^\star$ which can be mapped back to $x^\star$. Of course, $x^\star$ may not take binary values due to the dropped rank constraint. We now briefly recall the geometric intuition for the solution to MAXCUT in [49] with many technical details omitted here for brevity. Let $Z^\star \in \mathbb{R}^{m_z \times m_z}$ be a rank r_z solution to the rank-relaxed MAXCUT SDP problem. Decompose $Z^\star = W^\top W$ with $W \in \mathbb{R}^{r_z \times m_z}$, and notice the columns of W given by $w_i \in \mathbb{R}^{r_z}$, $i \in \{1,\dots,m_z\}$, are vectors on the r_z-dimensional unit ball due to $Z^\star_{ii} = 1$, $i \in \{1,\dots,m_z\}$. Then, generate a uniformly random unit vector $p \in \mathbb{R}^{r_z}$ which may define a hyperplane. If the vector w_i lies on one side of the hyperplane, i.e. $\langle w_i, p\rangle > 0$, place the corresponding node i in the first partition. If it is on the other side of the hyperplane, $\langle w_i, p\rangle < 0$, place node i in the second partition (equality is an event with probability 0). Geometrically speaking, the stronger a vector

w_i is aligned with p, the more "correlated" (for lack of a better term) node i is with the first partition and vice-versa. From another perspective, consider the case $r_z = 1$ which implies the solution is equivalent to the nonrelaxed problem. Then, $\langle w_i, p\rangle \in \{-1, 1\}$ and the partitioning gives the exact optimizer for MAXCUT.

For our problem, decompose $\widetilde{Y}^\star = U^\top U$ with $U \in \mathbb{R}^{r\times m+1}$, and obtain unit-vectors $u_l \in \mathbb{R}^r$, $l \in \{1,\ldots,m+1\}$, from the columns of U. Because of the asymmetry of our problem, we do not implement a random approach to partition the solution. Instead, notice that the last column u_{m+1} is qualitatively different than u_l, $l \in \{1,\ldots,m\}$ due to the variable lift. We have that $y_l = \langle u_l, u_{m+1}\rangle, l \in \{1,\ldots,m\}$. Thus, larger entries of y_l correspond to vectors u_l on the unit ball which are more "aligned" with u_{m+1}, which hearkens to the geometric intuition for the MAXCUT solution. In this sense, u_{m+1} may be thought of as the partitioning vector p as before, and entries of y_l give a quantitative measure of the inclination for edge $l \sim (i, j)$ to be connected. For this reason, we suggest iteratively choosing the edge l associated with the largest element of y for which $l \notin \mathscr{E}_0$. If a particular edge is infeasible, this is elegantly accounted for by (5.5f) and is reflected in the relaxed solution to $\mathscr{P}4$. This approach may be iterated k times, updating $\mathscr{E}_0$ and decrementing k each time in accordance with (5.5h), to construct a satisfactory solution to the original NP-hard binary problem. In addition, this formulation is easily adaptable to solve $\mathscr{P}2$ via a similar transformation and variable lift in s.

5.4 Protecting Links Against an Adversary

This section begins by studying the Nash equilibria of a game between the coordinator and attacker where they take turns solving $\mathscr{P}2$ and $\mathscr{P}3$. We first study the (non)existence of the Nash equilibria of this game, and use this result to motivate the development of a *preventive* strategy for the coordinator. We then provide some auxiliary results about the solutions of $\mathscr{P}3$ and use these to justify methods for finding such a preventive strategy.

5.4.1 Nash Equilibria

We begin by adopting the shorthand notation $L(s,p) = \mathbb{E}\left[E(\mathrm{diag}(x))E^\top\right]$ with x distributed as in (5.2c)–(5.2d), i.e. $L(s,p)_{ij} = (1-s_l)p_l - 1$, $l \sim (i,j) \in \mathscr{E}, i \neq j$ and $L(s,p)_{ii} = -\sum_{j\in\mathscr{N}_i} L(s,p)_{ij}$. This matrix may be interpreted as a weighted Laplacian whose elements are given by the righthand side of (5.2c), (5.3c). We also adopt the shorthand $\alpha(s,p) = \inf_z\{z^\top L(s,p)z \mid z \perp \mathbf{1}_n, \|z\|_2 = 1\}$ to refer to the Fiedler eigenvalue of $L(s,p)$, and note that $\alpha(s,p)$, as a pointwise infimum of bilinear functions, is concave-concave in (s,p). From this, recall the first-order concavity relation [17]

$$\begin{aligned}
&\alpha(s^2,p) \leq \alpha(s^1,p) + \nabla_s\alpha(s^1,p)^\top(s^2-s^1),\ \forall s^1,s^2 \in \mathsf{S},\\
&\alpha(s,p^2) \leq \alpha(s,p^1) + \nabla_p\alpha(s,p^1)^\top(p^2-p^1),\ \forall p^1,p^2 \in \mathsf{P}.
\end{aligned} \tag{5.6}$$

To get a better grasp on this, we compute the gradient of α with respect to both s and p. Let v be the Fiedler eigenvector associated with the second-smallest eigenvalue (in this case, α) of $L(s,p)$. Then,

$$\frac{\partial\alpha}{\partial s_l} = v^\top\frac{\partial L(s,p)}{\partial s_l}v, = v^\top p_l e_l e_l^\top v = p_l(v_i - v_j)^2, \tag{5.7}$$

which is a straightforward extension of the computation shown near the end of Section 5.2.1. Additionally,

$$\frac{\partial\alpha}{\partial p_l} = \begin{cases} -(v_i-v_j)^2, & s_l = 0\\ 0, & s_l = 1. \end{cases} \tag{5.8}$$

The gradient with respect to s and p is a vector with elements in (5.7)–(5.8), which are nonpositive for p and nonnegative for s. Additionally, note that $v \neq \mathbf{1}_n$, $v \neq 0$, implying the quantity $(v_i - v_j)^2$ must be strictly positive for some edges $l \sim (i,j)$.

Now, consider a game where the coordinator and attacker take turns solving and implementing the solutions of $\mathscr{P}2$ and $\mathscr{P}3$, respectively. A Nash equilibrium is a point $(s^\star,p^\star)$ with

the property

$$\alpha(s, p^\star) \leq \alpha(s^\star, p^\star) \leq \alpha(s^\star, p), \forall s \in \mathsf{S}, \forall p \in \mathsf{P}, \tag{5.9}$$

which is a stationary point of the aforementioned game. We now state a lemma to motivate the remainder of this section.

Lemma 9. ***(Nonexistence of Nash Equilibrium).*** A Nash equilibrium point $(s^\star, p^\star)$ satisfying (5.9) is not guaranteed to exist in general.

Proof. To show this result, we provide a simple counterexample. Consider a complete graph with $n = 3$ nodes and $m = 3$ edges, $l \in \{1,2,3\} \sim \{(1,2),(1,3),(2,3)\} = \mathscr{E}$, and the coordinator and attacker feasibility sets $\mathsf{S} = \{0,1\}^3 \cap \{s \mid \sum_l s_l \leq 2\}$, $\mathsf{P} = \{p \mid \sum_l p \leq 1\}$. Note that, for $s_1 = 0, p_1 \in (0,1], p_{-1} = 0$, we have the Fiedler vector $v = (\sqrt{2}/2, -\sqrt{2}/2, 0)^\top$, implying $\dfrac{\partial \alpha}{\partial p_1} < 0$ on the interval $p_1 \in (0,1]$ (this extends to different choices of the "attacked" edge with the components of v reordered accordingly), with $\left.\dfrac{\partial \alpha}{\partial p_1}\right|_{p_1=0}$ not well defined due to nonuniqueness of $\lambda_2(s,p)$ at this point. We state this observation for completeness, but for the sake of simplicity the remainder of the proof does not require it.

We first establish the values of $\alpha(s,p)$ in a few cases. A play (s^1, p^1) with any $s^1 \in \mathsf{S}$ and $p^1 = \mathbf{0}_m$ gives $\alpha(s^1, p^1) = 3$; a play (s^2, p^2) with $p^2_l = 1, p^2_{-l} = 0$ and $s^2_l = 0, s^2_{-l} \in \{0,1\}$ gives $\alpha(s^2, p^2) = 1$; and a play (s^3, p^3) with $s^3_l = 1, s^3_{-l} \in \{0,1\}$ (one of these componenets must be zero for feasibility) and $p^3_l = 1, p^3_{-l} = 0$ gives $\alpha(s^3, p^3) = 3$. Note that $\alpha(s,p) \in [1,3], \forall s \in \mathsf{S}, \forall p \in \mathsf{P}$, and if $\nexists l$ such that $p_l = 1$, then $\alpha(s,p) > 1$. We refer to the latter condition as $(*)$ and the calculation is omitted for brevity.

Now, consider an initial play $s(0)$ by the coordinator with $\sum_l s_l(0) = 2$. The dynamics of the game dictates that the attacker solves $\mathscr{P}3$ and chooses the optimal play with $p_l(0) = 1$ for the edge l corresponding to $s_l(0) = 0$, giving $\alpha(s(0), p(0)) = 1$. Note that generality is not lost for the choice of $s(0)$ due to $(*)$. The lefthand side of (5.9) is violated, so $(s(0), p(0))$ is not a Nash equilibrium. The coordinator then solves $\mathscr{P}2$ and makes a play $s(1)$, with $s_l(1) = 1$ for the corresponding edge that $p_l(0) = 1$. This gives $\alpha(s(1), p(0)) = 3$. Now the righthand side of (5.9)

is violated, so $(s(1), p(0))$ is not a Nash equilibrium. The attacker then resolves $\mathscr{P}3$ and chooses $p_l(1) = 1$ for the edge l s.t. $s_l(1) = 0$, giving $\alpha(s(1), p(1)) = 1$. These dynamics continue with $\alpha(s(t), p(t-1)) = 3$ and $\alpha(s(t), p(t)) = 1, \forall t > 0$, and from generality we conclude a Nash equilibrium does not exist.

□

The near-trivial nature of the counterexample employed in the proof of Lemma 9 gives credence to the idea that Nash equilibria are unlikely to exist in more meaningful instances. This result should not come as a surprise: the solutions to $\mathscr{P}2$ and $\mathscr{P}3$ are in direct conflict with one another, and playing sequentially has the effect of the coordinator "chasing" the attacker around the network. We find that the cases for which we can construct a Nash equilibrium are trivial: for example, if $\mathbf{1}_m \in \mathsf{S}$, the coordinator may choose $s^\star = \mathbf{1}_m$, and the attacker's solution set is trivially the whole set P with the interpretation that the attacker is powerless to affect the value of α. Then, $(\mathbf{1}_m, p), \forall p \in \mathsf{P}$ are Nash equilibria, and are not interesting.

5.4.2 Coordinator's Preventive Strategy

Lemma 9 motivates the study of an optimal *preventive* strategy for the coordinator under the assumption that the attacker may always make a play in response to the coordinator's action. Instead of a Nash equilibrium satisfying (5.9), we seek a point $(s^\star, p^\star)$ satisfying the following:

$$(s^\star, p^\star) = \operatorname*{argmax}_{s \in \mathsf{S}} \operatorname*{argmin}_{p \in \mathsf{P}} \alpha(s, p). \tag{5.10}$$

The interpretation of $s^\star$ solving (5.10) is that it provides the best-case solution for the coordinator given that the attacker makes the last play. In this sense, $s^\star$ is not optimal for $p^\star$; rather, it is an optimal play with respect to the whole set P.

From the coordinator's perspective, the objective function $\operatorname*{argmin}_{p \in \mathsf{P}} \alpha(s, p)$ is a pointwise infimum of concave functions of s, and therefore the problem is a concave maximization. However, computing such a point may be dubious in practice, particularly since we have not

assumed the coordinator has the capability of solving the concave minimization problem $\mathscr{P}3$ or even knowledge of P. It would, however, be convenient to use the attacker's solutions to $\mathscr{P}3$ against herself. To do this, we establish some lemmas to gain insight on the solution sets of the attacker. This helps us construct heuristics for computing $s^\star$ in the sense of (5.10). We assume $\mathsf{S},\mathsf{P} \neq \emptyset$.

Lemma 10. ***(Attacker's Solution Tends to be Noninterior).*** Consider the set of solutions $\mathsf{P}^\star \subseteq \mathsf{P}$ to $\mathscr{P}3$ for some s. If there exists point $p^\star \in \mathsf{P}^\star$ which is an interior point of P, then $\mathsf{P}^\star = \mathsf{P}$.

Proof. Consider $p^\star \in \mathsf{P}^\star$ which is an interior point of P. Pick $\varepsilon > 0$ so $\mathscr{B}_\varepsilon(p^\star) \subseteq \mathsf{P}$. Then, from (5.6), the point $p' = p^\star - \varepsilon \nabla_p \alpha(s, p^\star) \in \mathsf{P}$ violates the condition that $p^\star$ is a solution to $\mathscr{P}3$ unless $\nabla_p \alpha(s, p^\star) = \mathbf{0}_m$, so this must be the case. Then, in accordance with (5.6), all $p' \in \mathsf{P}$ are solutions, and the statement $\mathsf{P}^\star = \mathsf{P}$ follows. □

This lemma implies the solution set $\mathsf{P}^\star$ consists of noninterior point(s) of P except in trivial cases. We now provide a stronger result in the case where P is a polytope, which shows that solutions tend to be contained in low-dimensional faces of P such as edges (line segments) and vertices (points).

Lemma 11. ***(Attacker's Solutions Tend Towards Low-Dimensional Faces).*** Let $\mathsf{P} = \mathscr{A}_1 \cap \cdots \cap \mathscr{A}_r \subset \mathbb{R}^m$ be a compact polytope with half-spaces $\mathscr{A}_i$ characterized by a_i, b_i for $i \in \{1, \ldots, r\}$, and $\mathscr{F}$ be a face of P with $a_j^\top p = b_j$ for $j \in \mathscr{J} \subseteq \{1, \ldots, r\}$, $\forall p \in \mathscr{F}$. If a point $p^\star \in \text{relint}\,(\mathscr{F})$ is a solution to $\mathscr{P}3$, then $\nabla_p \alpha(s, p^\star) \in \text{span}\{a_j\}_{j \in \mathscr{J}}$.

Proof. Suppose that $\nabla_p \alpha(s, p^\star) \notin \text{span}\{a_j\}$ and decompose $\nabla_p \alpha(s, p^\star) = c + d, c \in \text{span}\{a_j\}$, $d \in \text{span}\{a_j\}^\perp, d \neq \mathbf{0}_m$. Now consider the point $p' = p^\star - \varepsilon d / \|d\|_2 \in \mathscr{F}$ for some $\varepsilon > 0$. From (5.6), we have the relation $\alpha(s, p') \leq \alpha(s, p^\star) - \varepsilon \|d\|_2$ with $\varepsilon > 0$ and $\|d\|_2 > 0$, which contradicts $p^\star$ being a solution and completes the proof. □

To interpret the result of Lemma 11, notice that $p \in \text{relint}\,(\mathscr{F})$ implies p does not belong to a lower dimension face, and that the the dimension of $\text{span}\{a_j\}_{j \in \mathscr{J}}$ becomes large only as

the dimension of $\mathscr{F}$ becomes small, which intuitively suggests that the gradient of α at $p^\star$ may only belong to $\text{span}\{a_j\}_{j\in\mathscr{J}}$ for a $p \in \mathscr{F}$ if this span is large in dimension. This allows us to conservatively characterize the solution set $\mathsf{P}^\star$, and the result gives credence to the notion that solutions take values in low-dimensional faces of P.

These results are necessarily conservative due to the lack of understanding in the literature of how v evolves with a changing p, which shows up in the elements of $\nabla_p \alpha(s,p)$. For this reason, it is our intent to instead use Lemmas 10 and 11 to establish intuition for the problem and justify solution strategies to the hard problem of computing a preventive $s^\star$.

Before proceeding, we establish one more simple lemma and provide discussion on deterministically connected graphs.

Lemma 12. ***(Deterministic Connectivity).*** Assume $\exists p \in \mathsf{P}$ with $p \succ \mathbf{0}_m$ and that the attacker makes the last play. Let $L(x)$ indicate the random matrix whose elements are distributed as in (5.3c)–(5.3d). Then, a coordinator's strategy s gives $\lambda_2 > 0$ of $L(x)$ with probability 1 if and only if the elements of s equal to 1 are associated with edges of a connected graph.

Proof. $\Rightarrow$) We do not need to assume the attacker has made an optimal play with respect to $\mathscr{P}3$. Instead, consider any play $p \succ \mathbf{0}_m$ and note that each edge l associated with $s_l = 0$ has a nonzero probability of being disconnected. Then, there is a nonzero probability that $x_l = 0$ for each l with $s_l = 0$, and the remaining protected edges do not form a connected graph. Then, if the elements of s equal to one are not associated edges defining a connected graph, $\lambda_2 = 0$ is an event with nonzero probability.

$\Leftarrow$) This direction is trivial: $x_l = 1$ with probability 1 for edges l corresponding to $s_l = 1$. If these edges form a connected graph, then $\lambda_2 > 0$ with probability 1. □

The consequence of Lemma 12 is obvious: if the coordinator does not have the resources to protect edges which form a connected graph and the attacker targets all edges, then there is no guarantee the resulting graph will be connected. A subject of future work is to provide some insight on the lower bound of λ_2 for particular cases of S and P.

5.4.3 Heuristics for Computing a Preventive Strategy

Recall from the previous subsection that the goal is to compute $s^\star$ as a solution to (5.10). In this subsection, we describe three approaches to computing a satisfactory solution and formally adopt the following assumptions.

Assumption 18. ***(Coordinator's Problem is Solvable).*** Given a known vector p, the coordinator can find the optimal solution of $\mathscr{P}2$.

The binary constraint makes $\mathscr{P}2$ NP-hard, which we addressed in Section 5.3 by developing an effective relaxation which may be extended to solving $\mathscr{P}2$.

Assumption 19. ***(Attacker Plays Optimally and Last).*** The attacker's play p belongs to a convex, compact set P. In addition, she always makes optimal plays which solve $\mathscr{P}3$ given the coordinator's play s, and she may always play in response to the coordinator changing his decision.

Assumption 20. ***(Available Information).*** The coordinator is cognizant of Assumption 19 and has access to the current attacker play p. He may compute $\alpha(s,p)$ for a particular play (s,p).

Assumption 21. ***(Only Last Play Matters).*** The objective $\alpha(s,p)$ (and, by extension, λ_2) is only consequential once both the coordinator and attacker have chosen their final strategy and do not make additional plays.

With these assumptions, we construct three heuristics for computing $s^\star$ iteratively by observing the plays of the attacker $p(t)$ over a time horizon $t \in \{1,\dots,T\}$.

Algorithm 5 is simple and doesn't utilize the attacker's plays. For each t, it constructs $s(t)$ by picking edges uniformly randomly. Each loop terminates when no feasible edges remain. The value of α is recorded following the attacker's response $p(t)$, and the best $s(t)$ is returned.

Algorithm 6 utilizes a weighting function η with the property $\sum_{k=1}^{t}\eta(k)=1$. We suggest

Algorithm 5. Random Sampling

procedure RAND($\mathsf{S}, \mathscr{E}, T$)
  for $t = 1, \ldots, T$ **do**
    $\mathscr{M} \leftarrow \{1, \ldots, m\}$
    $s(t) \leftarrow \mathbf{0}_m$
    done $\leftarrow false$
    while done $= false$ **do**
      Choose $l \in \mathscr{M}$ uniformly randomly
      if $s_l(t) \leftarrow 1 \Rightarrow s(t) \in \mathsf{S}$ **then**
        $s_l(t) \leftarrow 1$
      $\mathscr{M} \leftarrow \mathscr{M} \setminus \{l\}$
      if $\nexists l \in \mathscr{M}$ s.t. $s_l(t) \leftarrow 1 \Rightarrow s(t) \in \mathsf{S}$ **then**
        done $= true$
    Play $s(t)$
    Store $s(t)$ and $\alpha(s(t), p(t))$
  $t^\star \leftarrow \underset{t}{\text{argmax}}\ \alpha(s(t), p(t))$
  return $s^\star = s(t^\star)$

Algorithm 6. Convex Combinations

procedure CVX($\mathsf{S}, \mathscr{E}, T$)
  $s(1) \leftarrow \mathbf{0}_m$
  Store $s(1)$, $p(1)$ and $\alpha(s(1), p(1))$
  $p_\theta(1) \leftarrow p(1)$
  for $t = 2, \ldots, T$ **do**
    $s(t) \leftarrow \underset{s \in \mathsf{S}}{\text{argmax}}\ \alpha(s, p_\theta(t-1))$
    Store $s(t)$, $p(t)$, and $\alpha(s(t), p(t))$
    for $k = 1, \ldots, t$ **do**
      $\theta(k) \leftarrow \eta(k)$
    $p_\theta(t) \leftarrow \sum_{k=1}^{t} \theta(k) p(k)$
  $t^\star \leftarrow \underset{t}{\text{argmax}}\ \alpha(s(t), p(t))$
  return $s^\star = s(t^\star)$

three possible choices for η:

$$\begin{aligned}
\eta_1(k) &= 1/t, \\
\eta_2(k) &= \frac{\gamma^{(t-k)}}{\sum_k \gamma^{(t-k)}},\ \gamma \in (0,1), \\
\eta_3(k) &= \frac{\alpha(s(k), p(k))}{\sum_k \alpha(s(k), p(k))}.
\end{aligned}$$

These may be interpreted as a uniform weighting of each observation $p(t)$, a recency-biased weighting, and a penalty-biased weighting. respectively. Algorithm 6 is motivated by a few observations. Firstly, recall Lemmas 9–11 and note that $p(t)$ may jump around extreme points of P as $s(t)$ evolves. The solution $s^\star$ from (5.10) accounts for the whole space P, so successive convex combinations of the solutions $p(t)$ effectively push the coordinator's decision towards a response to the vulnerable parts of the space over time.

Algorithm 7 adopts the following stronger version of Assumption 18.

Assumption 22. ***(Coordinator's Problem is Solvable Over Multiple Points).*** Given a finite set of points $p(t) = \overline{\mathsf{P}} \subset \mathsf{P}$, the coordinator may compute the solution

$$s^\star = \operatorname*{argmax}_{s \in \mathsf{S}} \min_{p \in \overline{\mathsf{P}}} \alpha(s, p),$$

where $s^\star$ is the best preventive play over the set $\overline{\mathsf{P}}$.

Algorithm 7 operates by computing the solution $s(t)$ as the optimal play with respect to each of the previous attacker plays $p(k), k < t$. Although it is more computationally demanding than Algorithms 5 and 6, it is more strongly rooted in the theoretical understanding of the problem we have developed in the following sense: the convex hull of these points $\mathrm{co}(\overline{\mathsf{P}})$ at time t is a compact polytope whose vertices are defined by the points $p(k)$, and applying Lemma 11, it stands to reason that points in the interior or in higher-dimensional faces of $\mathrm{co}(\overline{\mathsf{P}})$ are uncommon solutions. We expect $\mathrm{co}(\overline{\mathsf{P}})$ to grow in each loop of the algorithm and effectively reconstruct

Algorithm 7. Constructing P via Pointwise Search

procedure SEARCH($\mathsf{S}, \mathscr{E}, T$)
  $s(1) \leftarrow \mathbf{0}_m$
  Store $s(1)$, $p(1)$, and $\alpha(s(1), p(1))$
  $\overline{\mathsf{P}} \leftarrow p(1)$
  for $t = 2, \ldots, T$ **do**
    $s(t) \leftarrow \underset{s \in \mathsf{S}}{\operatorname{argmax}} \min_{p \in \overline{\mathsf{P}}} \alpha(s, p)$
    Store $s(t)$, $p(t)$ and $\alpha(s(t), p(t))$
    $\overline{\mathsf{P}} \leftarrow \overline{\mathsf{P}} \cup \{p(t)\}$
  $t^\star \leftarrow \underset{t}{\operatorname{argmax}}\ \alpha(s(t), p(t))$
  return $s^\star = s(t^\star)$

the attacker's feasibility set P. For now, convergence to the true $s^\star$ is not guaranteed due to the difficulty of characterizing the evolution of $\nabla_p \alpha(s, p)$ with p. However, we note in simulation studies that Algorithm 7 converges to the global optimizer in a few iterations.

Finally, we state the following trivial lemma for completeness.

Lemma 13. ***(Nondecreasing Performance of Algorithms 5–7).*** Let $p^\star$ solve $\mathscr{P}3$ for $s = s^\star(T)$, where $s^\star(T)$ is the returned strategy of Algorithm 5, 6, or 7 truncated at time T. For all $T > 1$, $\alpha(s^\star(T), p^\star) \geq \alpha(s^\star(T-1), p^\star)$.

Proof. The result is trivially seen in that $s^\star(T) = s(t^\star), t^\star = \underset{t \in \{1,\ldots,T\}}{\operatorname{argmax}}\ \alpha(s(t), p(t))$ and $\{1, \ldots, T-1\} \subset \{1, \ldots, T\}$. □

5.5 Simulations

We now examine our proposed SDP relaxation for solving $\mathscr{P}1$. For ease of comparison with the Fiedler vector heuristic given in [45], we do not include any additional convex constraints beyond (5.1c). For a network with 14 nodes, $|\mathscr{E}_0| = 28$ initial edges generated randomly, we implement the Fiedler method, our SDP method, and the approach of taking the convex hull of the feasibility set of $\mathscr{P}1$. We ran 100 trials with different topology initializations for each of the $k \in \{25, 40\}$ edge-addition cases. Our SDP design outperforms the Fiedler vector heuristic

in 75 trials of the $k = 25$ case, 80 trials of the $k = 40$ case, and it outperforms the convex hull approach in all 100 trials of each case. This improved performance is observed over a variety of network sizes and initial connectivities, and we observe that increasing k is likelier to improve the performance of our method over the alternatives.

One such instance of $k = 25$ added edges is plotted in Figure 5.1 and the performance is plotted in Figure 5.2. Firstly, note that both our SDP method and the Fiedler vector method greatly outperform the simple convex hull approach. We note that our SDP method is outperformed by the Fiedler method in early iterations. This is due to the greedy nature of the Fiedler heuristic: it does not account for the entire horizon of adding k edges, in contrast with our method. In later iterations, the performance of our method catches up with and surpasses the Fiedler vector heuristic (this is common behavior across other initializations). We contend that the reason for this is that the solution of $\mathscr{P}2$ at each iteration is cognizant of the entire problem horizon, as opposed to the Fiedler vector heuristic which greedily chooses edges in accordance with the direction of steepest ascent in λ_2.

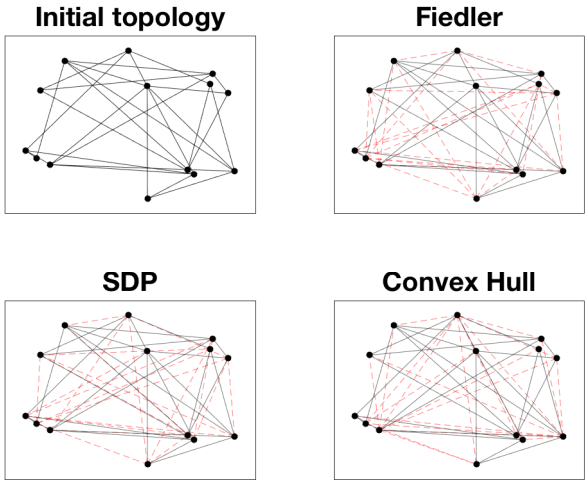


Figure 5.1. Initial topology of 14 nodes and 28 edges. Three methods are implemented to grow the network to 53 edges, with the additional edges plotted as red dotted lines.

Next, we study a small network of 7 nodes and 11 edges so that the solutions to $\mathscr{P}2$

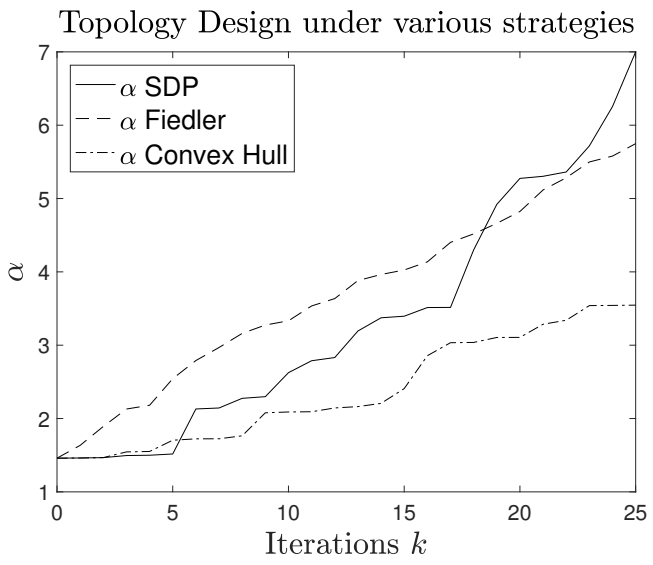


Figure 5.2. Performance each method over $k = 25$ iterations.

and $\mathscr{P}3$ may be brute-forcibly computed to test Algorithms 5–7, with $\eta_1(k)$ being used for Algorithm 6. The network is shown in Figure 5.3. We choose $\mathsf{P} = [0.25, 0.75]^{11} \cap \{p \mid \sum_l p_l \leq 4.25\}$ and $\mathsf{S} = \{0,1\}^{11} \cap \{s \mid \sum_l s_l \leq 5\}$. We run the algorithms for $T = 30$ iterations and plot the results at each iteration in Figure 5.4.

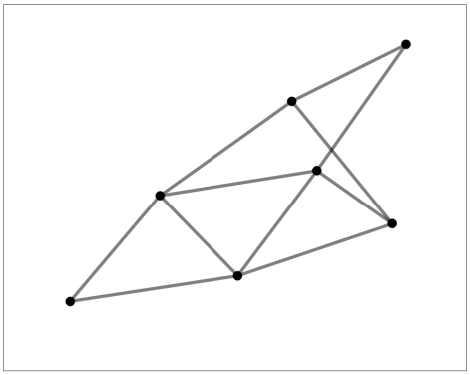


Figure 5.3. Network of 7 nodes and 11 edges for algorithm case study.

Clearly, Algorithm 5 does not improve across iterations due to the fact that it does not utilize information about the attacker's plays from previous iterations. Additionally, if general convex constraints are included, it may not account for these particularly well. This

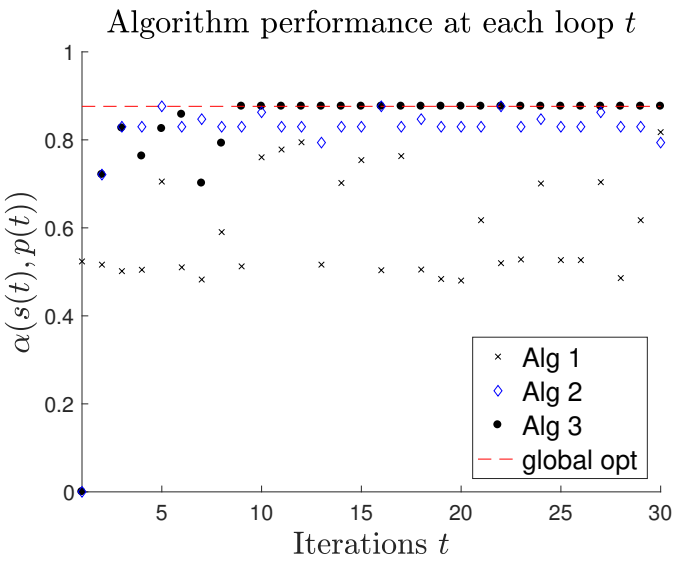


Figure 5.4. Performance of Algorithms 5–7 at each loop t.

algorithm does achieve a maximum value of $s(t^\star) = 0.8175$, which is a bit below the global optimum $\alpha(s^\star, p^\star) = 0.8762$, although there is certainly no guarantee that this may reliably occur in general. Algorithm 6 achieves the global optimum $\alpha(s(t^\star), p(t^\star)) = 0.8762$ the fastest, at $t^\star = 5$, although it never reaches this point again and instead oscillates around suboptimal points. The overall performance of Algorithm 7 is the strongest of the three. Counter-intuitively, the performance does not improve monotonically in t, although this should be expected: early points may get "lucky", in some sense, but the subsequent iteration may allow the attacker to jump to a new vulnerable part of the space. Once the algorithm achieves the global optimum at $t^\star = 9$, it does not dip below this for the remainder of the time horizon. We observe similar behavior of each algorithm when implemented on other small graphs.

Acknowledgements

The material in this chapter, in full, is a reprint of *Maximizing Algebraic Connectivity of Constrained Graphs in Adversarial Environments*, T. Anderson, C.Y. Chang and S. Martínez, 2018 European Control Conference (ECC), Limassol, 2018, pp. 125-130. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Frequency Regulation with Heterogeneous Energy Resources: A Realization using Distributed Control

This chapter presents one of the first real-life demonstrations of coordinated and distributed resource control for secondary frequency response in a power distribution grid. A series of tests involved up to 69 heterogeneous active distributed energy resources consisting of air handling units, unidirectional and bidirectional electric vehicle charging stations, a battery energy storage system, and 107 passive distributed energy resources consisting of building loads and solar photovoltaic systems. The distributed control setup consists of a set of Raspberry Pi end-points exchanging messages via an ethernet switch. Actuation commands for the distributed energy resources are obtained by solving a power allocation problem at every regulation instant using distributed ratio-consensus, primal-dual, and Newton-like algorithms. The problem formulation minimizes the sum of distributed energy resource costs while tracking the aggregate setpoint provided by the system operator. We demonstrate accurate and fast real-time distributed computation of the optimization solution and effective tracking of the regulation signal over 40-minute time horizons. An economic benefit analysis confirms eligibility to participate in an ancillary services market and demonstrates up to $49k of potential annual revenue for the selected population of distributed energy resources.

The results of this chapter are the outcome of a project under the ARPA-e Network

Optimized Distributed Energy Systems (NODES) program[1],which postulates DER aggregations as virtual power plants that enable variable renewable penetrations of at least 50%. The vision of the NODES program was to employ state-of-the-art tools from control systems, computer science, and distributed systems to optimally respond to dynamic changes in the grid by leveraging DERs while maintaining customer quality of service. The NODES program required testing with at least 100 DERs at power. Here, we demonstrate the challenges and opportunities of testing on a heterogeneous fleet of DERs for eventual operationalization of optimal distributed control at frequency regulation time scales.

6.1 Bibliographical Comments

To the best of our knowledge, real-world testing of frequency regulation by DERs has been limited. A Vehicle-to-Grid (V2G) electric vehicle (EV) [61] and two Battery Energy Storage Systems (BESS) [93] provided frequency regulation. 76 bitumen tanks were integrated with a simplified power system model to provide frequency regulation via a decentralized control algorithm in [29]. In buildings, a decentralized control algorithm controlled lighting loads in a test room [67], centralized frequency control was applied to an air handling unit (AHU) [66, 101], an inverter and four household appliances [68], and four heaters in different rooms [41]. A laboratory home with an EV and an AHU, and a number of simulated homes were considered for demand response in [8] through an aggregator at a 10 s level. Technologies for widespread, but centrally controlled, cycling of air conditioners directly by utilities cf. [87] and aggregators are common place for peak shifting, but occur over time scales of minutes to hours. Industrial solutions enabling heterogeneous DERs to track power signals also exist, but they are either centralized, cf. [30] or require all-to-all communication [95].

Our literature review exposes the following limitations: (i) centralized control or need for all-to-all communication [8, 30, 41, 61, 66, 68, 87, 93, 95, 101], which does not scale to millions of DERs; (ii) small numbers of DERs [8, 41, 61, 66, 68, 93, 101]; (iii) lack of diversity in

[1] https://arpa-e.energy.gov/arpa-e-programs/nodes

DERs [29,41,61,66,67,93,101], with associated differences in tracking time scales and accuracy. No trial has been reported that demonstrated generalizability to a real scenario with (i) scalable distributed control and a (ii) large number of (iii) heterogeneous DERs.

Statement of Contributions

To advance the field of real-world testing of DERs for frequency control, we conduct a series of tests using a group of up to 69 active and 107 passive heterogeneous DERs on the University of California, San Diego (UCSD) microgrid [104]. To the best of the authors' knowledge, this is the first work to consider such a large, diverse portfolio of real physical DERs for secondary frequency response. As such, the major contributions of this work are:

- A detailed account of the testbed, including the DER actuation and sampling interfaces, the distributed optimization setup, and communication framework.
- A description of techniques to work around technical barriers, provision of lessons learned, and suggestions for future improvement.
- Evaluation of the performance of both the cyber and physical layers, including an evaluation of eligibility requirements for and the economic benefit of participating in the ancillary services market.

Chapter Overview. Frequency regulation is simulated on the UCSD microgrid using real controllable DERs (Section 6.3.3) to follow the PJM RegD signal [3] interpolated from 0.5Hz to 1Hz (Sections 6.3.2). The DER setpoint tracking is formulated as a power allocation problem at every regulation instant (Section 6.3.1), and uses three types of provably convergent distributed algorithms from [**?**, 6, 31, 33] to solve the optimization problem; see the Appendix. Setpoints are computed distributively on multiple Raspberry Pi's communicating via ethernet switches (Section 6.3.4). The setpoints are implemented on up to 176 DERs at power using dedicated command interfaces via TCP/IP communication (Section 6.3.5), the DER power

outputs monitored (Section 6.3.6), and their tracking performance evaluated (Section 6.3.7). Results for the various test scenarios (Section 6.4.1) show that the test system tracks the signal with reasonable error despite delays in response and inaccurate tracking behavior of some groups of DERs, and qualifies for participation in the PJM ancillary services market (Section 6.4.2).

6.2 Problem Setting

This chapter validates real-world DER controllability for participation in secondary frequency regulation through demonstration tests implemented on a real distribution grid. The tests showcase the ability of aggregated DERs to function as a single market entity that responds to frequency regulation requests from the independent system operators (ISO) by optimally coordinating DERs. The goal is to monitor and actuate a set of real controllable DERs to collectively track a typical automatic generation control (AGC) signal issued by the ISO.

Three different distributed coordination schemes optimize the normalized contribution of each DER to the cumulative active power signal. Unlike simulated models, the use of real power hardware exposes implementation challenges associated with measurement noise, sampling errors, data communication problems, and DER response. To that end, precise load tracking is pursued at timescales that differ by DER type consistent with individual DER responsiveness and communication latencies, yet meet frequency regulation requirements in aggregation.

The 69 kV substation and 12 kV radial distribution system owned by UCSD to operate the 5 km^2 campus was the chosen demonstration testbed. It has diverse energy resources with real-time monitoring and control capabilities, allowing for active load tracking. This includes over 3 MW of solar photovoltaic (PV) systems, 2.5 MW/5 MWh of BESS, building heating ventilation and air conditioning (HVAC) systems in 14 million square feet of occupied space, and over 200 unidirectional V2G (V1G) and V2G EV chargers. The demonstration tests used a representative population of up to 176 such heterogeneous DERs to investigate tracking behavior of specific DER types as well as their cooperative tracking abilities. While the available DER

capacity at UCSD far exceeds the minimum requirements for an ancillary service provider set by most ISOs (typically $\sim$ 1 MW), logistical considerations and controller capabilities dictated the choice of a DER population size with less aggregate power capacity (up to 184 kW) for this demonstration. Since this magnitude of power is insufficient to measurably impact the actual grid frequency, we chose to simulate frequency regulation by following a frequency regulation signal.

6.3 Test Elements

Here, we elaborate on the different elements of the validation tests. These include the optimization formulation employed to compute DER setpoints (Section 6.3.1), the reference AGC signal (Section 6.3.2) and types of DERs used to track it (Section 6.3.3), the computing platform (Section 6.3.4), the actuation (Section 6.3.5) and monitoring interfaces (Section 6.3.6), the performance metrics used to assess the cyber and physical layers, and eligibility for market participation (Section 6.3.7).

6.3.1 Optimization Formulation

The optimization model for AGC signal tracking using DERs can be mathematically stated as a separable resource allocation problem subject to box constraints as follows:

$$\begin{aligned} \min_{p\in\mathbb{R}^n} \; & f(p) = \sum_{i=1}^{n} f_i(p_i), \\ \text{s.t.} \; & \sum_{i=1}^{n} p_i = P_{\text{ref}}, \\ & p_i \in [\underline{p}_i, \overline{p}_i], \quad \forall i \in \mathcal{N} = \{1,\dots,n\}. \end{aligned} \tag{6.1}$$

The agents $i \in \mathcal{N}$ each have local ownership of a decision variable $p_i \in \mathbb{R}$, representing an active power generation or consumption quantity (setpoint), a local convex cost function f_i, and local box constraints $[\underline{p}, \overline{p}]$, representing active power capacity limits. P_{ref} is a given active power reference value determined by the ISO and transmitted to a subset of the agents as problem

data, see e.g. [21]. P_{ref} is a signal that changes over time, so a new instance of (6.1) is solved in 1 s intervals corresponding to these changes.

For the validation tests, we used two types of cost functions: constant and quadratic. Constant functions were used for the Ratio-Consensus (RC) solver, which turns the optimization into a feasibility problem. Quadratic functions were used for the primal-dual based (PD) and Distributed Approximate Newton Algorithm (DANA) methods, see the Appendix. The quadratic functions were artificially chosen to produce satisfactorily diverse and representative solutions for each DER population. We split the total time period of the signal, P_{ref} into three equal segments, and implemented RC, PD, and DANA in that order. Box constraints $[\underline{p}_i, \overline{p}_i]$ were typically centered at zero for simplicity, see Section 6.3.3.

6.3.2 Regulation Signal

The 40 min RegD signal published by PJM [3] served as the reference AGC signal for the validation tests, and was used to obtain the value for P_{ref} in (6.1). The normalized RegD signal, contained in $[-1,1]$, was interpolated from 0.5 Hz to 1 Hz. The signal was then treated by subtracting the normalized contributions of building loads and PV systems, cf. Section 6.3.3. Finally, the normalized signal was scaled by a factor proportional to the total DER capacity $\sum_i(\overline{p}_i - \underline{p}_i)$ before sending to the optimization solvers. More precisely,

$$P_{\text{ref}} = \beta \frac{\sum_i(\overline{p}_i - \underline{p}_i)}{\|P_{\text{RegD}} + P_{\text{PV}} - P_{\text{b}}\|_\infty} \left(P_{\text{RegD}} + P_{\text{PV}} - P_{\text{b}}\right), \tag{6.2}$$

where P_{RegD} refers to the normalized RegD signal data, P_{PV} and P_{b} respectively refer to the normalized PV generation and building load data obtained from the UCSD ION server as described in Section 6.3.6, and $0 < \beta < 1$ is an arbitrary scaling constant. For most test scenarios, $\beta = 0.75$ to prevent extreme set points that would require all DERs to operate at either $\overline{p}_i$ or $\underline{p}_i$ simultaneously, which may be infeasible in some time steps due to slower signal update times, see Table 6.1. Each P in (6.2) is a vector with 2401 elements corresponding to each 1 s time

step's instance of (6.1) over the 40 min time horizon. The acquired target regulation signal is characterized by steep positive and negative ramps that range from -14 kW to +16 kW over 1 s intervals and an average absolute ramp-rate of 1.7 kW/s.

6.3.3 DERs

The reference AGC signal was to be collectively tracked using DERs consisting of HVAC AHUs, BESS, V1G and V2G EVs, PV systems, and whole-building loads. Since PV systems and (non-AHU) building loads were not controllable, they participated in the test as passive DERs. Consequently, the active DERs were commanded to track a modified target signal derived by subtracting the net active power output of passive DERs from the reference AGC signal and applying appropriate scaling (cf. Section 6.3.2). Table 6.1 lists the typical net power capacity $\overline{p}_i - \underline{p}_i$ of the different active DER types.

Table 6.1. DER counts and characteristics for each test.

DER Type	AHU	V1G EV	V2G EV	BESS
# DERs for Test 0	7	4	5	1
# DERs for Test 1	34	29	5	1
# DERs for Test 2	34	17	6	1
Signal update times	1 min	5 min (Test 0 & 1), 1 min (Test 2)	1 sec	20 sec
Typical power rating per DER type	2 kW	3.3 kW (Test 0 & 1), 4.9 kW (Test 2)	5 kW	3 kW

The contribution of each active DER to the target signal was defined with respect to a baseline power, around which $[\underline{p}_i, \overline{p}_i]$ was centered, to enable tracking of both positive and negative ramps in the target signal. For DERs like V2G EVs and BESS, which were capable of power adjustments in both directions, the baseline was 0 kW. The baseline for V1G EVs was defined to be halfway between their allowed minimum and maximum charging rates, where the former was restricted by the SAE J1772 charging standard to 1.6 kW. Similarly, the baseline for

AHUs was defined to be half of their power draw when on. Further, since AHUs were limited to binary on-off operational states, the continuous and arbitrarily precise AHU setpoints obtained by solving (6.1) were rounded to the closest discrete setpoint obtained from a combination of on-off states before actuation.

AHU control was restricted, by UCSD Facilities Management, to specifying only DER setpoints and duration of actuation; since building automation controllers could not be modified, model-based designs were impossible. This was to avoid malfunctioning or disruptions to real physical infrastructure in the networked building management system that also controls lighting, security, and fire protection systems.

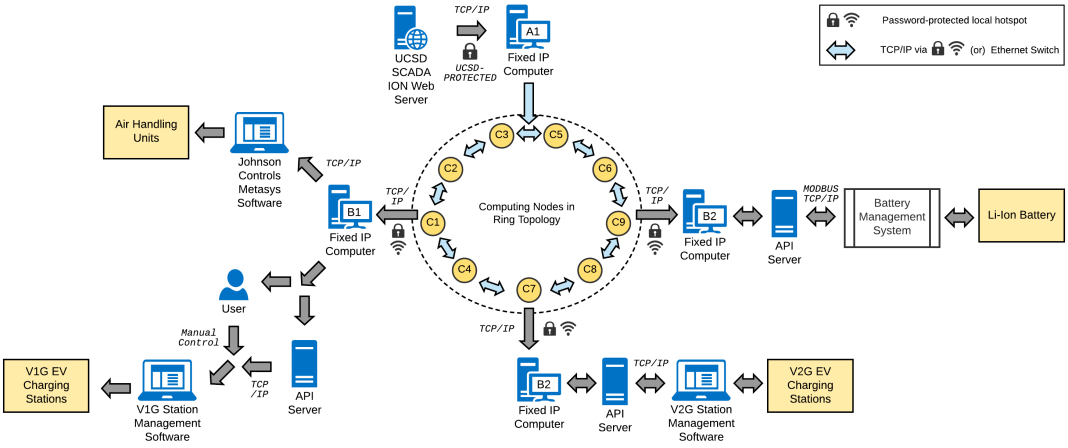


Figure 6.1. Communication architecture for computation and actuation of control policies.

6.3.4 Computing Setup

The DER active power setpoints were computed using a set of 9 Linux-based nodes, named C1-C9, that communicate with each other over an undirected ring topology, cf. Fig. 6.1. As one of the sparsest network topologies, where message passing occurs only between a small number of neighbors, the ring topology presents a challenging scenario for distributed control. Since there were more active DERs than computing nodes, the 9 nodes were mapped subjectively to the 69 active DERs such that nodes C1-C2 computed the actuation setpoints for the AHUs, C3 for V1G EVs, C4-C8 for V2G EVs and C9 for the BESS.

Each computing node generated actuation commands as CSV files containing the power setpoints for their respective group of DERs at a uniform update rate of 1 Hz. Preliminary testing revealed different response times across DER types, with AHUs and V1G EVs exhibiting slower response than other active DER types. DERs with response times greater than 1 s were subject to a stair-step control signal with a signal update time consistent with DER responsiveness and constant setpoints during intermediate time steps. Table 6.1 lists the signal update times for the different DER types.

6.3.5 Actuation Interfaces and Communication Framework

The actuation commands were issued using fixed IP computers through dedicated interfaces that varied by DER type as depicted in Fig. 6.1. The setpoints for AHUs were issued through a custom Visual Basic program that interfaced with the Johnson Control Metasys building automation software. The power rate of the BESS was set via API-based communication with a dedicated computer that controlled the battery inverter. The V1G and V2G EVs charging rates were adjusted through proprietary smart EV charging platforms of the charging station operators. EVs using ChargePoint® V1G stations were manually controlled via the load shedding feature of ChargePoint's station management software. The actuation of EVs using PowerFlex® V1G chargers and Nuvve® V2G chargers was automated and commands were issued via API-based communication.

6.3.6 Power Measurements

The active power of all DERs was metered at a 1 Hz frequency. The power outputs of PV systems and building loads were obtained prior to the test from their respective ION meters by logging data from the UCSD ION Supervisory Control and Data Acquisition (SCADA) system. A moving average filter with a 20 s time horizon was used to remove noise from the measured data for these passive DERs. V2G EVs and BESS power data were acquired using the same interfaces that were used for their actuation, which logged data from dedicated power meters.

Since neither AHUs nor the ChargePoint V1G EVs had dedicated meters, they were monitored via their respective building ION meters by subtracting a baseline building load from the building meter power output. Assuming constant baseline building load, any change in the meter outputs can be attributed to the actuation of AHUs and V1G EVs. This assumption is justifiable considering the tests were conducted at 0400 PT to 0600 PT on a weekend, when building occupancy was likely zero and building load remained largely unchanged. Noise in the ION meter outputs observed as frequent 15 - 30 kW spikes in the measured data for AHUs (Fig. 6.2) and ChargePoint V1G EVs was treated by removing outliers and passing the resulting signal through a 4 s horizon moving average filter. Here, outliers refer to points that change in excess of 50% of the mean of the 40 min signal in a 1 s interval.

6.3.7 Performance Metrics

The performance of the distributed implementation (cyber-layer) was measured by the normalized mean-squared-error (MSE) between the distributed and true (i.e. exact) centralized optimization solutions. The true solutions were computed for each instance of (6.1) using a centralized CVX solver in MATLAB [50]. The MSE was normalized by dividing by the mean of the squares of the true solutions.

The tracking performance of the DERs was evaluated through (i) the root-mean-squared-error (RMSE) in tracking

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{T}(P_t^{\text{prov}} - P_t^{\text{tar}})^2}{\sum_{t=1}^{T}(P_t^{\text{tar}})^2}}, \tag{6.3}$$

where P_t^{prov} is the total power that was provided (measured), and P_t^{tar} is the target (commanded) regulation power at time step $t \in \{1, \ldots, T = 2401\}$; and (ii) the tracking delay, computed as the time shift of the measured signal which yields the lowest RMSE between the commanded and measured signals.

The PJM Performance Score S following [4, Section 4.5.6] was computed as a test for eligibility to participate in the ancillary services market, and is given by the mean of a Correlation

Score S_c, Delay Score S_d, and Precision Score S_p:

$$S_c = \frac{1}{T-1}\sum_{t=1}^{T}\frac{(P_t^{\text{prov}}-\mu^{\text{prov}})(P_t^{\text{tar}}-\mu^{\text{tar}})}{\sigma^{\text{prov}}\sigma^{\text{tar}}},$$
$$S_d = \left|\frac{\delta - 5\text{ min}}{5\text{ min}}\right|, \quad S_p = 1 - \frac{1}{T}\sum_{t=1}^{T}\left|\frac{P_t^{\text{prov}}-P_t^{\text{tar}}}{\mu^{\text{tar}}}\right|,$$
$$S = 1/3(S_c + S_d + S_p),$$

where P_t^{prov} and P_t^{tar} are as in (6.3), $\mu^{\text{prov}}, \mu^{\text{tar}}$ and $\sigma^{\text{prov}}, \sigma^{\text{tar}}$ denote their respective means and standard deviations, and δ is the corresponding maximum delay in DER response for when S_c was maximized. A performance score of at least 0.75 is required for participating in the PJM ancillary services market.

6.4 Test Scenarios and Results

In this section, we describe the test scenarios carried out on the UCSD microgrid and present their outcome, elaborating on the challenges we faced and the differences across the tests.

6.4.1 Test Scenarios

Commonalities

A series of three tests were conducted on December 12, 2018 (Test 0), April 14, 2019 (Test 1) and December 17, 2019 (Test 2). All three tests involved a 40 min preparatory run followed by a 40 min final test. Table 6.1 lists the number and type of DERs used in each test. All tests were carried out during non-operational hours (between 0400 PT and 0540 PT) to maximize fleet EV availability and to avoid potential disruptions to building occupants. Day-time PV output data from February 24, 2019 was used as a proxy for an actual daytime PV signal.

Test 0

Test 0 was a preliminary calibration that used only a representative sample of 17 DERs. The purpose of Test 0 was to examine the response times and tracking behavior of every DER type and detect issues related to communication and actuation.

Test 1

Test 1 was identical to Test 0, but it used a larger population of 69 active DERs and 107 passive DERs.

a) *DERs.* The V1G and V2G population for Test 1 was composed of UCSD fleet EVs plugged in at ChargePoint and Nuvve charging stations, respectively. Since the ChargePoint V1G EVs were operated via manual input of DER setpoints (an interface to their API had not been developed yet), to avoid overloading the (human) operators, they were grouped into three groups and actuated in a staggered fashion such that each of the three groups maintained a signal update time of 5 min but were commanded 1 min apart from each other.

b) *Computing Setup.* For both Tests 0 and 1, 9 laptops running a Robotic Operating System (ROS) communicated via local Wi-Fi hotspot to implement the distributed coordination algorithms and compute the DER setpoints. Given that the available power capacity of fast-responding DERs such as V2G and BESS was smaller than slow-responding DERs, the steep ramping demands of the target signal were met by upscaling the power of the fast responding DERs in solving for the contribution of individual DERs. Another option would have been to reduce the number of slow responding DERs, but the funding agency stipulated prioritizing the number and types of heterogeneous DERs over accuracy in signal tracking. A real DER aggregator would instead require a more balanced capacity of slow and fast DERs to ensure feasibility of tracking these ramp features.

Test 2

Test 2 also used the entire population of DERs but substituted the cumbersome V1G population with more capable V1G chargers and used a new distributed computing setup and method of actuation based on lessons learned from Test 1.

a) *DERs.* The V1G EVs used in Test 1 performed poorly owing to an unreliable actuation-interface that experienced seemingly random stalling and lacked automated control capabilities. Therefore, 17 PowerFlex V1G charging stations at one location replaced the distributed 29 V1G charging stations in Test 1. Since the PowerFlex interface did not permit actuating individual stations, the 17 charging stations participated in the test as a single aggregate DER. The 0930 – 1010 PT timing of the V1G EV part of the test coincided with the start of the workday and a V1G EV population that had only recently plugged in and therefore had ample remaining charging capacity. The EVs were contributed by UCSD employees and visitors randomly plugging in at the PowerFlex charging stations just before the start of the trial. An aggregate signal of 15 kW to 19 kW was distributed equally amongst the 17 EVs.

In addition to the new V1G EVs, the V2G population in Test 2 was replaced with a different set of Nuvve chargers to resolve a tracking/noise issue during discharge-to-grid observed in Test 1 and expanded to include an additional charger, amounting to a total of six V2Gs charging six 5 kW EVs.

The order of AHU actuation was modified to allow for device settling time and prevent interference. In particular, in Tests 0 and 1, individual AHUs were ordered and actuated using a protocol that was not cognizant of settling times or building groupings, while the protocol was revised in Test 2 to systematically command the entire population of AHUs in a manner which maximized time between consecutive actuations for an individual unit.

b) *Computing Setup.* Test 2 featured a fully distributed architecture, unlike the ROS-based semi-centralized computing setup in Test 1. The new distributed setup consisted of a network of Raspberry Pi's that asynchronously communicated with each other via an ethernet

switch. In addition, a modified synchronization technique was implemented in the software which improved the fidelity and robustness of message-passing. This upgraded message-passing framework and synchronization technique for both software and hardware resulted in significantly faster communication between nodes.

c) *Two-Stage Actuation.* Test 2 also featured a two-stage approach of actuation that was a result of the DER tracking behavior in Test 1. Some DERs, such as BESS, V1G EVs and V2G EVs, tracked quickly and accurately, whereas others, such as AHUs, tracked poorly. The overall tracking performance in Test 2 was improved by using "well-behaved" DERs to compensate for AHU tracking errors by incorporating the error signal from actuating AHUs in Stage 1 to the cumulative target signal for BESS, V1G EVs and V2G EVs in Stage 2. Although synchronous actuation of all participating DERs is preferred in practice, the two-stage approach highlights the significance of systematic characterization of DERs in minimizing ACE.

6.4.2 Test Results

Distributed Optimization/Cyber-Layer Results

In Table 6.2, we present MSE results of our 1 s real-time Raspberry-pi distributed optimization solutions (the "cyber-layer" of the system).

Table 6.2. Normalized mean-squared-error of distributed solutions obtained from real-time 1-second intervals compared to centralized solver solution for Test 2 (Section 6.3.7)

DER Type	RC	PD	DANA	all
AHU	0	1.4×10^{-7}	2.8×10^{-9}	4.6×10^{-8}
V1G EVs	0	7.0×10^{-8}	1.7×10^{-9}	2.3×10^{-8}
V2G EVs	0	6.6×10^{-5}	5.0×10^{-7}	2.1×10^{-5}
BESS	0	2.0×10^{-6}	9.1×10^{-8}	6.5×10^{-7}
Total	0	1.8×10^{-5}	1.1×10^{-7}	4.9×10^{-6}

RC converged to the exact solution in all instances. This is unsurprising, as the RC problem formulation does not account for individual DER costs and thus, is a much simpler problem with a closed-form solution. For PD and DANA, we obtained excellent convergence,

with errors on the order of 0.001% in the worst cases. In general, DANA tended to converge faster than PD and obtained more accurate solutions. For our application with 1 s real-time windows, accuracy and convergence differences did not affect the physical layer results in any tangible way, but applications with more stringent accuracy or speed requirements may benefit from using a faster algorithm like DANA. The differences between DER populations can be largely attributed to the faster time scale of the V2G EVs (and to a lesser extent the BESS), see Table 6.1. Since the V2G EVs were responsible for the high-frequency component of P_{ref}, the solver was required to converge to new solutions at every time step, which induced more error compared to the slow V1G EVs and AHUs with relatively static solutions.

Physical-Layer Test Results

We now present the results of the tracking performance pertaining to the physical-layer of the experiment. We provide only some selective plots for Test 0 and Test 1 in Fig. 6.2, and a complete set of plots for each Test 2 DER population in Fig. 6.3. Error and tracking delay data defined in Section 6.3.7 is given in Table 6.3 for Test 1 and Test 2. Data for Test 0 is omitted due to its preliminary nature. The optimal shift described in Section 6.3.7 is applied to each time series and hence some areas in plots may appear like the provided signal anticipated the target.

Signal tracking accuracy in Test 0 was generally poor despite the small number of DERs employed, largely due to inexperience in actuating the AHUs and V1Gs. In particular, Fig. 6.2 reveals some oscillations in the AHU response. It is overall difficult to determine if even large-feature, low-frequency components of the signal were tracked. Further, data gathering for V1Gs and AHUs was done via noisy and unreliable building ION meters, which motivated the need for outlier treatment (Section 6.3.6) in Tests 1 and 2, and resulted in the smoother and better tracking signal in the top plot of Fig. 6.3.

Test 1 yielded a 111% rMSE for AHUs. We speculate that the small 4 s delay is not representative of the actual AHU delay due to random correlations dominating the time shift for this large error. This is confirmed by a much better AHU response in Test 2 with rMSE 12%,

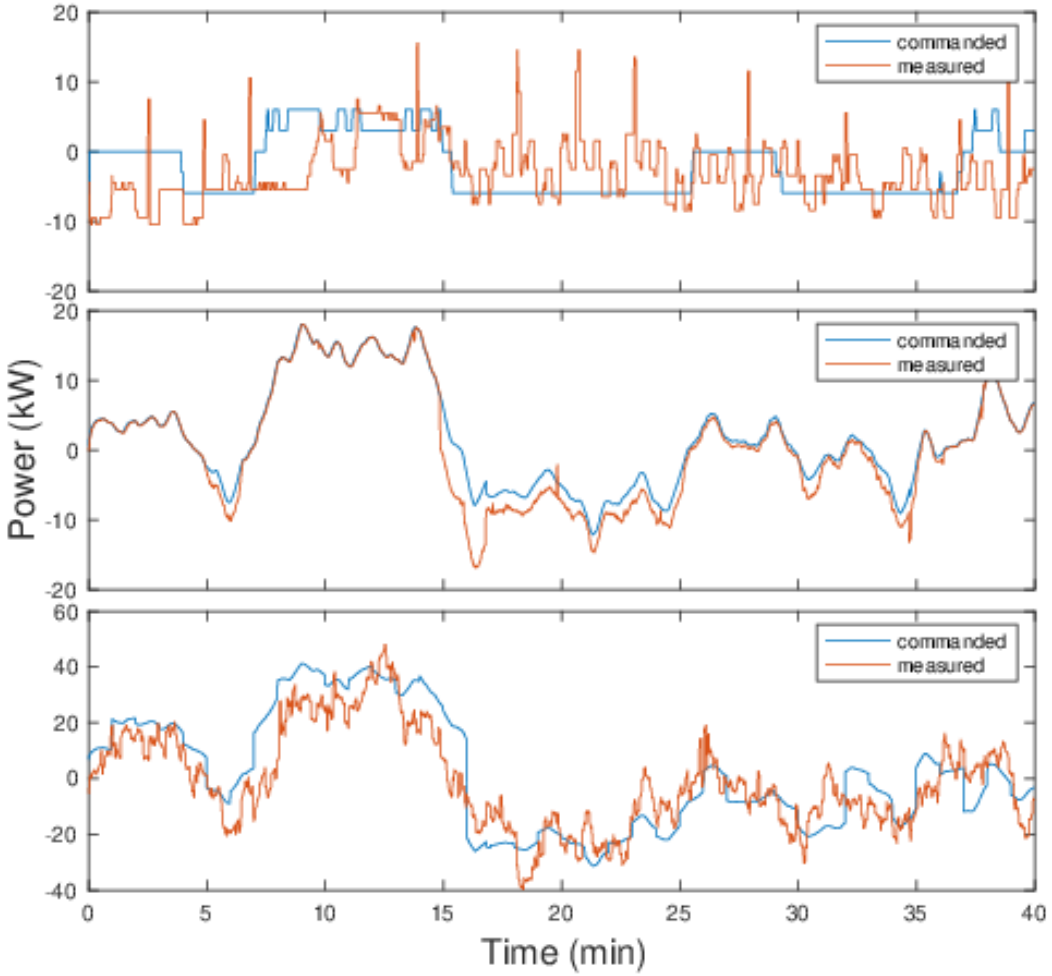


Figure 6.2. Top: AHU response in Test 0. **Middle:** V2G response in Test 1. **Bottom:** Total response in Test 1.

where a 105 s delay is more likely to be representative of the true AHU actuation delay. Given the poor visibility into AHU and V1G controllers explained in Section 6.4.1, it is challenging to identify the source of the poor tracking behavior. We speculate that DER metering at the building level rather than the DER level was a major source of error for AHU and V1G in Test 1. This was largely resolved in Test 2 by utilizing a different population of V1Gs with dedicated meters and by modifying the actuation scheme for AHUs to be less susceptible to metering errors as described in Section 6.4.1. Additionally, the actuation-interface stalling for V1G EVs, described in Section 6.4.1, was dominant in Test 1, resulting in the poor tracking for V1Gs. Actuating-interface issues were resolved in Test 2 by utilizing an automated control scheme for

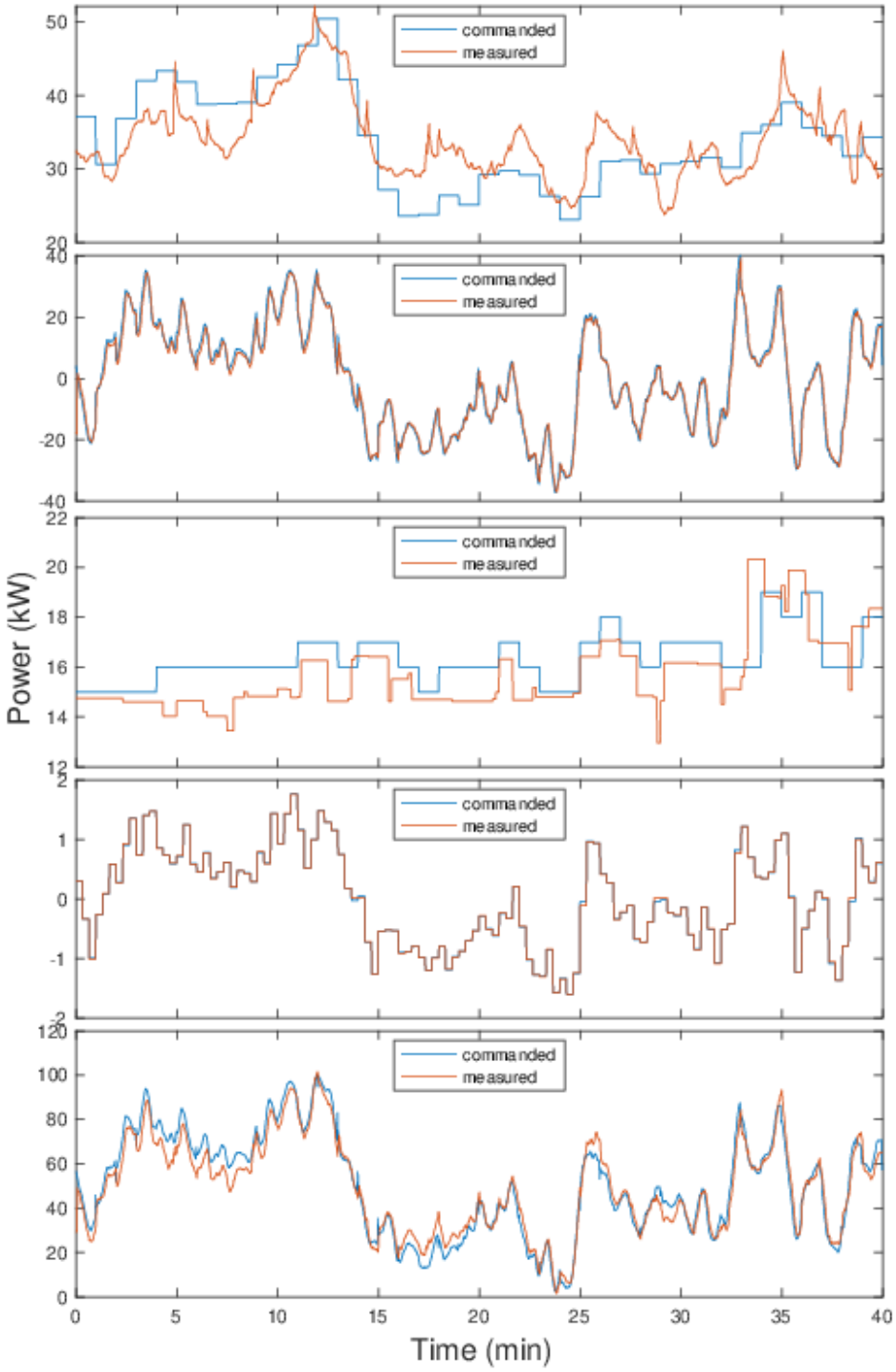


Figure 6.3. From **top to bottom**, AHU, V2G EVs, V1G EVs, BESS, and total responses in Test 2.

the V1Gs, which led to significantly lower error.

The BESS emerged as the star performer achieving very accurate tracking across all tests

with no delay. The V2G EVs also performed relatively well aside from a signal overshoot issue observed during the discharge cycle in Test 1 seen in Fig. 6.2. The issue was resolved in Test 2 by using V2G EV charging stations from a different manufacturer (Princeton Power), as described in Section 6.4.1. The V2G charging stations deployed for these tests were pre-commercial or early commercial models that had a few operating issues, such as the overshoot issue during Test 1.

The inability of the AHUs to respond to steep, short ramps (Fig. 6.3) could be due to slow start-up sequences programmed into the building automation controllers to increase device longevity or due to transients associated with driving their AC induction electric motors. Tackling this would require dynamic models and parameter identification of signal response and delay. With the new V1G EV population in Test 2, tracking delay reduced from 40 s to 10 s and the tracking accuracy improved significantly. The 1 kW bias seen in Fig. 6.3 is likely due to rounding errors arising from the inability of PowerFlex charging stations to accept non-integer setpoints.

The superior performance of the BESS and V2Gs motivated the two-stage actuation scheme described in Section 6.4.1, which contributed to reducing the total RMSE from 50% in Test 1 to 10% in Test 2 (compare the bottom plots of Figs 6.2 and 6.3). The two-stage approach allows a sufficiently large proportion of accurately tracking DERs to compensate for the errors of the first stage, where tracking is worse. In this way, poorly-tracking DERs, such as AHUs, can still contribute by loosely tracking some large-feature, low-frequency components of the target signal. The low-frequency contribution reduces the required total capacity of the strongly-performing DERs in the second stage leading to more fine-tuned signal tracking in aggregation. Some recommended rules of thumb for two-stage approach are: (i) Total capacity of first-stage DERs is less than or equal to total capacity of second-stage DERs. (ii) DERs in the first stage are capable of tracking with $< 50\%$ rMSE. (iii) DER cost functions are such that the deviation from the baseline is lower cost for first-stage DERs than for second-stage. (iii) allocates a significant portion of the target signal initially to first-stage DERs, freeing up DER capacity in the second-stage for error compensation.

Table 6.3. Left: Relative root mean-squared-error of tracking error by DER type. **Right:** Delay (optimal time-shift) of DER responses in sec.

DER Type	Test 1	Test 2
AHU	1.11	0.12
V1G EVs	0.68	0.077
V2G EVs	0.30	0.060
BESS	0.054	0.018
Total	0.50	0.097

DER Type	Test 1	Test 2
AHU	4	105
V1G EVs	40	10
V2G EVs	5	3
BESS	0	0
Total	N/A	N/A

Economic Benefit Analysis

Here, we evaluate the economic benefit of the proposed test system, which is vital for wider scale adoption of DERs as a frequency regulation resource in real electricity markets. To this end, we take an approach similar to [66] to first demonstrate that the testbed is eligible to participate in the PJM ancillary services market. Following the PJM Manual 12 [4] (Section 6.3.7), we compute a Correlation Score $S_c = 0.98$, Delay Score $S_d = 0.65$, and Precision Score $S_p = 0.91$ from data for Test 2, and obtain a Performance Score $S = 0.85 \geq 0.75$, which confirms the eligibility to participate in the PJM ancillary service market.

Next, we compute the estimated annual revenue assuming that the resources are available throughout the day. Using PJM's capability clearing price data[2] with our total (active) DER capacity of 184 kW and performance score of 0.85, the revenue for this population of resources (cf. [2, Section 4]) would be \$135 for July 9, 2020. This gives an estimated amount of \$49,210 as the total annual revenue. Note that the 184 kW DER capacity employed in this work represents less than 5% of the total DER capacity and less than 0.5% of the total capacity of the UCSD microgrid, cf. [104]. As such, the revenue would significantly increase if more microgrid resources are utilized for regulation, even with reduced availability.

[2] https://dataminer2.pjm.com/feed/reg_prices/definition

Appendix: Distributed Coordination Algorithms

In this section we describe the algorithms used in our distributed computing platform to solve (6.1).

Ratio-Consensus (RC): The ratio-consensus of [**?**] computes equitable contributions from all DERs without DER-specific cost functions (or constant DER costs). The ratio-consensus algorithm for providing P_{ref} is given by

$$y_i[k+1] = \sum_{j \in \mathscr{N}_i} \frac{1}{|\mathscr{N}_i|} y_j[k], \quad z_i[k+1] = \sum_{j \in \mathscr{N}_i} \frac{1}{|\mathscr{N}_i|} z_j[k],$$

$$y_i[0] = \begin{cases} \frac{P_{\text{ref}}}{|\mathbb{I}|} - \underline{p}_i, & i \in \mathbb{I}, \\ -\underline{p}_i, & i \notin \mathbb{I}, \end{cases} \qquad z_i[0] = \overline{p}_i - \underline{p}_i,$$

where, k is the iteration number, y_i and z_i are two auxiliary variables maintained by each agent, $\mathscr{N}_i$ denotes the neighboring DERs of DER i, and $\underline{p}_i$ and $\overline{p}_i$ are the minimum and maximum power level for DER i from the problem formulation in Section 6.3.1. $\mathbb{I}$ denotes the subset of DERs which know the value of the reference signal. One can see that

$$\begin{aligned} p_i^\star &= \underline{p}_i + \lim_{k \to \infty} y_i[k]/z_i[k](\overline{p}_i - \underline{p}_i) \\ &= \underline{p}_i + \frac{P_{\text{ref}} - \sum_i \underline{p}_i}{\sum_i \overline{p}_i - \underline{p}_i}(\overline{p}_i - \underline{p}_i), \end{aligned}$$

where $p_i^\star$ is then the power assignment for DER i.

Primal-Dual (PD): Both this dynamics and DANA (described next) take into account the cost functions of the DER types when computing the power setpoints, i.e., f_i are nonconstant. These functions are modeled as quadratics, which is a common choice in generator dispatch [5]. The dynamics is based on the discretization of the primal-dual dynamics [33] for the augmented Lagrangian of the equivalent reformulated problem, see [31], and it has a linear

rate of convergence to the optimizer. The algorithm is given by

$$\begin{bmatrix} \dot{p}_i \\ \dot{y}_i \\ \dot{\lambda}_i \end{bmatrix} = \begin{bmatrix} -\left(f_i'(p_i) + \lambda_i + p_i \sum_{j\in\mathcal{N}_i} L_{ij} y_j - P_{\text{ref}}/n\right) \\ -\left(\sum_{j\in\mathcal{N}_i} L_{ij}(\lambda_j + x_j - P_{\text{ref}}/n) + \sum_{j\in\mathcal{N}_i^2} L_{ij}^2 y_j\right) \\ p_i + \sum_{j\in\mathcal{N}_i} L_{ij} y_j - P_{\text{ref}}/n \end{bmatrix},$$

where, L is the Laplacian matrix of the communication graph (see [18]), y_i is an auxiliary variable, and λ_i is the dual variable associated with agent i. The update step is followed by a projection of the primal variable p_i onto the box constrained local feasible set. These dynamics converge from any set of initial conditions. Since this algorithm evolves in continuous time, we use an Euler discretization with fixed step-size to implement it in discrete time.

Distributed Approximate Newton Algorithm (DANA): The Distributed Approximate Newton Algorithm (DANA) of [6] has an improved rate of convergence compared to PD. This algorithm solves the equivalent reformulated problem

$$\begin{aligned} \min_{z\in\mathbb{R}^n} f(p^0 + Lz) &= \sum_{i=1}^{n} f_i(p_i^0 + L_i z), \\ \text{subject to } \underline{p} - p^0 - Lz &\leq \mathbf{0}_n, \\ p^0 + Lz - \overline{p} &\leq \mathbf{0}_n, \end{aligned} \tag{6.4}$$

where p^0 is a vector of initial power levels of all the DERs with $\sum_i p_i^0 = P_{\text{ref}}$, and z is the new variable of optimization. The continuous time dynamics are given by

$$\begin{aligned} \dot{z} &= -A_q \nabla_z \mathscr{L}(z, \lambda), \\ \dot{\lambda} &= [\nabla_\lambda \mathscr{L}(z, \lambda)]_\lambda^+, \end{aligned}$$

where $\mathscr{L}$ is the Lagrangian of (6.4) and A_q is a positive definite weighting on the gradient direction which provides distributed second-order information. For brevity, we do not provide the full details of the algorithm here, which can instead be found in [6]. The cost functions are

again taken to be quadratic with strictly positive leading coefficients.

Acknowledgements

The material in this chapter, in full, is under revision for publication in IEEE Transactions on Smart Grid. It may appear as *Frequency Regulation with Heterogeneous Energy Resources: A Realization using Distributed Control*, T. Anderson, M. Muralidharan, P. Srivastava, H.V. Haghi, J. Cortés, J. Kleissl, S. Martínez and B. Washom. The dissertation author was one of three primary investigators and authors of this paper.

We would like to thank numerous people in the UCSD community and beyond for their generous contributions of time and resources to enable such an ambitious project to come together. We extend thanks to: (i) Aaron Ma and Jia (Jimmy) Qiu for assisting with hardware setup and software development for the distributed computation systems; (ii) Kevin Norris for coordinating the fleet vehicles; (iii) Abdulkarim Alamad for overseeing V1G drivers in Test 2; (iv) Kelsey Johnson for managing the Nuvve contributions; (v) Ted Lee, Patrick Kelly, and Steven Low for managing the PowerFlex contribution; (vi) Marco Arciniega, Martin Greenawalt, James Gunn, Josh Kavanagh, Jennifer Rodgers, Patricia Roman and Lashon Smith from UCSD parking for reserving EV charging station parking spaces; (vii) Charles Bryant, Harley Crace, John Denhart, Nirav Desai, John Dilliott, Mark Gaus, Martin Greenawalt, Gerald Hernandez, Brandon Hirsch, Mark Jurgens, Josh Kavanagh, Jose Moret, Chuck Morgan, Curt Lutz, Jose Moret, Cynthia Wade, Raymond Wampler and Ed Webb for contributing their EVs in Test 1; (viii) Adrian Armenta, Adrian Gutierrez and Minghua Ong who helped with ChargePoint manual control; (ix) Bob Caldwell (Centaurus Prime), Gregory Collins, Charles Bryant, and Robert Austin for programming and enabling the AHU control; (x) Gary Matthews and John Dilliott for permitting the experimentation on "live" buildings and vehicles; and (xi) Antoni Tong and Cristian Cortes-Aguirre for supplying the BESS. Finally, we would like to extend a sincere thanks to the ARPA-e NODES program for its financial support and to its leadership, including

Sonja Glavaski, Mario Garcia-Sanz, and Mirjana Marden, for their vision and push for the development of large-scale power-in-the-loop testing environments.

Chapter 7

Conclusion

In this thesis, we studied a class of separable resource allocation problems, and we developed three types of Newton-like algorithms to approach three different scenarios of the resource allocation. Each algorithm was theoretically analyzed and rigorously shown to satisfy some convergence criteria, and the efficacy of each was validated in simulation with comparisons to relevant alternatives available in literature. We now summarize chapter-by-chapter the more specific conclusions that can be drawn and suggestions for future work.

In Chapter 2, motivated by economic dispatch problems and separable resource allocation problems in general, this work proposed a class of novel DISTRIBUTED APPROX-NEWTON algorithms. We first posed the topology design proplem and provided an effective method for designing communication weightings. The weight design we propose is more cognizant of the problem geometry, and it outperforms the current literature on network weight design even when applied to a gradient-like method. Our contribution on the second-order weight design approach is novel but is limited in scope to the given problem formulation. Distributed second-order methods are quite immature in the present literature, so an emphasis of future work is to generalize this weight design notion to a broader class of problems. Ongoing work also includes generalizing the cost functions for box-constrained settings and discretizing the continuous-time algorithm. In addition, we aim to develop distributed Newton-like methods suited to handle more general constraints and design for robustness under uncertain parameters or

lossy communications. Another point of interest is to further study methods for solving bilinear problems and apply these to weight design within the Newton framework.

Chapter 3 studied a nested, distributed stochastic optimization problem and applied a Distributed Stochastic Cubic-Regularized Newton (DiSCRN) algorithm to solve it. In order to compute the DiSCRN update, a batch of approximate solutions to realizations of the inner-problem are obtained, and we developed a locally-checkable stopping criterion to certify sufficient accuracy of these solutions. The accuracy parameter is directly leveraged in the analysis of the outer-problem, and simulations justify both faster and more robust convergence properties than that of comparable gradient-like and Newton-like approaches. Future work involves developing and analyzing a saddle-point dynamics approach for solving ¶3 (extending the work of [24]), extending the analysis to accommodate small disagreements in the agent states x_i, and exploring adaptive batch size techniques.

In Chapter 4, we posed an optimal generator dispatch problem for settings in which the agents are generators with binary controls. We first showed that the centralized problem is amenable to solution via a Centralized NEWTON-LIKE NEURAL NETWORK approach and proved convergence to a local minimizer with probability one under light assumptions. Next, we developed an approach to make the dynamics computable in a distributed setting in which agents exchange messages with their two-hop neighbors in a communication graph. The methods scale and perform well compared to standard greedy and SDP-relaxation approaches, and the latter method enjoys the qualities of a distributed algorithm, unlike previous approaches. Future research directions include application of the methods to a broader class of problems which may include additional cost terms or constraints and a deeper analysis of the Deterministic Annealing variant as it pertains to the online adjustment of the learning-rate T/τ.

Chapter 5 introduced three related problems motivated by studying the algebraic connectivity of a graph by adding edges to an initial topology or protecting edges under the case of a disturbance or attack on the network. We developed a novel SDP relaxation to address the NP-hardness of the design and demonstrated in simulation that it is superior to existing methods

which are greedy and cannot accommodate general constraints. In addition, we studied the dynamics of the game that may be played between a network coordinator and strategic attacker. We developed the notion of an optimal preventive solution for the coordinator and proposed effective heuristics to find such a solution guided by characterizations of the solutions to the attacker's problem. Future work includes characterizing the performance of our SDP relaxation and developing an algorithm which provably converges to the optimal preventive strategy.

Finally, in Chapter 6 we presented one of the first real-world demonstrations of secondary frequency response in a distribution grid using up to 176 heterogeneous DERs. The DERs include AHUs, V1G and V2G EVs, a BESS, and passive building loads and PV generators. The computation setup utilizes state-of-the-art distributed algorithms to find the solution of a power allocation problem. We show that the real-time distributed solutions are close to the true centralized solution in an MSE sense. Tests with real, controllable DERs at power closely track the given active-power reference signal in aggregation. These tests highlight the importance of dedicated and noise-free measurement sensors and a well-understood and reliable DER control interface for precise signal tracking. Further, our economic benefit analysis shows a potential annual revenue of $49K for the chosen DER population. As is already recognized by the power systems community and federal funding agencies such as ARPA-e and NSF, large-scale power-in-the-loop testing is needed for transitioning distributed technologies to real distribution systems.

We hope that the work of this thesis spurs further study, testing, and ultimately widespread adoption of distributed algorithms by relevant players in industry, particularly in the renewable energy sector. Returning to the philosophical motivation of the Introduction, it is paramount to anticipate and resolve the issues of scale that are emerging as a result of computing systems transitioning from the "single-cellar" to the "multi-cellular" model. To this end, rigorous theory must continue to be developed in order to deeply understand distributed intelligence systems and to ensure they continue to improve quality of life and serve humanity.

Bibliography

[1] CAISO business practice manual for market operation. https://bpmcm.caiso.com/Pages/BPMDetails.aspx?BPM=Market%20Operations, 2018. Version 57.

[2] PJM manual 28: Operating agreement accounting, December 3 2019. Revision 83.

[3] PJM markets and operations: Anciliary services. https://www.pjm.com/markets-and-operations/ancillary-services.aspx, 2019. RegD Normalized Signal Test (after 1.30.2017).

[4] PJM manual 12: Balancing operations, March 26 2020. Revision 40.

[5] G. Sheblé A. Wood, B. Wollenberg. *Power Generation, Operation, and Control*. John Wiley, 3 edition, 2012.

[6] T. Anderson, C.-Y. Chang, and S. Martínez. Distributed approximate Newton algorithms and weight design for constrained optimization. *Automatica*, 2019. https://doi.org/10.1016/j.automatica.2019.108538.

[7] T. Anderson and S. Martínez. Distributed stochastic nested optimization via cubic regularization. *ArXiv. Preprint arXiv:2008.13291*, 2020.

[8] K. Baker, X. Jin, , D. Vaidhynathan, W. Jones, D. Christensen, B. Sparn, J. Woods, H. Sorensen, and M. Lunacek. Frequency regulation services from connected residential devices: Short paper. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, pages 119–122, Palo Alto, CA, 2016.

[9] Saeid Bashash and Hosam K. Fathy. Optimizing demand response of plug-in hybrid electric vehicles using quadratic programming. In *American Control Conference*, pages 716–721, 2013.

[10] S. Bauk and Z. Avramović. Hopfield network in solving travelling salesman problem in navigation. In *Seminar on Neural Network Applications in Electrical Engineering*, pages 207–2010, 2002.

[11] D. P. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[12] S. Bhattacharya, A. Gupta, and Tamer Basar. Jamming in mobile networks: a game-theoretic approach. *Numerical Algebra, Optimization, and Control*, 3(1):1–30, 2013.

[13] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Conference on Computational Statistics*, pages 177–186, Paris, France, 2010.

[14] L. Bottou, F.E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[15] S. Boyd. Convex optimization of graph Laplacian eigenvalues. In *Proc. Int. Congress of Mathematicians*, volume 3, page 1311–1319, 2006.

[16] S. Boyd and L. Vandenberghe. Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization. In A. Paulraj, V. Roychowdhuri, and C. Schaper, editors, *Communications, Computation, Control and Signal Processing: A Tribute to Thomas Kailath*, chapter 15, pages 279–288. Kluwer Academic Publishers, 1997.

[17] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[18] F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks*. Applied Mathematics Series. Princeton University Press, 2009.

[19] CAISO. Pay for performance regulation: Draft final proposal addendum, February 22 2012.

[20] CAISO Business Practice Manual for Market Operation, 2017. Version 51. Available at https://bpmcm.caiso.com/Pages/BPMDetails.aspx?BPM=Market%20Operations.

[21] CAISO. Business practice manual for market operation, May 24 2018. Version 57.

[22] R. Carli and G. Notarstefano. Distributed partition-based optimization via dual decomposition. In *IEEE Int. Conf. on Decision and Control*, 2013.

[23] R. Carli, G. Notarstefano, L. Schenato, and D. Varagnolo. Analysis of Newton-Raphson consensus for multi-agent convex optimization under asynchronous and lossy communications. In *IEEE Int. Conf. on Decision and Control*, page 418–424, Osaka, Japan, 2015.

[24] Y. Carmon and J. Duchi. Gradient descent finds the cubic-regularized nonconvex newton step. *SIAM Journal on Optimization*, 29(3):2146–2178, 2019.

[25] C. Cartis, N. Gould, and P. Toint. Adaptive cubic regularisation methods for unconstrained optimization. part I: motivation, convergence and numerical results. *Mathematical Programming*, 127:245–295, 2009.

[26] C. Cartis, N. Gould, and P. Toint. Adaptive cubic regularisation methods for unconstrained optimization. part II: worst-case function- and derivative-evaluation complexity. *Mathematical Programming*, 130:295–319, 2010.

[27] P. Chardaire and A. Sutter. A decomposition method for quadratic zero-one programming. *Management Science*, 41(4):704–712, 1995.

[28] X. Chen, B. Jiang, T. Lin, and S. Zhang. On adaptive cubic regularized Newton's methods for convex optimization via random sampling. *preprint arXiv:1802.05426*, 2018.

[29] M. Cheng, J. Wu, S. J. Galsworthy, C. E. Ugalde-Loo, N. Gargov, W. W. Hung, and N. Jenkins. Power system frequency response from the control of bitumen tanks. *IEEE Transactions on Power Systems*, 31(3):1769–1778, 2016.

[30] S. Cherian and P. Asmus. Liberating microgrids (and all DER): Aligning customer needs with solutions provider offerings. White Paper, 2016.

[31] A. Cherukuri and J. Cortés. Distributed algorithms for convex network optimization under non-sparse equality constraints. In *Allerton Conf. on Communications, Control and Computing*, pages 452–459, Monticello, IL, September 2016.

[32] A. Cherukuri and J. Cortés. Initialization-free distributed coordination for economic dispatch under varying loads and generator commitment. *Automatica*, 74:183–193, 2016.

[33] A. Cherukuri, B. Gharesifard, and J. Cortés. Saddle-point dynamics: conditions for asymptotic stability of saddle points. *SIAM Journal on Control and Optimization*, 55(1):486–511, 2017.

[34] A. Cherukuri, E. Mallada, S. H. Low, and J. Cortés. The role of convexity in saddle-point dynamics: Lyapunov function and robustness. *IEEE Transactions on Automatic Control*, 63(8):2449–2464, 2018.

[35] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3 edition, 2009.

[36] Y. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Int. Conf. on Neural Information Processing Systems*, pages 2933–2941, 2014.

[37] N. de Abreu. Old and new results on algebraic connectivity of graphs. *Linear Algebra and Its Applications*, 423:53–73, 2006.

[38] X. Ding and T. Jiang. Old and new results on algebraic connectivity of graphs. *The Annals of Applied Probability*, 20(6):2086–2117, 2010.

[39] T. Doan and C. Beck. Distributed Lagrangian methods for network resource allocation. In *IEEE Conf. on Control Technology and Applications*, 2017.

[40] S. Du, C. Jin, J. Lee, M. Jordan, B. Poczos, and A. Singh. Gradient descent can take exponential time to escape saddle points. In *Int. Conf. on Neural Information Processing Systems*, pages 1067–1077, Long Beach, CA, USA, 2017.

[41] L. Fabietti, T. T. Gorecki, F. A. Qureshi, A. Bitlislioğlu, I. Lymperopoulos, and C. N. Jones. Experimental implementation of frequency regulation services using commercial buildings. *IEEE Transactions on Smart Grid*, 9(3):1657–1666, 2018.

[42] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.

[43] S. Friedberg, A. Insel, and L. Spence. *Linear Algebra*. Pearson, 4 edition, 2003.

[44] W. A. Gardner. Learning characteristics of stochastic-gradient-descent algorithms: A general study, analysis, and critique. *Signal Processing*, 6(2):113–133, 1984.

[45] A. Ghosh and S. Boyd. Growing well-connected graphs. In *IEEE Int. Conf. on Decision and Control*, page 6605–6611, San Diego, USA, 2006.

[46] P. Gill, W. Murray, and M. Wright. *Practical optimization*. Academic Press, 1981.

[47] C. D. Godsil and G. F. Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics*. Springer, New York, 2001.

[48] R. Goebel, R. G. Sanfelice, and A. Teel. Hybrid dynamical systems. *IEEE Control Systems Magazine*, 29(2):28–93, 2009.

[49] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the Association for Computing Machinery*, 42(6):1115–1145, 1995.

[50] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014.

[51] J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer, 1983.

[52] G. Gutin, A. Yeo, and A. Zverovich. Traveling salesman should not be greedy: Domination analysis of greedy-type heuristics for the TSP. *Discrete Applied Mathematics*, 117(1-3):81–86, 2002.

[53] S. Hassan-Moghaddam and M. Jovanovic. On the exponential convergence rate of proximal gradient flow algorithms. In *IEEE Int. Conf. on Decision and Control*, 2018.

[54] A. Hassibi, J. How, and S. Boyd. A path-following method for solving BMI problems in control. In *American Control Conference*, page 1385–1389, San Diego, CA, USA, 1999.

[55] X. He, X. Fang, and J. Yu. Distributed energy management strategy for reaching cost-driven optimal operation integrated with wind forecasting in multimicrogrids system. *IEEE Transactions on Systems, Man, & Cybernetics. Part A: Systems & Humans*, 49(8):1643–1651, 2019.

[56] X. He, J. Yu, T. Huang, and C. Li. Distributed power management for dynamic economic dispatch in the multimicrogrids environment. *IEEE Transactions on Control Systems Technology*, 27(4):1651–1658, 2019.

[57] J. Hopfield and D. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52(3):141–152, 1985.

[58] B. Huang, L. Liu, H. Zhang, Y. Li, and Q. Sun. Distributed optimal economic dispatch for microgrids considering communication delays. *IEEE Transactions on Systems, Man, & Cybernetics. Part A: Systems & Humans*, 49(8):1634–1642, 2019.

[59] D. Jakovetic, J. Xavier, and J. Moura. Fast distributed gradient methods. *IEEE Transactions on Automatic Control*, 59(5):1131–1146, 2014.

[60] B. Kamgar-Parsi and B. Kamgar-Parsi. Dynamical stability and parameter selection in neural optimization. In *Int. Joint Conf. on Neural Networks*, page 566–571, 1992.

[61] W. Kempton, V. Udo, K. Huber, K. Komara, S. Letendre, S. Baker, D. Brunner, and N. Pearre. A test of vehicle-to-grid (V2G) for energy storage and frequency regulation in the PJM system, 2008. Available at http://www1.udel.edu/V2G/resources/test-v2g-in-pjm-jan09.pdf.

[62] H. Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

[63] M. Kintner-Meyer. Regulatory policy and markets for energy storage in North America. *Proceedings of the IEEE*, 102(7):1065–1072, 2014.

[64] B. Kroposki. Basic research needs for autonomous energy grids-Summary report of the workshop on autonomous energy grids. Technical report, NREL, September 13-14 2017. NREL/TP-5D00-70428.

[65] D. Li, X. Sun, S. Gu, J. Gao, and C. Liu. Polynomially solvable cases of binary quadratic programs. In A. Chinchuluun, P. Pardalos, R. Enkhbat, and I. Tseveendorj, editors, *Optimization and Optimal Control*, pages 199–225. Springer, 2010.

[66] Y. Lin, P. Barooah, S. Meyn, and T. Middelkoop. Experimental evaluation of frequency regulation from commercial building HVAC systems. *IEEE Transactions on Smart Grid*, 6(2):776–783, 2015.

[67] J. Liu, W. Zhang, and Y. Liu. Primary frequency response from the control of led lighting loads in commercial buildings. *IEEE Transactions on Smart Grid*, 8(6):2880–2889, 2017.

[68] B. Lundstrom, S. Patel, S. Attree, and M. V. Salapaka. Fast primary frequency response using coordinated DER and flexible loads: Framework and residential-scale demonstration. In *2018 IEEE Power Energy Society General Meeting*, pages 1–5, Portland, OR, August 2018.

[69] Z. Q. Luo, W. K. Ma, A. So, Y. Ye, and S. Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine*, 27(3):20–34, 2010.

[70] E. Mallada, C. Zhao, and S. Low. Optimal load-side control for frequency regulation in smart grids. *IEEE Transactions on Automatic Control*, 62(12):6294–6309, 2017.

[71] J. Mandziuk. Solving the travelling salesman problem with a Hopfield-type neural network. *Demonstratio Mathematica*, 29(1):219–231, 1996.

[72] R. Merris. Laplacian graph eigenvectors. *Linear Algebra and Its Applications*, 278(1–3):221–236, 1998.

[73] B. Mityagin. The zero set of a real analytic function. *arXiv:1512.07276v1*, 2015.

[74] A. Mokhtari, Q. Ling, and A. Ribeiro. An approximate Newton method for distributed optimization. *IEEE Transactions on Signal Processing*, 65(1):146–161, 2017.

[75] M. Mozaffaripour and R. Tafazolli. Suboptimal search algorithm in conjunction with polynomial-expanded linear multiuser detector for FDD WCDMA mobile uplink. *IEEE Transactions on Vehicular Technology*, 56(6):3600–3606, 2007.

[76] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions-I. *Mathematical Programming*, 14(1):265–294, 1978.

[77] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[78] Y. Nesterov and B.T. Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108:177—-205, 2006.

[79] Z. A. Obaid, L. M. Cipcigan, L. Abrahim, and M. T. Muhssin. Frequency control of future power systems: reviewing and evaluating challenges and new control methods. *Journal of Modern Power Systems and Clean Energy*, 7(1):9–25, 2019.

[80] V. Pan and Z. Chen. The complexity of the matrix eigenproblem. In *ACM Symposium on Theory of Computing*, pages 507–516, 1999.

[81] P. Parrilo and S. Lall. Semidefinite programming relaxations and algebraic optimization in control. *European Journal of Control*, 9(2-3):307–321, 2003.

[82] S. Poljak, F. Rendl, and H. Wolkowicz. A recipe for semidefinite relaxation for (0,1)-quadratic programming. *Journal of Global Optimization*, 7(1):51–73, 1995.

[83] E. Ramírez-Llanos and S. Martínez. Distributed discrete-time optimization algorithms with application to resource allocation in epidemics control. *Optimal Control, Applications and Methods*, 2017. To appear. Available at the Wiley Online Library.

[84] K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of IEEE*, 86(11):2210–2239, 1998.

[85] S. Paternain, A. Mokhtari, and A. Ribeiro. A Newton-based method for nonconvex optimization with fast evasion of saddle points. *SIAM Journal on Optimization*, 29(1):343–368, 2019.

[86] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[87] SDGE. AC saver for business.

[88] S. Y. Shafi, M. Arcak, and L. E. Ghaoui. Designing node and edge weights of a graph to meet Laplacian eigenvalue constraints. In *Allerton Conf. on Communications, Control and Computing*, page 1016–1023, UIUC, Illinois, USA, 2010.

[89] M. Shamaiah, S. Banerjee, and H. Vikalo. Greedy sensor selection: Leveraging submodularity. In *IEEE Int. Conf. on Decision and Control*, pages 2572–2577, 2010.

[90] K. Smith. *Solving Combinatorial Optimization Problems Using Neural Networks*. PhD thesis, University of Melbourne, March 1996.

[91] P. Srivastava, C.-Y. Chang, and J. Cortés. Participation of microgrids in frequency regulation markets. In *American Control Conference*, pages 3834–3839, Milwaukee, WI, May 2018.

[92] G.W. Stewart. *Matrix Algorithms Volume 1: Basic Decompositions*. SIAM, 1998.

[93] M. Swierczynski, D. Stroe, A. Stan, R. Teodorescu, R. Lærke, and P. C. Kjær. Field tests experience from 1.6MW/400kWh Li-ion battery energy storage system providing primary frequency regulation service. In *IEEE PES ISGT Europe*, pages 1–5, 2013.

[94] N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. Jordan. Stochastic cubic regularization for fast nonconvex optimization. In *Int. Conf. on Neural Information Processing Systems*, pages 2904–2913, 2018.

[95] A. Tuckey, S. Zabihi, and S. Round. Decentralized control of a microgrid. In *European Conference on Power Electronics and Applications*, pages 1–10, Warsaw, Poland, September 2017.

[96] R. Tutunov, H. Bou-Ammar, and A. Jadbabaie. Distributed Newton method for large-scale consensus optimization. *IEEE Transactions on Automatic Control*, 64(10):3983–3994, 2019.

[97] C. Uribe and A. Jadbabaie. A distributed cubic-regularized Newton method for smooth convex optimization over networks. *preprint arXiv:2007.03562*, 2020.

[98] J. Urschel, J. Xu, X. Hu, and L. Zikatanov. A cascadic multigrid algorithm for computing the Fiedler vector of graph laplacians. *Journal of Computational Mathematics*, 33(2):209–226, 2015.

[99] J. VanAntwerp and R. Braatz. A tutorial on linear and bilinear matrix inequalities. *Journal of Process Control*, page 363–385, 2000.

[100] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[101] E. Vrettos, E. C. Kara, J. MacDonald, G. Andersson, and D. S. Callaway. Experimental demonstration of frequency regulation by commercial buildings—part II: Results and performance evaluation. *IEEE Transactions on Smart Grid*, 9(4):3224–3234, 2018.

[102] P. Wang, C. Shen, A. Hengel, and P. Torr. Large-scale binary quadratic optimization using semidefinite relaxation and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(3):470–485, 2017.

[103] X. Wang, S. Ma, D. Goldfarb, and W. Liu. Stochastic quasi-newton methods for nonconvex stochastic optimization. *SIAM Journal on Optimization*, 27(2):927–956, 2017.

[104] B. Washom, J. Dilliot, D. Weil, J. Kleissl, N. Balac, W. Torre, and C. Richter. Ivory tower of power: Microgrid implementation at the University of California, San Diego. *IEEE Power and Energy Magazine*, 11(4):28–32, 2013.

[105] E. Wei, A. Ozdaglar, and A. Jadbabaie. A distributed Newton method for network utility maximization, I: Algorithm. *IEEE Transactions on Automatic Control*, 58(9):2162–2175, 2013.

[106] E. Wei, A. Ozdaglar, and A. Jadbabaie. A distributed Newton method for network utility maximization, II: Convergence. *IEEE Transactions on Automatic Control*, 58(9):2176–2188, 2013.

[107] L. Xiao and S. Boyd. Optimal scaling of a gradient method for distributed resource allocation. *Journal of Optimization Theory & Applications*, 129(3):469–488, 2006.

[108] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S., Srinivasa, and R. Sukthankar. Decentralized estimation and control of graph connectivity for mobile sensor networks. *Automatica*, 46(2):390–396, 2010.

[109] Z. Yang, A. Bose, H. Zhong, N. Zhang, Q. Xia, and C. Kang. Optimal reactive power dispatch with accurately modeled discrete control devices: A successive linear approximation approach. *IEEE Transactions on Power Systems*, 32(3):2435–2444, 2016.

[110] P. Yi, Y. Hong, and L. Feng. Initialization-free distributed algorithms for optimal resource allocation with feasibility constraints and its application to economic dispatch of power systems. *Automatica*, 74:259–269, 2016.

[111] F. Yousefian, A. Nedić, and U. Shanbhag. Stochastic quasi-newton methods for non-strongly convex problems: Convergence and rate analysis. In *IEEE Int. Conf. on Decision and Control*, pages 4496–4503, 2016.

[112] F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato. Newton-Raphson consensus for distributed convex optimization. *IEEE Transactions on Automatic Control*, 61(4):994–1009, 2016.

[113] Jinshan Zeng and Wotao Yin. On nonconvex decentralized gradient descent. *IEEE Transactions on Signal Processing*, 66(11):2834–2848, 2018.

[114] F. Zhang. *The Schur complement and its applications*, volume 4. Springer, 2005.

[115] M. Zhu and S. Martínez. *Distributed Optimization-Based Control of Multi-Agent Networks in Complex Environments*. Springer-Briefs in Electrical and Computer Engineering. Springer, 2015.