Agent-Based Time Scaling in Cooperative Multi-Agent Systems

Dillon R. Foight

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2021

Reading Committee:

Mehran Mesbahi, Chair

Kristi Morgansen

Sam Burden

Program Authorized to Offer Degree:
Aeronautics & Astronautics

©Copyright 2021
Dillon R. Foight

University of Washington

Abstract

Agent-Based Time Scaling in Cooperative Multi-Agent Systems

Dillon R. Foight

Chair of the Supervisory Committee:
Professor Mehran Mesbahi
Aeronautics & Astronautics

One of the most basic, yet important, assumptions we tend to make when considering a set of dynamics is that it evolves with respect to some standard measure of time. This is not true in all cases, of course, and taming the two-time scale phenomenon exhibited by many physical systems has given control theorists powerful tools in model reduction and control synthesis. Today's world, however, is becoming dominated by a new layer of complexity - systems that we seek to control and automate are not composed of a single entity, but entire networks of possibly heterogeneous agents working in concert. Under this new paradigm, we need not only to concern ourselves with the control of individual agents, but questions of what role the network topology play in overall system behavior and how it informs control strategies also present themselves.

This dissertation presents research into the confluence of these two ideas. For multi-agent systems cooperating via the consensus protocol in order to support a global objective, we allow the agent dynamics to evolve on unique time scales. Relatedly, we also consider problems of networked agents operating with imperfect clocks which may bias measurements of relevant state quantities. Within this framework, we seek to explore the effects of agent-based time scales and unique clocks on traditional graph-theoretic interpretations of system performance and resilience. In many cases, we find that the presence of these non-ideal notions of time significantly impact the performance and system level behavior. To this

end, we attempt to quantify the errors associated with implementing consensus on clock-biased systems, as well as suggesting some possible strategies for error mitigation and clock synchronization.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

To restate a cliché in a context fitting of this document, no node is truly a singleton. My being able to undertake the research presented here is the result of literal decades of love and unflinching support from family and friends spread out over both time and geography. I am extremely privileged to have two sets of incredible parents who have never ceased to support my endeavors and endured my explanations of whatever new thing I had decided to pursue. Many of those pursuits took me quite far from home, but the distances have never attenuated the support and love that I felt from family. I have also always been extremely lucky to fall into great groups of people. The posse in high school, the Faraday Force in college, the Chandra crew and MIT Bio friends in Boston, the game night groups in Seattle, and the RAIN lab group...it continually perplexes me that so many amazing people will listen to my endless complaining, but I have been so very enriched by all of your friendships. Lastly, and undoubtedly most importantly, I would be a faint impression of who I am today without my amazing wife, Glenna; none of this would be possible without her continuous and unfailing love and support.

During my time at University of Washington, I have been deeply indebted to the academic and social community here. In particular, I don't think I would have made it through without Anna Sheppard's willingness to joke about the fountain in our first year, and her continual willingness to grab a beer since. Adam Tahir, Max Spetzler, Peter Uth, and Armand Awad were the best first-labmates that I could have asked for, and AERB 117 will always hold a special, windowless, place in my heart. Mathias Hudoba de Badyn was an excellent collaborator and brew partner during

our overlapping years. Jingjing Bu was an excellent resource for all of my basic math questions, and also a much needed friend to chat over coffee with (i.e. bash ML with). Sierra Adibi is the OG of speaking truth to power in A&A, and I wouldn't have been nearly as involved in our department if not for the privelege of getting to know her and following her example. Everyone involved in starting and keeping the GSAC going were incredibly important to adding a sense of (non-research-related) purpose to my time at UW. Finally, a huge thank you to anyone who attended a Controls Coffee Hour; I know I repeatedly dominated the conversations with esoterica, but I loved creating that space with you all.

On the academic side, Professor Anshu Narang-Siddarth was an amazing first mentor, and I wouldn't have developed a keen interest in the temporal aspects of dynamic systems if not for her early guidance, not to mention her support and confidence during my many fellowship applications. After she left UW, Professors Mehran Mesbahi and Kristi Morgensen were inexhaustibly patient while I figured out what to do, and I am extremely grateful for how supported they made me feel during a difficult time. Of course, I am further indebted to Mehran for taking me on and subsequently putting up with years of time scaled consensus. He has also never failed to kindly and expertly pull me out of whatever rabbit hole I'd squeezed into and point me back down a productive path.

Chapter 1

WHY TIME SCALES IN NETWORKED SYSTEMS?

We live in an increasingly connected world. The "internet of things" is comprised of modern electronics from cell phones to refrigerators sharing information with other devices in their proximity, and with the cost of computational and communication hardware ever decreasing, it is becoming easier and easier to create massive networks of objects sharing information. Furthermore, with the rise in access to rapid 3-D prototyping the barrier to actualization for robots and devices is incredibly low. The result of advances in these complimentary fields is our day-to-day world is rapidly filled with objects with potentially complex intrinsic dynamics interacting over a communication network of possibly unknown topology, giving rise to the potential for even more complex emergent behaviors as these factors combine. Furthermore, this trend of doing more with a larger number of cheaper agents also results in systems that will potentially exhibit large errors due to lack of quality in sensors or agent computational power. If we hope to understand the dynamics of such interconnected systems, for example, in order to exert some level of control or automation over these systems, we will need to be ready to handle to high level of complexity on both the agent-level, as well as on the communication network-level. Furthermore, understanding how input signals impact the individual agents as well as the network as a whole allows for effective leader selection and minimal energy control signals. From the other direction, if we are constructing these ecosystems of connected devices, understanding their dynamics and their performance under input signals or noise will allow the design of networks of agents that are robust to malicious influences while still being beneficial to the desired users.

Traditionally, one method for tackling the problem of complex dynamics was to employ a reduced order modeling method to generate asymptotically valid models for the system

on a reduced number of state variables. This allows for analysis and controller design to be done on a lower dimensional, and thus computationally less burdensome, set of dynamics. One of the most successful family of reduced order methods have been those stemming from the application of singular perturbation theory to complex systems and their analysis. For systems exhibiting dynamics on multiple time scales, these methods allow for the separation of the effects on the various time scales, and reduced order models that allow for the analysis of system-theoretic qualities such as stability and controllability, as well as for controller synthesis. Singular perturbation theory was first introduced as a method to facilitate system analysis and controller design in the late 1960s, and since then has been developed into a mature and well understood methodology for handling high dimensionality and multiple time scale behavior in systems.

On the other hand, we cannot ignore the interconnections between agents, and should seek to preserve any advantage that the networked nature of the problem grants us. Fortunately, the study of networked dynamical systems, as well as the behavior that it gives rise to, is also a well trod path in control theory and other fields. One of the most common and widely studied dynamics over networks is the consensus algorithm, which has been applied to networks of dynamical agents in many applications. What has been shown, and utilized to great success, is that many system-theoretic qualities that we are interested in from a controls perspective, again, such as controllability, stability, and certain performance metrics, can be reasoned about using graph-theoretic concepts derived from the topology of the communication graph under which the agents share information. However, in many applications of the consensus algorithm, an assumption of identical agent dynamics is assumed. There are, of course, exceptions to this rule, but in general heterogeneous agent dynamics are assumed to be able to track the reference signal that would be generated by a network of single-integrator agents.

The intersection of these two ideas, that is, networked systems that exhibit behavior on multiple time scales, is thus an interesting avenue of research that could have applications in the analysis and control of the increasingly complex and varied systems that are proliferating in our world today. To that end, this dissertation presents work into the analysis and

performance of consensus networks featuring distinct agent time scales. After discussing some necessary mathematical background and relevant results from the literature in Chapter 2, we will dive into Chapter 3. First, we will consider how non-ideal clocks can give rise to both biased state measurements as well as be an impediment to accurate realization of control inputs. This motivates a discussion and distinction of how agents in consensus networks are exchanging information, and allows us to define agent types and information types. Finally, we methodically asses how these various observation and agent types result in several different variations on the basic consensus algorithm. Multiple categories of problems under this umbrella can be categorized by what we term the "scaled consensus problem," which motivates a deep dive into those formulations.

In Chapter 4 we dive into the analysis of scaled consensus problems. First in Section 4.1, we detail some early results on using concepts from singular perturbation theory for reduced order modeling for systems being controlled via "slow" actuators. Conceptually, this motivates study of scenarios where any agent within a network can be the input of a control signal, even if it is operating at a slower time scale than other agents or some network-level behavior. In order to investigate this further, in Sections 4.2, 4.3, and 4.4 we will present work that considers the performance of consensus on networks of time scaled agents under the influence of both noise inputs, as well as control signals into single nodes within the network. What we find is that even without resorting to reduced order models, the multi-time scale consensus problems can be evaluated in a similar fashion as their mono-scale analogs that are well-studied from a graph-theoretic angle.

In Chapter 5, we return to considering general clock-biased consensus problems, and attempt to characterize the errors arising from implementation of consensus for agents with non-uniform clocks. We also propose a clock synchronization algorithm that allows agents to develop a shared clock model without knowledge of their local clock parameters. This can lead to reduction of error when combined with periodic state updates for double-integrator systems. Finally, in Chapter 6 we detail some minor results and unexpected places that the scaled consensus results can be applied. We consider some alternative graph-theoretic

interpretations of agent-based time scales, as well as updating some classic graph centrality measures under the inclusion of time scales. Also, we highlight a potential application for saturated input systems, for which the scaled consensus problem makes a fortuitous appearance. Some parting remarks summarizing the body of work and suggesting some future directions are included in Chapter 7.

Chapter 2

TECHNICAL MOTIVATION AND MATHEMATICAL NOTATION

In this chapter, we will first agree on some common notation that will be adhered to throughout the rest of the report. From that starting point, we will discuss relevant literature background and basic problem set-ups for three concepts that will be central to the prior and proposed work in later chapters. In §2.3 we delve into reduced order modeling of systems in standard form, and present the main result (Tikhonov's Theorem) that quantifies the validity of those models. Then, we will move on to some basic concepts from graph theory in §2.5, and define the multi-time scale consensus protocol that will be the focus of analysis in later chapters. Finally, we define the $\mathcal{H}_2$ system norm in §2.6; this norm will be our preferred performance metric for discussions of network performance.

2.1 Basic Notation

To start, let us define some basic mathematical notation that will be used throughout this report.

The set of real numbers is denoted by $\mathbb{R}$, the non-negative reals as $\mathbb{R}_+$, the positive reals as $\mathbb{R}_{++}$, and the real n-dimensional Euclidean vector space as $\mathbb{R}^n$. The set of positive-(semi) definite matrices will be denoted by $\mathcal{S}^n_{++}$ ($\mathcal{S}^n_+$). We will deal extensively with vector quantities, which will be written in lower-case with dimension given at instantiation, e.g. $x \in \mathbb{R}^n$ or $z \in \mathbb{R}^m$. Matrices are written in capital-case with dimension similarly given, e.g. $M \in \mathbb{R}^{n\times m}$, $\Omega \in \mathbb{R}^{k\times k}$. I will be used to denote the identity matrix, with dimension given as a subscript (I_n) when the dimension is not immediately clear from context. When scalar values (beyond ubiquitous indexes such as i, j, k) are invoked they will predominately

be individual entries in vectors or matrices, and denoted using subscripts, e.g. x_i for the ith entry in x, or $[M]_{ij}$ for the scalar value residing in the ith row and jth column of matrix M. Sets (beyond $\mathbb{R}$) and graphs will be denoted with capital-case script, $\mathcal{V}, \mathcal{E}$. Functions will be denoted with lower-case with arguments given with parenthesis, e.g. $f(x, y), g(t)$. Time dependent quantities such as functions, vectors, and matrices will be signified with (t), though this time dependence will be omitted for brevity when time dependence is clear from context.

The Kronecker product of two matrices A, B is denoted by $A \otimes B$, and the operation Blkdiag$(A_1, \ldots, A_n)$ yields the matrix which has the matrices $A_1, \ldots, A_n$ on its diagonal. For matrices A and B, $A \preceq B$ implies $B - A \in \mathcal{S}_+$. The ith eigenvalue of a matrix will be denoted by λ_i.

2.2 Clock Models

At the core of many applications of coordinated multi-agent systems, especially those which consider consensus over some positional states, is an effective rendezvous between agents. One practical complication of achieving the necessary rendezvous for real-world systems is dealing with differences in agent clocks in order to agree on the time frame over which actions or controls are being applied. The ubiquity of the Global Positioning System (GPS [117]) in terrestrial applications allows for individual agents to access a common and accurate clock, however, in size or power-limited systems agents may not have GPS receivers, resulting in agents having to rely on internal clocks. In the context of wireless sensor networks, this problem has been investigated and a number a algorithms for clock synchronization have been proposed [107, 59, 12, 45]. For networked satellite applications, similar considerations are relevant, as a GPS signal may be unavailable (in the eventual case of extra-terrestrial applications), or it may be cost, power, or mass prohibitive to include a GPS receiver on a satellite (in the case of small or low-cost, mass produced satellites).

A primary motivation and assumption for the problems we will be considering are that agents have access to an internal clock, that may be distinct from the clocks of its neighbors.

These internal clocks are taken to be continuous approximations of physical hardware clocks, which use the output from an oscillator to increment a time counter to define a computer clock,

$$\tau(t) = k \int_0^t w(s)\mathrm{d}s + \tau(t_0)$$

where $w(t)$ is the frequency of the oscillator, the coefficient k translates the time counter units to the clock time units, and $\tau(t_0)$ is the initial clock setting. If the angular frequency of the oscillator is stable and can be approximated by a constant, then the clock model can be expressed as an affine equation,

$$\tau(t) = \alpha t + \beta \tag{2.1}$$

where α is the clock rate (skew), β is the clock offset, and t is the "true" time [72]. For physical systems operating in a range of temperatures, pressures, or voltages, the clock rate may actually be time varying. However, under the assumption that it is varying much more slowly than the plant dynamics or update rate, it can be assumed to be constant (or, a higher order clock model can be adopted to allow for a changing parameters).

Now, consider the dynamics of an individual agent, for which we can define a distinct clock, $\tau_i(t) = \alpha_i t + \beta_i$. Self measurements of any state information will be made with respect to this hardware clock. However, given the clock model in (2.1), we can see that dynamics on an intrinsic clock can be expressed in terms of the true clock,

$$\begin{aligned} \frac{d}{dt} &= \frac{d\tau}{dt}\frac{d}{d\tau}, \\ \frac{d}{dt} &= \alpha\frac{d}{d\tau}. \end{aligned} \tag{2.2}$$

We will further explore how this relationship between the agent clock time scale/skew rate and a notion of true clock time impacts consensus systems in later chapters.

2.3 Model Reduction via Singular Perturbation Theory (SPT)

Valid approximations for the solution to singularly perturbed problems were first proposed in 1905 by Ludwig Prandtl motivated by boundary layer problems in fluid flow [64]. Due to the brevity of the initial publication and its venue this contribution was not widely noticed, however, despite a surge in the study of asymptotic methods motivated by problems such as quantum mechanics, fluid dynamics, and circuit dynamics. By the mid-20$^{\text{th}}$ century, however, singular perturbation problems were discovered anew, and saw application based research by American researchers such as Friedrichs and Wasow [41], and Levinson [67], as well as work from a pure mathematical direction by Russians Tikhonov [118] and his student Vasil-eva [122]. By the 1960s, the mathematical foundations were well-established and motivated, and Kokotović and Sannuti [105, 106] applied the existing theory to optimal control problems. Since that time, SPT has been developed into a mature methodology for system analysis and controller design [62], particularly in aerospace applications [78, 102].

Perturbation problems, most broadly, are algebraic or differential equations that contain terms multiplied by small, positive parameters, often denoted by $\epsilon \in \mathbb{R}_{++}$ [56]. These parasitic parameters often introduce numerical stiffness into computational approximation of roots or solutions to such equations. Furthermore, in many cases, the dimension of perturbed equations can be reduced by making simplified assumptions about the perturbed terms. These reduced order models, if it can be shown that they are valid approximations to the full dynamics or equation solutions, are very useful for analysis of the full system. In particular, reduced order systems aid in the analysis of dynamic system solutions, which is an integral step in determining stability and controllability in dynamic systems which paves the way into controller design.

When the solution to a perturbed equation (with state variable x) can be uniformly approximated by an asymptotic expansion of the form,

$$\phi(x) \simeq \sum_{n} \delta_n(\epsilon)\psi_n(x)$$

as $\epsilon \to 0$, where $\delta_n(\epsilon)$ are functions of increasing order in ϵ, we say that the problem has a *regular perturbation*. We can see from the expansion above that regular perturbations allow for the separation of the perturbation effects (captured in the $\delta(\epsilon)$ functions) from the state behavior (captured by the $\psi(x)$ functions). However, not all perturbed equations permit such expansions. When the approximation's dependence on ϵ cannot be separated from the state parameters, it is termed a *singular perturbation* [56]. Typically, this is the result of the phenomenon that, unlike in regular perturbation problems, the naïve approximation of $\epsilon = 0$ reduces the order of the system, creating a degenerate problem that cannot be reconciled with the initial conditions of the full problem. As we'll see later in this section, an interpretation of this phenomenon is that a "boundary layer" exists where the solution changes rapidly, and the initial/boundary conditions "lost" in the degenerate problem are "buried" in this boundary layer. To recover the initial conditions - and hence an approximation to the solution to the full system - the boundary layer can be "stretched" through a transformation of the dependent variable.

Determination of whether or not a perturbed system is singularly perturbed is not trivial, though, the existence of eigenvalues or functions that are well separated (which gives rise to behavior on at least two distinct time scales or frequencies) are a clear indication that a system potentially is. If a system is singularly perturbed, there exists a coordinate transformation that puts the system into what is known as standard form (but again, the triviality of finding the coordinate transformation is not guaranteed,

$$\begin{aligned} \dot{x} &= f(x, z, t; \epsilon), \;\; x(t_0) = x_0(\epsilon) \\ \epsilon \dot{z} &= g(x, z, t; \epsilon), \;\; z(t_0) = z_0(\epsilon), \end{aligned} \tag{2.3}$$

Making the naïve assumption of $\epsilon = 0$ gives the *degenerate approximation* or *reduced problem*,

$$\begin{aligned} \dot{x} &= f(x, z, t; 0), \;\; x(0) = x_0(0) \\ 0 &= g(x, z, t; 0), \;\; z(0) = z_0(0), \end{aligned}$$

From the algebraic equation $0 = g(x, z, t; 0)$, a quasi-steady state behavior for $z(t)$ can be found,

$$\bar{z} = h(x, t)$$

when substituted into the degenerate dynamics of $x(t)$, this yields a reduced order (and independent of z and ϵ) model for x,

$$\dot{x} = f(x, h(x, t), t; 0), \;\; x(0) = x_0(0). \tag{2.4}$$

This degenerate model might intuitively capture the dynamics of $x(t)$, but obviously any dynamics in $z(t)$ has been lost. To recover these lost dynamics, we can introduce a shift of the quasi-steady-state equilibrium to the origin,

$$y = z - h(x, t) \tag{2.5}$$

and introduce the time scaling of,

$$\begin{aligned} \tau &= \frac{t}{\epsilon} \\ \frac{d}{dt} &= \frac{1}{\epsilon}\frac{d}{d\tau}. \end{aligned} \tag{2.6}$$

Then in the regime of $\epsilon = 0$ in the y-dynamics gives the *boundary layer model*,

$$\frac{dy}{d\tau} = g(x, y + h(x, t), t; 0) \tag{2.7}$$

where (x, t) can be treated as fixed parameters for the purposes of solving the differential equation.

We now have reduced order models in two regimes, a *degenerate* problem that assumes an equilibrium in z, and a *boundary layer* problem that stretches time so that x, t are static and recovers the fast-z dynamics. A relevant question, then, is when the solutions to (2.4)

and (2.7) are valid approximations to the full solution of (2.3). The answer to this question is the main result from SPT what will be leveraged in this report. It is commonly known as Tikhonov's Theorem, and we present it for completeness in the following theorem.

Theorem 1 (Theorem 11.1 from [58])**.** *Tikhonov's Theorem states that for assumptions as to the continuity of on the vector fields (and their derivatives) for the fast and slow variables given in (2.3), there exists a critical value of ϵ, below which the degenerate and boundary layer models are valid (error on order ϵ). We state this formally below.*

Consider the singular perturbation problem in standard form as given by (2.3) and let $z = h(x,t)$ be an isolated root of $0 = g(x,z,t;0)$. Assume that the following conditions are satisfied for all,

$$[t, x, z - h(x,t), \epsilon] \in [0, t_1] \times D_x \times D_y \times [0, \epsilon_0]$$

for some domains $D_x \in \mathbb{R}^n$ and $D_y \in \mathbb{R}^m$, in which D_x is convex and D_y contains the origin:

- *The functions f, g and their first partial derivatives with respect to (x, z, ϵ) and the first partial derivative of g with respect to t are continuous; the function $h(x,t)$ and the Jacobian $[\partial g(x,z,t;0)/\partial z]$ have continuous first partial derivatives with respect to their arguments; the initial data $x_0(\epsilon)$ and $z_0(\epsilon)$ are smooth functions of ϵ.*

- *The reduced problem (2.4) has a unique solution $\bar{x}(t) \in S$ for $t \in [t_0, t_1]$ where S is a compact subset of D_x.*

- *The origin is an exponentially stable equilibrium point of the boundary layer problem (2.7), uniformly in (x,t); let $\mathcal{R} \subset D_y$ be the region of attraction and Ω_y be a compact subset of R_y.*

Then, there exists a positive constant ϵ^* such that for all $z_0 - h(x_0, t_0) \in \Omega_y$ and $0 < \epsilon < \epsilon^*$ the singular perturbation problem of (2.3) has a unique solution $x(t,\epsilon)$, $z(t,\epsilon)$ on $[t_0, t_1]$, and

$$x(t,\epsilon) - \bar{x}(t) = \mathcal{O}(\epsilon)$$

$$z(t,\epsilon) - h(\bar{x}(t),t) - \hat{y}(t/\epsilon) = \mathcal{O}(\epsilon)$$

hold uniformly for $t \in [t_0, t_1]$*, where* $\hat{y}(\tau)$ *is the solution of the boundary layer problem* (2.7)*. Moreover, given any* $t_b > t_0$*, there is* $\epsilon^{**} \leq \epsilon^*$ *such that,*

$$z(t,\epsilon) - h(t,\bar{x}(t)) = \mathcal{O}(\epsilon)$$

holds uniformly for $t \in [t_b, t_1]$ *whenever* $\epsilon < \epsilon^{**}$*.*

Proof. See [58, Appendix C.17]. □

This result allows for the reduced order models in both the boundary layer and where the dynamics evolve on the solve manifold are asymptotically valid, and thus can be utilized for modeling, system analysis, and controller design [63]. In Chapter 4.1, we will develop a similar result for a system in non-standard form for when a system is controlled through an actuator slower than the plant dynamics. However, Tikhonov's Theorem will form the rigorous backbone of reduced order modeling that forms a graph-theoretic interpretation of multi-scaled dynamics.

2.4 Relating Clocks and Time Scale Parameters

In the previous two sections, we have introduced agent-based clocks, which evolve with respect to some non-unity skew rate, and also introduced the concepts of time scales, which transform a set time measure. As both concern the rate at which we are assigning or perceiving time, they are of course intimately related. From (2.2) and (2.6), we can explicitly see,

$$\alpha_i = \frac{1}{\epsilon_i}.$$

This relation means that we can always interpret a problem over a mix of time scales as an equivalent problem of states or agents relying on a clock model with the inverse clock skew parameter (with zero clock offset). Somewhat confusingly, however (due to the inverse

nature of the relation), when we are reasoning about a "slow" agent, this corresponds to an agent with a fast clock rate, or conversely, a "fast" agent can be perceived to be on a slow clock. Due to this correspondence between clocks and time scales, for the remaining discussions, we will alternatively refer to both *time scale parameters* as well as *clock skews* or *clock rate parameters*, depending on the source of the work and its primary motivation.

2.5 Graph Theory and Consensus Problems

Graph theory got its start in the 18th century with the famous "Seven Bridges of Königsberg" paper by Leonhard Euler. At the core of graph theory is the simple idea that a graph captures relationships between multiple objects. The objects are represented by nodes (or vertices), and if two objects are related to each other they are connected by an edge (or link). Throughout the incredible range of applications of graph theory (in nearly every field imaginable), the interpretation of what the nodes (individual agents, areas, molecules) and edges (physical proximity, information flow, social influence) represent are incredibly varied, as well as what the topology of the graph represents. In systems and control theory, a natural interpretation is that nodes represent physical agents and the edge distribution signifies that communication network over which agents share information about their state or observations of some external process.

In the context of this report, which slightly extends the convention adopted in most control applications, we define a *graph*, denoted $\mathcal{G}$, over n nodes is a quartet of sets $\mathcal{G}\left(\mathcal{V}, \mathcal{E}, W, E\right)$ where $\mathcal{V} = [n]$ is a set of numbered nodes (or agents), $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of m edges, $W \in \mathbb{R}^{m \times m}$ is a diagonal matrix of edges weights, and $E \in \mathbb{R}^{n \times n}$ is a diagonal matrix of nodal time scale parameters. Each agent will be indexed by subscripts, e.g. $\nu_i \in \mathcal{V}$ represents the ith agent where $1 \leq i \leq |\mathcal{V}|$, or simply by the index (i). If $ij \in \mathcal{E}$ ν_i and ν_j are connected by an edge $(i \sim j)$, and they are referred to as adjacent agents or neighbors. For a given agent ν_i, $\mathcal{N}(\nu_i) = \{\nu_j \mid \nu_i \sim \nu_j \forall j \in \mathcal{V}\}$ denotes the neighborhood (all neighbors of i), and $\deg(i) = \mathcal{N}(\nu_i)$ denotes the unweighted degree of i. We will not consider signed graphs in this report, so $W \in \mathcal{S}_{++}^m$. The edge set can be ordered by a mapping, $\kappa(\cdot)$, such that $l = \kappa(ij)$

iff $ij \in \mathcal{E}$. By this mapping we can denote the weight on edge ij by w_l (corresponding to $[W]_{ll}$) or w_{ij}, interchangeably. The time scale matrix $E = \text{diag}(\epsilon_i, \ldots, \epsilon_n)$, and individual time scale parameters are taken to be strictly positive. Except where later noted, there is no assumptions as to the relationship or ratio between any two time scale parameters.

There are several matrices that capture the communication topology between agents that will be useful throughout the subsequent chapters. The incidence matrix, $D(\mathcal{G})$ is a $|\mathcal{V}| \times |\mathcal{E}|$ matrix, where the l-th column denotes an edge between two nodes in the form of an edge vector, $a_{\kappa(ij)} = e_i - e_j$ (equivalently, $e_j - e_i$). The *adjacency matrix*, $\mathcal{A}$, of a graph also encodes the connections between agents, with entries

$$[\mathcal{A}]_{ij} = \begin{cases} w_{ij}, & i \sim j, i \neq j \\ 0, & i \nsim j \text{ or}, i = j \end{cases}$$

The *degree matrix*, Δ, of a graph is defined as the diagonal matrix with $[\Delta]_{ii} = \sum_{j \sim N(i)} w_{ij}$. With these descriptive matrices defined, we can now define the Laplacian matrix associated with the graph, $\mathcal{L}$, which will be of particular interest,

$$\mathcal{L} = \Delta - \mathcal{A}$$

or equivalently,

$$\mathcal{L} = D(\mathcal{G}) W D(\mathcal{G})^T.$$

A simple choice for the individual dynamics is to assume each node is a single integrator,

$$\dot{x}_i = u_i,$$

where u_i is a signal designed to achieve some control objective. A weighted, decentralized feedback controller that seeks to bring the agents into consensus is given by,

$$u_i(t) = \sum_{j \in \mathcal{N}(i)} w_{ij} \left(x_j(t) - x_i(t) \right),$$

When the states of the nodes are stacked into a single vector, combining the previous dynamics and control law gives the standard consensus problem,

$$\begin{aligned} \dot{x} &= -D_{\mathcal{G}} W D_{\mathcal{G}}^T x \\ \Rightarrow \dot{x} &= \mathcal{L} x. \end{aligned} \tag{2.8}$$

The consensus algorithm is of particular interest because it is a linear system (appealing from a systems-theoretic perspective) with the system matrix given by a graph Laplacian, which encodes the communication topology between agents. Control schemes based on average consensus algorithms have been successfully applied in areas of formation control, distributed estimation, and multi-vehicle control among others [85, 83, 116, 53, 44]. In many of these applications an important consideration is the network's response to external influence through a subset of agents. This can be important for robustness analysis, identifying potential leader nodes [91], or as a way to analyze a networked system's susceptibility to intruders and nefarious inputs [19, 87]. This is typically referred to as *influenced consensus*, and has been considered for the case of a single input node [15] as well as for multiple grounded nodes [96]. Finally, note that consensus on higher-dimensional agent states is possible through a Kronecker product formulation of (2.8), or in the case of coupled substates, a matrix-weighted consensus protocol.

From (2.8), we can consider a related problem that will also be featured heavily in later sections. We introduced the incidence matrix as encoding the edges within the graph, but we can also note that,

$$s(t) = D_{\mathcal{G}}^T x(t),$$

results in $s(t) \in \mathbb{R}^m$ being a vector of state differences, where m is the number of edges in $\mathcal{G}$. The state differences are a measure of the disagreement over each edge, and can be interpreted as an edge state. Furthermore, if we consider the dynamics of the edge states for a (unity weighted) consensus network, we find,

$$\begin{aligned}\dot{s}(t) &= D_{\mathcal{G}}^T \dot{x}(t) \\ &= -D_{\mathcal{G}}^T D_{\mathcal{G}} D_{\mathcal{G}}^T x \\ &= -D_{\mathcal{G}}^T D_{\mathcal{G}} s(t) \\ &:= -\mathcal{L}_e(\mathcal{G}) s(t). \end{aligned} \tag{2.9}$$

The system matrix here, $\mathcal{L}_e(\mathcal{G}) = D_{\mathcal{G}}^T D_{\mathcal{G}}$ is defined as the *edge Laplacian*. It can be shown that the origin is the equilibrium point corresponding to the consensus subspace of the consensus dynamics on the nodes, as well as the edge Laplacian sharing the non-zero spectra with the graph Laplacian [75]. We will use these properties, along with a more complete way of transforming consensus on the nodes to consensus over the edges in Chapter 4.

We will also consider the case where the agent states are taken to represent a scalar value (often associated with a position), along with that scalar's velocity, that is each agent state is take to be $[x_i, v_i]$ where $x_i, v_i \in \mathbb{R}$. When the acceleration of the agent (that is, the derivative of the velocity substate) is available as the input for a control signal, it is termed *double integrator dynamics*. This form is especially relevant for vehicle or robotic applications where the scalar state is taken as the agent position and agents accelerate to promote rendezvous or flocking behaviors. To specifically codify this case, we have,

$$\begin{aligned}\dot{x}_i &= v_i, \\ \dot{v}_i &= u_i.\end{aligned}$$

An extension of the consensus protocol for single integrators that encourages alignment of positions and velocities is given by,

$$u_i = \sum_{j \sim i} (x_j - x_i) + \sum_{j \sim i} (\dot{x}_j - \dot{x}_i).$$

Similar to the single-integrator case, when the stacked vectors of agent positions and velocities are subsequently stacked, can be expressed in vector form as,

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & I \\ -\mathcal{L} & -\mathcal{L} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}. \tag{2.10}$$

As in the single integrator case, consensus on higher dimension agent states (to represent positions and velocities in 2 or 3-D, say) can be achieved via Kronecker products. As those formulations do not fundamentally change the analyses undertaken in later chapters (and add significant clutter to the requisite derivations), we will restrict ourselves to considerations in 1-D whenever higher dimensions are not strictly required.

Properties of Consensus Dynamics

One reason for the popularity of consensus dynamics has enjoyed in the control theory literature is its ability to be interpreted through a graph theoretic lens. Thus, properties of the underlying communication topology of agents can inform system-level behavior. Here, we will discuss a few of these properties, and underscore an assumption that will be made in the future chapters.

First, consider the spectra of the Laplacian matrix. Being a symmetric, positive semi-definite matrix, the eigenvalues are all real and non-negative. An immediate consequence of this is the consensus dynamics, driven by $-\mathcal{L}$, are marginally stable. The eigenvalues can be ordered $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. The eigenvalues equal to zero correspond to the equilibrium states for the consensus dynamics for each connected components of the graph. Thus, if the graph is *connected* (in the sense there is a single component, and there is a path from each agent to every other agent in the network), then it has a single zero eigenvalue. Results pertaining to connected graphs are applicable to each connected component of unconnected networks, so for our analyses we only consider networks with a single connected component. We codify this in the following assumption.

Assumption 1. *The underlying graph for all dynamics considered, $\mathcal{G}$, is connected.*

The smallest non-zero eigenvalue (λ_2 for any connected graph) is known as the algebraic connectivity and is a measure of how connected the network is. For unity edge weightings,

it is lower bounded by the connectivity of a path graph, and upper bounded by that of a complete graph. In fact, the largest eigenvalue of the Laplacian is also bounded by the maximum eigenvalue of the complete graph [75]. The spectra of the graph Laplacian under the presence of time scaling will be an important consideration in Chapter 4.

Now, returning to the zero eigenvalue of the Laplacian. This represents the null space of the Laplacian matrix, of course, and we can note that from construction (the row and column sums of the Laplacian matrix are zero), the left and right eigenvector corresponding to the zero eigenvalue are given by $\mathbf{1}$ (and its transpose for the left). We can use this fact, along with the general solution to the consensus dynamics to find that in the case of single integrator agents, the consensus subspace is **span1**, and the consensus value of a network of agents with initial condition $x(0)$ is given by,

$$x_c = \frac{\mathbf{1}^T x(0)}{\mathbf{1}^T \mathbf{1}}, \tag{2.11}$$

which corresponds to the average of the initial conditions. It is for this reason that the consensus protocol is sometimes referred to in the literature as the "average agreement" protocol. For the double integrator case, the velocity states come to this average quantity,

$$v_c = \frac{\mathbf{1}^T v(0)}{\mathbf{1}^T \mathbf{1}},$$

while the positional states will approach consensus to a point that is evolving at this consensus velocity,

$$x_{c,di}(t) = \frac{\mathbf{1}^T x(0)}{\mathbf{1}^T \mathbf{1}} + \frac{\mathbf{1}^T v(0)}{\mathbf{1}^T \mathbf{1}} t.$$

2.5.1 Matrix Weighted Graphs

We have mentioned a few times previously that we can allow agents to have higher dimensions, that is, they can be vector-valued instead of scalar valued. This can allow, then, for edges to be interpreted as matrix-valued, which allows for a coupling between agent states to

be considered. Thus, we consider that associated with the graph are $\mathcal{W}$, a set of matrix edge weights, and $\mathcal{T}$, a set of time scaling factors for agents' states. This can be accommodated by the following adjustments to the preceding definitions.

Individual agents states are vector-valued, $x \in \mathbb{R}^k$, and each agent will be indexed by subscripts, e.g. $\nu_i \in \mathcal{V}$ to represent the i-th agent where $1 \leq i \leq |\mathcal{V}|$. If $(i,j) \in \mathcal{E}$, the i-th and j-th agents are connected by an edge $(i \sim j)$, and they are referred to as adjacent agents. For a given agent, i, $N(i) = \{j \mid i \sim j \ \forall j \in \mathcal{V}\}$ denotes the neighbors of i, and $\deg(\nu_i) = |N(i)|$ denotes the unweighted degree of i. The k values comprising an agent's state will be referred to as *substates*, and the j-th substate of the i-th agent is denoted as $x_{i,j}$.

Matrix-valued weights will be denoted $W_e \in \mathcal{S}^k_{++}$, and so $\mathcal{W} = \{W_e \mid e \in \mathcal{E}\}$. The *weight matrix* $\mathbf{W}$ is a $k|\mathcal{E}| \times k|\mathcal{E}|$ blockwise diagonal matrix containing the weights W_{ij} of each edge e. As in graphs with scalar agent states, the edge set can be ordered by a mapping, $\kappa(\cdot)$, such that $l = \kappa(ij)$ if and only if $(i,j) \in \mathcal{E}$. By this mapping, we can denote the weight on edge $\kappa(ij)$ by W_l or W_{ij}, interchangeably.

For the formulation of matrix-valued weights, we define $\mathbf{D}(\mathcal{G}) \triangleq D(\mathcal{G}) \otimes I_k$ and $\mathbf{a}_{ij} \triangleq a_{ij} \otimes I_k$. The *weighted graph Laplacian* $\mathcal{L}_w$ of an undirected matrix-valued graph $\mathcal{G}$ can be defined thusly as $\mathcal{L}_w \triangleq \mathbf{D}(\mathcal{G})\mathbf{W}\mathbf{D}(\mathcal{G})^T = \sum_{ij\in\mathcal{E}} \mathbf{a}_{ij} W_{ij} \mathbf{a}_{ij}^T$. Equivalently, it can be defined blockwise with the $k \times k$ block whose rows are associated with the ith node and whose columns are associated with the jth node given by $\sum_{j\in\mathcal{N}_i} W_{ij}$ if $i = j$, $-W_{ij}$ if $(i,j) \in \mathcal{E}$, and $\mathbf{0}_{k\times k}$ if $(i,j) \notin \mathcal{E}$.

2.6 $\mathcal{H}_2$ Performance of Consensus Networks

As mentioned in the previous section, a natural question for consensus networks is how the underlying network topology affects the behavior of the dynamics operating over the network. This question has attracted significant interest in systems and control communities, particularly as certain notions of performance and control can be directly related to graph theoretic properties of the network. Of particular interest for this work are system-theoretic

measures such as $\mathcal{H}_2$ and $\mathcal{H}_\infty$ system norms. For networked dynamical systems, the $\mathcal{H}_2$-norm can be interpreted as a measure of how input energy is attenuated over the network, or how noise drives deviations from the natural consensus state [109].

In light of these interpretations, there have been several works investigating the characterization of $\mathcal{H}_2$ performance for consensus networks. In [16, 21, 50], the performance of leader-follower networks is considered, and algorithms for rewiring and reweighting the network for optimal noise rejection are discussed. Similarly, [5, 91, 92] have utilized the $\mathcal{H}_2$-norm as a measure of coherence in networks and considered problems such as local feedback laws and leader selection to promote coherence. Most relevant to the present contribution, the works [127, 128] investigated the impact of cycles on the $\mathcal{H}_2$ performance of noise-driven consensus networks. The examination of networks under noise inputs is important for real-world implementation of consensus onto physical systems, and for considering network resilience in the presence of adversarial noise injections.

Broadly speaking, these methods are intimately related to robust control methodologies in the context of control theory. Two classic methods are those of $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control synthesis [110], which can be summarized as seeking to minimize all singular values or the maximum singular value of the closed loop transfer function, respectively. In traditional controller design, however, the plant model is known/determined by the application in question, and the form of the controller is open to design (or for a given controller form there are some parameters to optimize). In much of the aforementioned performance literature, the form of the controller is known (either consensus, or similar feedback law), but the plant model is determined by the communication structure between agents in the network. Thus, instead of considering controller design, the robust control problem is solved by considering the optimal structure or advantageous edge weighting to minimize the system norm in question. In either case, however, the goal of robustness to extraneous noise is unchanged. In consensus systems, and in particular the $\mathcal{H}_2$ norm that we will focus on in later sections, the robustness can be be interpreted as the mean deviation from the non-noise perturbed consensus value (which is the coherence in question in previous citations), which is in-line

with deviation from a planned trajectory or nominal state typically considered in controller design problems.

We now define the $\mathcal{H}_2$ norm, which we will utilize as a performance metric. The input-output excitation properties of a linear system $\dot{x} = Ax + Bu,\ y = Cx$ with transfer function $G(s) = C(sI - A)^{-1}B$ can be described using the $\mathcal{H}_2$ norm,

$$\mathcal{H}_2^2(G) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathbf{tr}\left[G(j\omega)^\dagger G(j\omega)\right] d\omega. \tag{2.12}$$

The $\mathcal{H}_2$ norm measures the steady-state covariance of the output of the system under zero-mean, unit-covariance white noise inputs, or equivalently the root-mean-square of the impulse response of the system. Beyond the relevant graph-centric interpretations of the $\mathcal{H}_2$ norm mentioned previously, there is another reason that it can be readily used as a performance metric in linear systems. We can note that the integral over the transfer function product in frequency in (2.12) is equivalent to an integral of the impulse response matrix product in the time domain,

$$\mathcal{H}_2^2(G) = \int_0^{\infty} \mathbf{tr}\left[g(\tau)^T g(\tau)\right] d\tau. \tag{2.13}$$

where $g(\tau) = Ce^{A\tau}B$ is the impulse response matrix at $t = \tau$. Thus, it can be seen that the $\mathcal{H}_2$ norm can also be written as,

$$\mathcal{H}_2^2(G) = \mathbf{tr}\left[B^T X_o B\right] = \mathbf{tr}\left[C X_c C^T\right]$$

where X_o and X_c denote the observability and controllability gramians of the linear system. This is advantageous because in addition to the integral expressions for stable dynamics, the gramians can be found via algebraic matrix equations, such as the controllability gramian from the algebraic Lyapunov equation,

$$AX + XA^T = -BB^T.$$

For consensus dynamics, the A matrix is a graph Laplacian for which by-inspection solutions to the Lyapunov equation exist (when the consensus subspace has been accounted for); thus, the $\mathcal{H}_2$ norm permits closed-form equations to assess overall performance. In Chapter 4.2, we will formulate closed form solutions to the Lyapunov equation for noise-driven consensus problems to assess the performance of the time scaled consensus system, and in Chapter 4.4 we will present a bound on $\mathcal{H}_2$ performance for single-input scaled consensus.

2.7 Chapter Summary

In this chapter, we've introduced and outlined some core tenants from singular perturbation theory, graph theory and the consensus algorithm, and the $\mathcal{H}_2$ system norm as a metric of performance for consensus problems. In the following chapter these concepts will be applied to address problems in reduced order modeling of systems in nonstandard form, and then analysis of consensus performance due to noise or directed inputs.

Chapter 3

CONSIDERING CLOCKS IN CONSENSUS FORMULATIONS

An implicit assumption of both (2.8) and (2.10) is that all agents have access to a single, true clock, and the relevant dynamics evolve with respect to this clock. Here, we will now relax that assumption by allowing each agent to possess an internal clock, which it must rely on to measure its own, or in some cases, neighboring agent states. Thus, for an agent i, assume that it only as access to a clock with dynamics,

$$\tau_i(t) = \alpha_i t + \beta_i,$$

where α_i and β_i are the agent clock skew and offset from the true clock, and can be distinct from any other agent in the network. To begin, let us consider how the reliance on an internal clock biases measurements of relevant quantities.

3.1 Clock Biasing

We can see that in the consensus control laws for (2.8) and (2.10) depend on the relative positions (scalar states for both single and double integrator systems) and relative state velocities (double integrator). Thus, in determining how distinct clocks affect consensus dynamics, the first consideration must be how those close affect those measurements, as well as the implementation (realization) of a commanded velocity or acceleration.

3.1.1 Relative State Measurement

Agents can determine the relative state measurements by either determining (or being informed of) their and their neighbors state information then calculating it explicitly, or by measuring the difference directly. We cover the various cases below:

True global positions: The simplest case is the assumption that agents are able to measure their own or their neighbors global positions/states, independent of their clock. Note that this could be 'either/or' in that agents may possess the ability to measure their own state, but not their neighbors, etc. Examples of such systems would be multi-agent robotic systems that use landmark identification to determine global positions, or such sensors as a star tracker to determine attitude states in satellite systems.

Inertial Navigation: When no direct measurement of a global position can be made, but a history or control inputs is available along with an known initial condition, the control inputs can be integrated to yield a state estimate. In a physical system, this estimate is accurate if there are no external disturbances and all control inputs are able to be realized.

Relative measurements: If the agent states correspond to a position, then relative distance between agents can be found via a range-finding methods without knowledge of either global position. However, commonly, this relies on measuring light travel time (in the case of laser/infrared rangefinders), and thus dependent on a time interval measurement. With the clock model in (2.1), we see that and Δt measured in true time corresponds to $\alpha \Delta t$ measured with the biased clock, thus,

$$d_{\text{measured}} = \alpha d_{\text{true}}.$$

In later sections, state measurements that are biased/erroneous based on a non-perfect agent clock will be referred to as *clock measurement biased* or simply as having a *biased measurements*.

3.1.2 Biased Velocity and Acceleration

As velocities and acceleration are inherently measures of rates, any measurements of these quantities will be affected by any biased clock. From 2.2, we can see that for velocities,

$$v_{\text{measured}} = \frac{1}{\alpha} v_{\text{true}},$$

and for accelerations,

$$a_{\text{measured}} = \frac{1}{\alpha^2} a_{\text{true}}.$$

Importantly, these biases hold for both measurements of rates (clock biased measurements as in the positional case), as well as for realizations (when agents are asked to update their velocity/acceleration to a new value). That is, if commanded a velocity or acceleration to realize, the above relations give that the true velocity or acceleration that will be achieved are $\alpha_i v_{\text{commanded}}$ or $\alpha^2 a_{\text{commanded}}$, respectively. In later sections, we will refer to effects due to agents with biased clocks attempting to implement a given command velocity/acceleration as *implementation bias/error*.

Remark 1. *In light of these discussion of measurement/implementation errors, it's important to state that while we are suggesting such measurement and implementation biases are possible, they are by no means assured or required. There are certainly sensors (such as off-the-shelf accelerometers found in many electronics today) that make measurements of positions, velocities, or accelerations without necessitating a time interval be measured. In these cases, the clock implementation error will be nonexistent (that is, $\alpha = 1$, in the sense that it is equivalent to being implemented with respect to a perfect clock). In future results, we present what might be called the "worst case scenarios" for clock biases and implementations, but whenever possible, we will note when there are reasonable situations that errors will not be as severe, as well as strive to keep our analyses as general as possible to cover a wide range of applications/sensor configurations.*

3.2 Types of Agents

If we consider the dynamics presented in (2.8) and (2.10), along with the discussions in the previous section, we can immediately see that distinct clocks will lead to biased measurements and control inputs. So it is important to differentiate where the measurements are being generated for determining the control inputs. To help reason about and identify the possible combinations of communication and sensing available, consider the following scenarios for

which consensus might want to be employed.

Example 1: Laboratory Robotics: In Figure 3.1, we can see a pictorial representation of a multi-agent robotic platform (represented here by quadrotors) in a laboratory setting - that is, all agent position/velocity states are being tracked by a "global" observer (global in the sense that it can detect all activity within some bounded region of interest), such as a motion capture system. This setup is present in many university research settings, ranging in size from a couple of quadrotors to the massive scale of Robotarium [95].

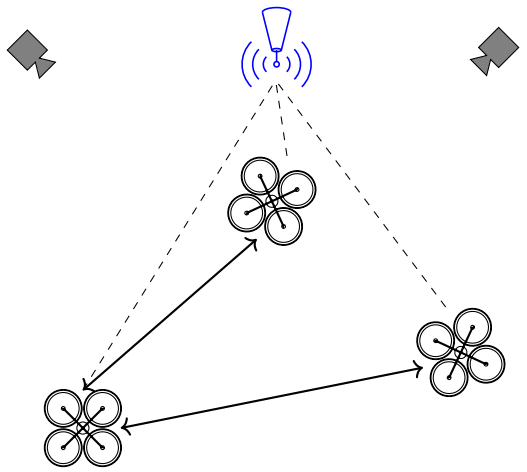


Figure 3.1: Robotic agents in a laboratory setting.

In this setting, agents can be very simple - possibly having no real ability to sense their own or neighbors' states, and with no actual agent-to-agent communication. However, communication topologies can be simulated by globally transmitting relevant information to all agents and selectively acting on neighbors' states. This is not entirely unrealistic outside of the laboratory setting, either, as indoors applications may have a centralized tracking system to provide robots with position information (for example, in automated warehouse applications). If we are considering agents within this system, the state information that they have access to may be a blend of global information with inertially navigation between updates (if updates are sparse).

Example 2: Self Driving Cars/Platooning: In Figure 3.2 we consider a group of autonomous vehicles that we might want to act in concert. Consensus applied to these types of systems can support activities such as platooning, which would be interpreted as a type of formation keeping between agents' positions and velocities.

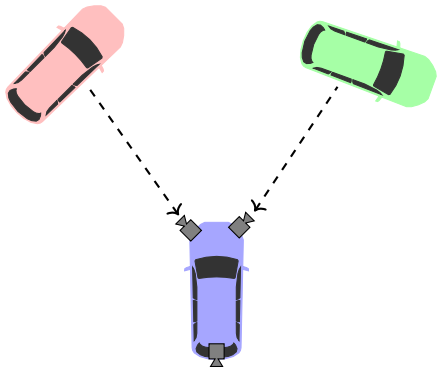


Figure 3.2: Autonomous Vehicles with Optical Sensors

However, in this case, there is no centralized/global observer of the system. If the cars are sufficiently "smart" they may be outfitted with sensors that allow them to identify and respond to other cars/hazards within some sensing radius. Thus, the information about other agents is determined solely by/with respect to an agent in question. Like in the previous example, there is no explicit communication of state information between agents (though various vehicle-to-vehicle information sharing protocols have been suggested in the context of autonomous cars for use in the not-too-distant future). Due to the assumed level of technology in each agent, however, we might assume that agent's self-states are able to be observed by onboard sensors, or inferred from observations of the environment.

Example 3: Satellite Attitude Alignment: Finally, in Figure 3.3 we consider a constellation or formation of satellite agents that might want to utilize the consensus protocol to align their pointings, as has been suggested multiple times in the literature [101, 26, 66]. These agents are sufficiently sensored such that they can measure/estimate their own attitudes and angular velocities, but do not have significant ability to measure those quantities

in their neighbors. However, assume that they can broadcast their states and receive information from neighbors in communication range. The communication range (possibly determined by orbits/timing, so may be static or dynamic) determines the network topology. So here we see that the consensus protocol could be implemented with agents' using their own measurements of self dynamics as well as those broadcasts received from their neighbors.

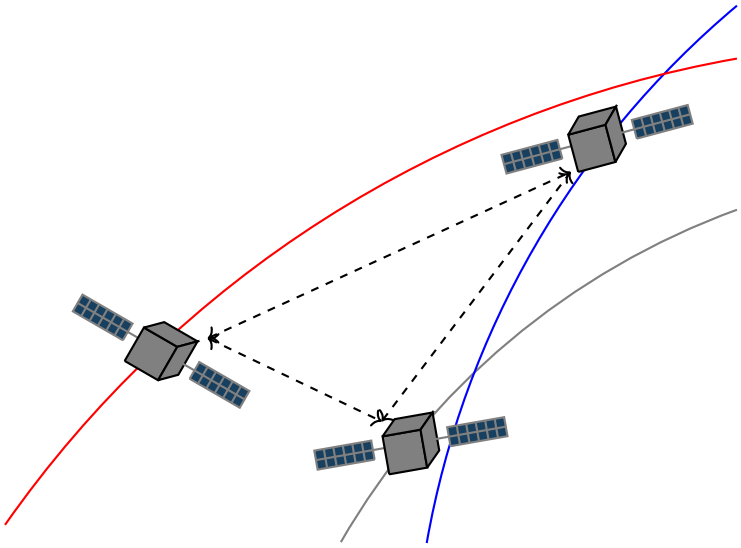


Figure 3.3: Satellite Agents with Line of Sight Communications

What is apparent in the previous examples is that we can conceivably devise scenarios where we seek to implement consensus using information from a variety of sources. In the satellite example, all neighbor information came from the neighbors themselves. In the autonomous vehicle example, all the information was passively gathered by the agents. In the laboratory setting, we can see that having global information allows for either interpretation to be valid. In light of these examples, we propose the following types of agents.

Definition 1. *Perceptive Agents: We call agents perceptive if at any time instant, agent i measures (perceives) the relevant state information of neighboring agents. Any relative or global state information measurement is made with respect to to agent i's internal clock.*

Definition 2. *Receptive Agents: We call agents receptive if they are unable to measure neighbor states, but can measure/estimate their own state, and broadcast a signal containing that information. At any time instant, agents can receive the state information as broadcast by their neighbors, and thus, these measurements are made with respect to the respective neighbors' clocks.*

Now let us reconsider (2.8) and (2.10) in light of the identified agent types and measurement restrictions.

3.3 Single Integrator Agents

In this section, we will detail the interesting cases for the single integrator agent case. We go into more detail in subsequent sections, but the resulting dynamics for each case is summarized in Table 3.1. First, we consider the case of agents with access to global state measurements.

Agent/Information Category	Resulting Consensus Dynamics
Perceptive or Receptive with Global Info	$\dot{x} = -\mathcal{S}\mathcal{L}x(t)$
Perceptive with Relative Info	$\dot{x} = -\mathcal{S}^2\mathcal{L}x(t)$
Receptive Agents with Inertial Navigation (RAIN)	$\dot{x}_{\mathrm{m}} = -\mathcal{S}\mathcal{L}x_{\mathrm{m}}(t)$ $\dot{x}_{\mathrm{t}} = -\mathcal{S}\mathcal{L}x_{\mathrm{m}}(t)$

Table 3.1: Resultant system dynamics for various manifestations of clock biases in single integrator consensus systems.

Perceptive/Receptive Agents with Global Measurements: If agents are able to make global position measurements (either in the perceptive or receptive agent case), then the relative state measurements are unbiased by any agent clocks. However, in both those cases, the control input still needs to be realized by individual agents, which as discussed in the previous

section, will be biased by individual clocks. Thus, for both perceptive and receptive agents with global state measurements, the single integrator consensus problem becomes,

$$\frac{dx_i}{d\tau_i} = \sum_{j\sim i} x_j - x_i$$
$$\frac{1}{\alpha_i}\dot{x}_i = \sum_{j\sim i} x_j - x_i$$

Bringing the clock skew parameter to the right-hand-side of the equation and putting into vector/matrix by stacking the individual state values into a vector of states $x \in \mathbb{R}^n$ gives,

$$\dot{x} = -\mathcal{S}\mathcal{L}x. \tag{3.1}$$

This resulting formulation, (3.1) has been investigated in the literature under the terminology of "scaled consensus" or "multi-rate consensus." It was first considered in [85], where the consensus value for a system of "multi-rate integrators" was determined. Since that time, there has been a slowly growing body of literature that addresses questions of convergence, controllability, and controller design for specific network topologies [93, 18, 98, 99]. However, to the best of our knowledge, before the work presented here, there has been little work that combined both agent-based clocks rates with arbitrary graph topologies and investigated how these impact the behavior of the network. As presented in Chapter 4.4, time scaling of agent dynamics changes some of the basic properties of the consensus problem, which motivates deeper study into the basic properties of networks of scaled agents.

For completeness, we also note that time scales in networked problems can also arise from the network's topology. This was of particular interest for large-scale power networks, and reduced order modeling based on singular perturbation techniques for both linear [23] and non-linear [7] network dynamics have been investigated. Network design and synthesis for specific time scale behavior have also been considered [104]. In much of the prior work

presented here, we do not consider time scale behavior that is the result of network topology, instead, we restrict our discussion to networks for which the multiple time scales are present in the individual agent dynamics as they interact through a linear consensus algorithm; however in Chapter 6.1.2 this idea will be central to the proposed work.

In the case where global measurements are not available to agents, the case of perceptive and receptive agents diverge. We detail the single-integrator ramifications for the two relevant cases below.

Perceptive Single Integrators with True Position Measurements

Proposition 1. *For agents capable of measuring true global positions of their own and neighbor states (independent of their internal clock), the resulting consensus dynamics are given by the scaled consensus problem given in* (3.1).

Proof. Under this framework, each single integrator agent can measure the true global position of all neighboring agents, independent of its internal clock. Thus, the calculated command input for the ith agent will be,

$$u_{i,\text{measured}} = \sum_{j\sim i} x_{j,\text{true}} - x_{i,\text{true}}$$

However, this command input will be subject to the implementation bias of each agent's clock,

$$u_{i,\text{true}} = \alpha_i \sum_{j\sim i} x_{j,\text{true}} - x_{i,\text{true}}.$$

Stacking the agent states and command inputs gives (3.1) for the true agent states. □

Remark 2. *It is perhaps seemingly unrealistic to assume agents that can measure global positions without relative measurements - however, this is commonly the case for laboratory settings where agents are precisely tracked (e.g. by a motion capture system) and communication between agents is simulated by transmitting the positions of neighbor groups to agents*

from a centralized position. This can be interpreted as each agent having the ability to measure those global positions independent from their clocks.

Perceptive Single Integrators with Relative Measurements: As previously discussed, for biased relative measurements, the control input for a perceptive agent will be equal to,

$$u_i = \alpha_i \sum_{j \sim i} x_j - x_i.$$

However, this will also be biased by implementation bias, so,

$$\begin{aligned} \frac{dx_i}{d\tau_i} &= \alpha_i \sum_{j \sim i} x_j - x_i \\ \frac{1}{\alpha_i} \dot{x}_i &= \alpha_i \sum_{j \sim i} x_j - x_i \\ \Rightarrow \dot{x}_i &= \alpha_i^2 \sum_{j \sim i} x_j - x_i \end{aligned}$$

Thus, we see an interesting result that the perception errors of relative sensing, and the biasing of implementing the velocity input compound, resulting in an amplification of the effects of any non-unity skew clocks.

Perceptive Single Integrators with Inertial Navigation: It is not clear how to consistently interpret $x_{j,\text{measured}}$ for perceptive agents in the dead reckoning frame work. If it is a true position, then it stands to reason that $x_{i,\text{measured}}$ is also an unbiased measure of the true position, which reverts the problem to the perceptive agent problem with true positions. Alternatively, if we blindly assume that agents can measure neighbor's true positions, but not their own, then the problem will be similar (but not identical) to the receptive agent/dead reckoning case. The dynamics in this case are given by,

$$\begin{aligned} \dot{x}_{i,\text{m}} &= \sum_{j \sim i} x_{j,\text{true}} - x_{i,\text{measured}} \\ \dot{x}_{i,\text{t}} &= \alpha_i \sum_{j \sim i} x_{j,\text{true}} - x_{i,\text{measured}}, \end{aligned}$$

where x_m and x_t denote the measured and true state, respectively, and $x_{i,\text{measured}}$ is the result of integrating $\dot{x}_m$, so is the dead reckoned position of each agent. Recalling that $\mathcal{L} = \Delta - \mathcal{A}$, we can note that the above dynamics can be rewritten as,

$$\begin{bmatrix} \dot{x}_m \\ \dot{x}_t \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & \mathcal{S} \end{bmatrix} \begin{bmatrix} -\Delta & \mathcal{A} \\ -\Delta & \mathcal{A} \end{bmatrix} \begin{bmatrix} x_m \\ x_t \end{bmatrix}.$$

This system is not stable, thus will not achieve consensus. Due to the inconsistency in the conception of such a system, it suffices to provide a simple example that exhibits instability (and not attempt to explore if there are instantiations that can be stabilized).

Claim 1. *The system defined above is not stable for all choices of $\mathcal{S}$.*

Proof. Consider the following instantiation that exhibits instability:

$$\mathcal{L} = \begin{bmatrix} 2 & 0 & -1 & -1 \\ 0 & 2 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix}$$

$$\mathcal{S} = \begin{bmatrix} 2.2 & 0 & 0 & 0 \\ 0 & 2.8 & 0 & 0 \\ 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 1.7 \end{bmatrix}$$

For these values, the system defined previously (after trivially decomposing $\mathcal{L}$ into the degree and adjacency matrices, has an eigenvalue $\lambda_p \simeq 0.67 > 0$, thus the system is unstable. □

Receptive Single Integrators with True Position Measurements: As discussed in the remark following the true position measurements in the perceptive agent case, the result can be interpreted identically for this case as well. However, we can note that the assumption

that each agent is capable of measuring its own true position independent of its clock and transmitting that to its neighbors is less offensive than the necessary assumption previously made.

Receptive Single Integrators with Relative Position Measurements: The receptive agent case is inconsistent with relative measurements, because in the receptive agent case all agents broadcast their measured positions - not relative positions.

Receptive Single Integrators with Inertial Navigation (RAIN): In this case, agents are reliant on inertial navigation for tracking their internal states, and then broadcast the estimates of those states to their neighboring agents. This naturally presents an interesting wrinkle, as due to the implementation bias due to distinct clocks, the internal estimate of an agent's state and the true position will diverge from a given "true" initial condition. We can see this by considering agents to have two states, an inaccessible state corresponding to their true state, and a clock-biased estimate that each agent reports to be their state (as that is what would be determined via their internal dead reckoning). Thus, each agent dynamics can be written as,

$$\begin{aligned}
\frac{d}{d\tau_i}\dot{x}_{i,\text{measured}} &= \sum_{j\sim i} x_{j,\text{measured}} - x_{i,\text{measured}} \\
\Rightarrow \dot{x}_{\text{measured}} &= -\mathcal{S}\mathcal{L}x_{\text{measured}}, \\
\dot{x}_{i,\text{true}} &= \alpha_i \sum_{j\sim i} x_{j,\text{measured}} - x_{i,\text{measured}} \\
\Rightarrow \dot{x}_{\text{true}} &= -\mathcal{S}\mathcal{L}x_{\text{measured}}.
\end{aligned} \tag{3.2}$$

where $x_{i,\text{measured}}$ denotes the estimated state that agent i is tracking as their own position. This is a bit of a misnomer, of course, as this state is not necessarily "measured," however, we adopt this labeling here to be consistent with the previous sections. This interesting result is summed up in the following proposition.

Proposition 2. *For single integrator dynamics, receptive agents with inertial navigation (RAIN case), the dynamics are self-correcting, in that the internal measured states do not*

diverge from the true states due to clock implementation bias.

Proof. This follows simply from the derivation of (3.2), which shows that both the true states and the measured states evolve with respect to a scaled Laplacian. We show this explicitly below, however. First, for brevity, denote x_{measured} and x_{true} as x_m and x_t, respectively. Then, we know that the general solution to the $x_m(t)$ dynamics is simply,

$$x_m(t) = e^{-\mathcal{S}\mathcal{L}t}x_m(0).$$

Thus, we have,

$$\begin{aligned} \dot{x}_t &= -\mathcal{S}\mathcal{L}e^{-\mathcal{S}\mathcal{L}t}x_m(0) \\ \Rightarrow x_t(t) &= x_m(0) - \int_0^t \mathcal{S}\mathcal{L}e^{-\mathcal{S}\mathcal{L}t}x_m(0)\mathrm{d}t, \end{aligned} \tag{3.3}$$

where we have used the fact that (by design) $x_t(0) = x_m(0)$. We can then simply solve (3.3),

$$\begin{aligned} x_t(t) &= x_m(0) - \int_0^t \mathcal{S}\mathcal{L}e^{-\mathcal{S}\mathcal{L}t}x_m(0)\mathrm{d}t \\ &= x_m(0) + \left(e^{-\mathcal{S}\mathcal{L}t} - I\right)x_m(0) \\ &= e^{-\mathcal{S}\mathcal{L}t}x_m(0). \end{aligned}$$

Thus, the true state trajectories are identical to the internal "measured" states. □

Now, an important distinction in light of the previous proposition must be made, however. When we say that the dynamics are self correcting, we are acknowledging that the true states will come to consensus, and that consensus value is equal to the measured consensus. However, the consensus value that is reached will be different from the consensus value that one might expect if the initial conditions were known *a priori*. In the next section, we assess this *error from predicted consensus* for this class of problem.

3.3.1 Bounding of Error due to Single Integrator Scaled Consensus Problems

In all of the single integrator cases, we found that the resulting problem was a scaled consensus problem of the form,

$$\dot{x}(t) = -\mathcal{S}^p \mathcal{L} x(t),$$

where we have denoted the true agent states simply as $x(t)$, and p is the exponent of the scaling matrix - which we observed to either be 1 or 2 depending on the case being considered. It is obvious that for a given initial condition, the resulting trajectory of the appropriate scaled problem will deviate from the unscaled problem. Thus, if given an application for which we are ignorant of clock biases and we attempt to implement consensus, there will be a discrepancy between the true consensus value and that which may have been predicted. We see this in action in Figure 3.4.

The error between the true agent states and the naïve prediction can be simply stated in the next proposition.

Proposition 3. *The error between the scaled consensus state and the naïve prediction consensus state is,*

$$\Delta = \left| \frac{\mathbf{1}^T x(0)}{n} - \frac{\mathbf{1}^T \mathcal{S}^{-p} x(0)}{\mathbf{1}^T \mathcal{S}^{-p} \mathbf{1}} \right| \tag{3.4}$$

Proof. The existence of the zero eigenvalue and marginal stability about the scaled consensus subspace for the scaled Laplacian matrix is formally detailed in Section 4.4, so we will simply posit those facts here. Given that, we simply note that the left and right eigenvectors associated with the zero eigenvalue of the scaled Laplacian are w^T and $\mathbf{1}$, where we define w such that,

$$S^{-p} = \text{diag}(w),$$

that is, $[w]_i = \alpha_i^{-p}$. Then we can easily see that the consensus value for the scaled consensus dynamics is given by,

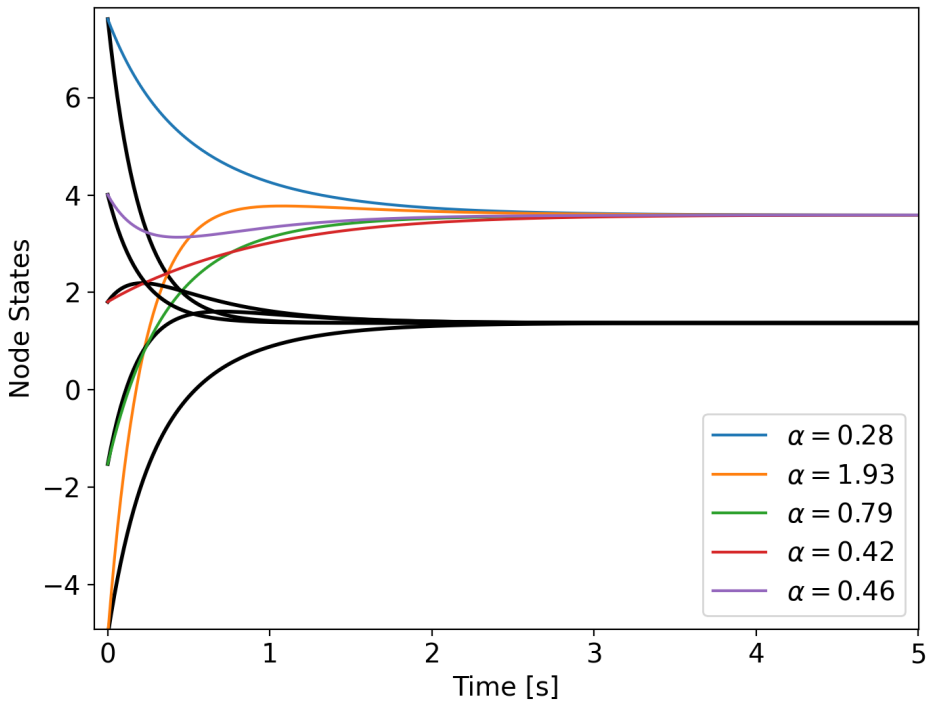


Figure 3.4: Difference in equilibrium state of scaled (colored lines) and unscaled consensus (black lines). For non-unity clock skew parameters non-zero error between the naïve predicted consensus value and the true consensus value is possible.

$$x_{c,s} = \frac{\mathbf{w}^T x(0)}{\mathbf{w}^T \mathbf{1}} = \frac{\mathbf{1}^T \mathcal{S}^{-p} x(0)}{\mathbf{1}^T \mathcal{S}^{-p} \mathbf{1}}.$$

Then, recalling that the un-scaled consensus problem has consensus value given by (2.11) (denote by x_c), taking the absolute value of $x_c - x_{c,s}$ gives the intended result. □

Remark 3. *It is natural to consider if there are any network design considerations that can be exploited to minimize the total spread between true consensus and the naïve prediction - thus minimizing the practical effects of the different clock rates. However, we can note from (3.4) that the steady state value of the true trajectories (and thus, the total magnitude of true error from naïve consensus), is determined by the initial conditions alone, thus, is independent of the network topology.*

3.4 Double Integrator Agents

Agent/Information Category	Resulting Consensus Dynamics
Perceptive or Receptive with Global Info	$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & I \\ -\mathcal{S}^2\mathcal{L} & -\mathcal{S}^2\mathcal{L} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}$
Perceptive with Relative Info	$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & I \\ -\mathcal{S}^3\mathcal{L} & -\mathcal{S}\mathcal{L} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}$
Receptive Agents with Inertial Navigation (RAIN)	$\begin{bmatrix} \dot{x}_m \\ \dot{v}_m \\ \dot{x}_t \\ \dot{v}_t \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathcal{S} & \mathbf{0} & \mathbf{0} \\ -\mathcal{S}\mathcal{L} & -\mathcal{S}\mathcal{L} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I \\ -\mathcal{S}^2\mathcal{L} & -\mathcal{S}^2\mathcal{L} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} x_m \\ v_m \\ x_t \\ v_t \end{bmatrix}$

Table 3.2: Resultant system dynamics for various manifestations of clock biases in double integrator consensus systems.

We will now consider the previous scenarios for the double-integrator agent dynamics case. We start with the assumption of perceptive agents, though, as in the previous section, we summarize the results for the interesting cases in Table 3.2.

Perceptive Agents, True Positions and Velocities: When true positions and velocities can be measured independent of the internal clock, the only clock-based error is due to implementation. Thus, the resulting dynamics take the form,

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & I \\ -\mathcal{S}^2\mathcal{L} & -\mathcal{S}^2\mathcal{L} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}. \tag{3.5}$$

Proposition 4. *For the dynamics in (3.5), the equilibrium position is given by,*

$$\begin{bmatrix} \bar{x} \\ \bar{v} \end{bmatrix} = (\bar{\omega}^T \otimes e^{Pt}) \begin{bmatrix} s_1(0) \\ \vdots \\ s_n(0) \end{bmatrix}$$

where $s_i(0) = [x_i(0), v_i(0)]^T$, $\bar{\omega}$ *is the normalized left eigenvector of* $\mathcal{S}^2\mathcal{L}$ *corresponding to the 0 eigenvalue, and,*

$$P = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Proof. When the position and velocity states are evolving at the same rates (here given by $\mathcal{S}^2$ matrix), then they share the left eigenvector $\omega = [\alpha_1^{-2}, \ldots, \alpha_n^{-2}]^T$, thus, $\bar{\omega} = \omega / \mathbf{1}^T\omega$. With this shared, unit norm, eigenvector, the result follows directly from [120, Theorem 1]. □

Remark 4. *Like in the single-integrator case, when agents have access to true positions/velocities, the perceptive and receptive agent cases are identical. Also as in the single-integrator case, the assumption of true position measurements being available to all agents is more realistic in the receptive agent case, but as they are equivalent we present both interpretations here.*

Perceptive Agents, Relative Measurements: Now, assume here that agents are perceptive, and that the position and velocity measurements are relative measurements that have been

biased by the agents' clocks as previously discussed. This can be understood by the following series of control law updates,

$$\begin{aligned} u_{i,\text{ideal}} &= \sum_{j\sim i} (x_{j,\text{true}} - x_{i,\text{true}}) + \sum_{j\sim i} (\dot{x}_{j,\text{true}} - \dot{x}_{i,\text{true}}) \\ u_{i,\text{measured}} &= \sum_{j\sim i} (x_{j,\text{measured}} - x_{i,\text{measured}}) \\ &\quad + \sum_{j\sim i} (\dot{x}_{j,\text{measured}} - \dot{x}_{i,\text{measured}}) \\ &= \alpha_i \sum_{j\sim i} (x_{j,\text{true}} - x_{i,\text{true}}) + \frac{1}{\alpha_i} \sum_{j\sim i} (\dot{x}_{j,\text{true}} - \dot{x}_{i,\text{true}}) \,. \end{aligned}$$

Agents in the network will attempt to implement $u_{i,\text{measured}}$, but will be further hampered by the implementation error, that is, $\dot{v}_i = \ddot{x}_i = \alpha_i^2 (d^2 x_i / d\tau^2)$. Thus, the perceptive agent case resolves to,

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} &= \begin{bmatrix} \mathbf{0} & I \\ -\mathcal{S}^3\mathcal{L} & -\mathcal{S}\mathcal{L} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}, \\ &:= A_D \begin{bmatrix} x \\ v \end{bmatrix}. \end{aligned} \tag{3.6}$$

This resulting set of dynamics is similar to the single-integrator scaled case, but the position and velocity states have different relative scales associated with them, which diverges from the formulations that have previously been investigated. Below, we investigate the properties of this model.

Theorem 2. *For perceptive, double integrator agents with dynmics given by (3.6), the clock-biased consensus dynamics achieve a global equilibrium about the velocity states. Formally, for $x(0), v(0) \in \mathbb{R}^n$, with $\mathcal{S} \succ 0$ and where $\mathcal{L}$ is a undirected Laplacian matrix for a connected network (Assumption 1), the velocity states for (3.6) acheive the following consensus value,*

$$v_c = \frac{\mathbf{1}^T \left(\mathcal{S}^{-2}\mathcal{L}x(0) + \mathcal{S}^{-3}v(0)\right)}{\mathbf{1}^T \mathcal{S}^{-3}\mathbf{1}}$$

Proof. To prove the specific result in question, we can first note that we can generalize this problem to finding the equilibrium velocity for systems defined by,

$$\tilde{A}_D = \begin{bmatrix} \mathbf{0} & I \\ -\mathcal{S}^r\mathcal{L} & -\mathcal{S}^p\mathcal{L} \end{bmatrix}$$

We can see that the A_D defined by (3.6) is simply this general case with $r = 3$ and $p = 1$. Now, observe that the left eigenvector associated with the zero eigenvalue of $\tilde{A}_D$ is given by,

$$\nu^T = \begin{bmatrix} \mathbf{1}^T\mathcal{S}^{p-r}\mathcal{L} & \mathbf{1}^T\mathcal{S}^{-r} \end{bmatrix}.$$

Note that,

$$\frac{d}{dt}\left(\nu^T \begin{bmatrix} x(t) \\ v(t) \end{bmatrix}\right) = \nu^T \begin{bmatrix} \dot{x}(t) \\ \dot{v}(t) \end{bmatrix} = \nu^T\tilde{A}_D = \mathbf{0},$$

thus, any quantity $\nu^T[x^T(t)\ v^T(t)]^T$ is invariant. Given this, we can equate it at the initial time and in the steady state,

$$\mathbf{1}^T\mathcal{S}^{p-r}\mathcal{L}x(0) + \mathbf{1}^T\mathcal{S}^{-r}v(0) = \mathbf{1}^T\mathcal{S}^{p-r}\mathcal{L}x_{ss} + \mathbf{1}^T\mathcal{S}^{-1}v_c\mathbf{1}$$

where we have denoted the steady state position x_{ss}, and the consensus value of the velocity states as v_c. What we can note is that in consensus, all agent positions will be equal (though evolving in time at the consensus velocity if non-zero), so $\mathcal{L}x_{ss} = \mathbf{0}$ as x_{ss} is in $span\mathbf{1}t$. Then, we can solve the above equation for the consensus velocity value,

$$v_c = \frac{\mathbf{1}^T\left(\mathcal{S}^{p-r}\mathcal{L}x(0) + \mathcal{S}^{-r}v(0)\right)}{\mathbf{1}^T\mathcal{S}^{-r}\mathbf{1}} \tag{3.7}$$

Then, our desired result for the (3.6) is simply substituting in the known values of $r = 3$ and $p = 1$. □

Remark 5. *In the proof of Theorem 2, we abstracted the exponents of the skew parameter matrices and solved the general case. This is because, while we have presented what we feel is*

one possible interpretation for the derivation of the (3.6) dynamics, there are other reasonable assumptions of how clocks might bias individual measurements and implementations of those control inputs. For example, many accelerometers do not need a clock reading to discern the acceleration (it is measured via a displacement/the force required to resist a displacement of a test mass).[1] *Thus, in those applications, the implementation bias is nullified, and taking* $r = 1$ *and* $p = -1$ *would be a valid interpretation of clock biased dynamics. Alternatively, sensors such as Doppler radar can measure relative velocity without a clock reading, so another justified choice for our exponents are* $r = 1$ *and* $p = 0$*. Thus, the general result provided in the proof covers many different sensor configurations.*

Receptive Agents with Inertial Navigation (RAIN): This case is very similar to the single-integrator case, in that each agent will now have an internal state (position and velocity), which is updated via dead reckoning and the reported measured states of its neighbors. However, in reality, the agents internal "measured" states will evolve with respect to the individual clock rates, and the true velocity dynamics will be subject to implementation errors. Thus, we can write the internal and true dynamics as,

$$\begin{bmatrix} \dot{x}_m \\ \dot{v}_m \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathcal{S} \\ -\mathcal{S}\mathcal{L} & -\mathcal{S}\mathcal{L} \end{bmatrix} \begin{bmatrix} x_m \\ v_m \end{bmatrix}$$
$$\dot{v}_t = -\mathcal{S}^2\mathcal{L}x_m(t) - \mathcal{S}^2\mathcal{L}v_m(t)$$
$$\dot{x}_t = v_t(t),$$

which can be put into a more compact form as,

[1] However, we can note that even in applications that typically enjoy instrumentation for measuring velocities and accelerations with accelerometers/IMUs (such as satellite platforms), methods for measuring those quantities without such instrumentation have been explored [11, 43]. Utilization of clock biased velocity or acceleration state measurement can be seen as either a "worst case" sensor case, or equally applying to non-traditionally-sensored platforms.

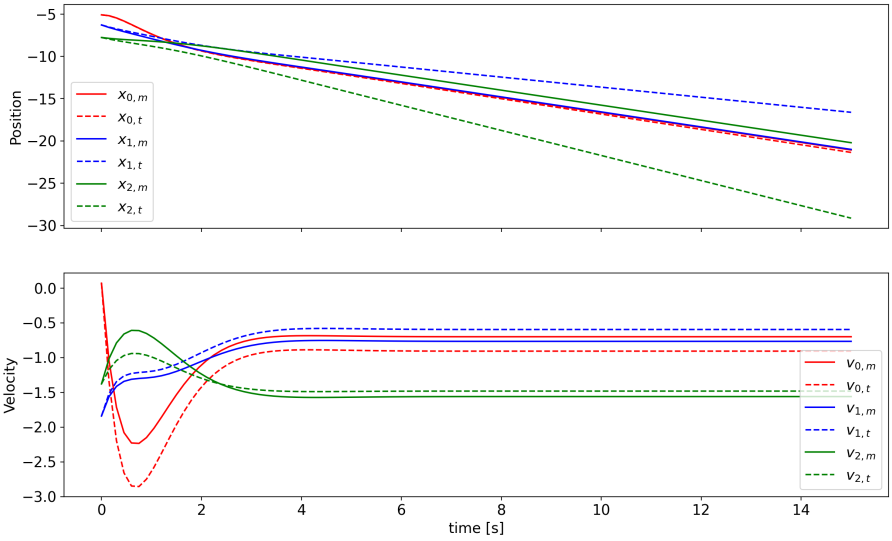


Figure 3.5: Failed consensus on velocity and position for the receptive, double integrator agent case. This is due to both implementation errors, as well as the intrinsic clock rate of the internal inertial navigation. Note that the internal states fail to achieve consensus, thus, this is the source of the true states lack of consensus as well.

$$\begin{bmatrix} \dot{x}_m \\ \dot{v}_m \\ \dot{x}_t \\ \dot{v}_t \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathcal{S} & \mathbf{0} & \mathbf{0} \\ -\mathcal{S}\mathcal{L} & -\mathcal{S}\mathcal{L} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I \\ -\mathcal{S}^2\mathcal{L} & -\mathcal{S}^2\mathcal{L} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} x_m \\ v_m \\ x_t \\ v_t \end{bmatrix} \tag{3.8}$$

In Figure 3.5, we see the result of a simulation of (3.8) for a simple path graph on three agents. Even in this simple case, the agents clock-biased implementation of the velocity control leads to a failure to achieve any consensus on either velocity or position. In Chapter 5 we will propose some strategies for attempting to correct these types of errors.

Remark 6. *As discussed previously, there are common sensors that detect both linear and*

angular acceleration without a clock bias. Thus, it could be argued that a non-worst case version (that is still justified) of the dynamics are such that,

$$\begin{bmatrix} \dot{x}_m \\ \dot{v}_m \\ \dot{x}_t \\ \dot{v}_t \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathcal{S} & \mathbf{0} & \mathbf{0} \\ -\mathcal{S}\mathcal{L} & -\mathcal{S}\mathcal{L} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I \\ -\mathcal{L} & -\mathcal{L} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} x_m \\ v_m \\ x_t \\ v_t \end{bmatrix}. \tag{3.9}$$

It is a valid question, then, if this resolves the lack of consensus observed in the simulation presented in Figure 3.6. We simulate (3.9) and present the results for random initial conditions and skews randomly sampled from a uniform distribution between $[0.5, 1.5]$*, and can observe that the results are virtually the same. As proposed, it stands to reason that the source of the lack of consensus is the disparate clock rates in the inertial navigation dynamics.*

3.5 Chapter Summary

In this chapter, we have investigated the ways that individual agent clocks can manifest in single and double integrator dynamics. We've seen that the presence of clock implementation bias can cause consensus error from the predicted consensus value in all of the single integrator cases. In the double integrator case, for global and relative information availability, we saw that consensus was achieved, albeit at consensus states that were biased from the unscaled versions (like in the single integrator case). More troubling, however, was the RAIN case, which can showed that the different rates of internal navigation impeded consensus from forming on any of the velocity or position states. More generally, we've seen that the inclusion of differing clocks can be interpreted as agent's dynamics being defined on various time scales, which we will investigate further in the next chapter.

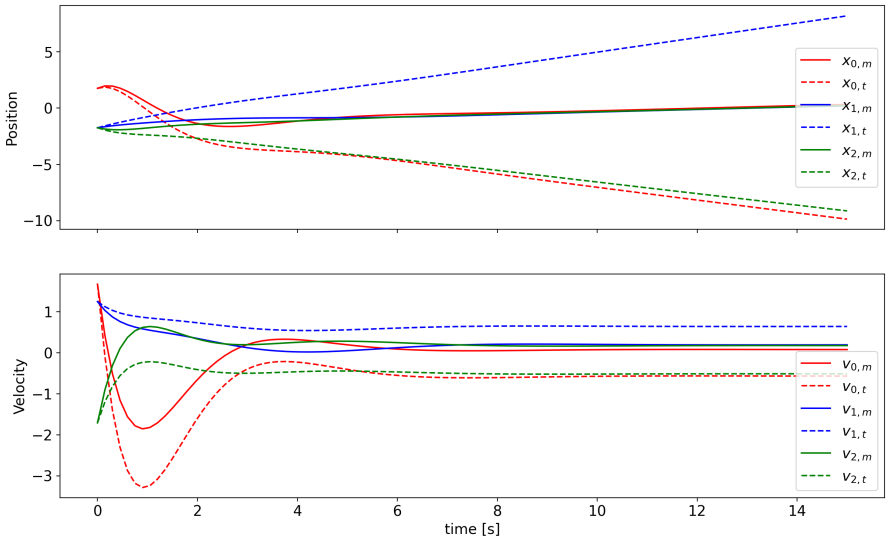


Figure 3.6: Failed consensus in the case double integrator RAIN case, when accounting for no acceleration implementation error. Thus, we can see that (as assumed), the source of the lack of consensus is the unique clock rates in the inertial navigation.

Chapter 4

SCALED CONSENSUS PROBLEMS AND PERFORMANCE

Having established the necessary mathematical background in Chapter 2, here we dive into some work that has been undertaken towards the broad goals outlined in the introduction. We will start in §4.1 with work that proposes a reduced order modeling procedure for systems in non-standard form, with an application to a consensus system. This work serves as a preliminary work in reduced order modeling, but more conceptually, serves as motivation for studying control of systems via actuators or agents that are slower than some reference time scale. In the later two sections,Section 4.2 and Section 4.4, we will more directly tackle performance analysis of consensus systems under noise inputs, as well as single-input signals. These works serve to establish some of the interesting ways that scaled graph problems differ from their mono-scale progenitors, as well as motivate some of the future work proposed in later chapters.

In the context of the classification of clock-biased consensus problems presented in Chapter 3, all of the results contained within this chapter fall under the class of problems where it is assumed that all agents have access to the true global/relative positions of their neighbors, so the clock bias is introduced due to the implementation bias on the control input. However, as mentioned in Chapter 2, here we will be discussing results in terms of the time scale parameters - so the scaling factors in those formulations will take the values of the inverse of the time scale parameters utilized here.

4.1 Reduced Order Modeling for Slowly Actuated Systems[1]

Consider the closed loop behavior of a singularly perturbed dynamic system under the influence of slow actuation, formulated as

$$\begin{aligned} \dot{x} &= f(x,z),\ x(t_0) = x_0 \\ \dot{z} &= \varepsilon g(x,z),\ z(t_0) = z_0, \end{aligned} \tag{4.1}$$

where the $x \in \mathbb{R}^n$ is the plant state, $z \in \mathbb{R}^m$ is the actuator state, $f(\cdot)$ and $g(\cdot)$ describe the plant and actuator dynamics, respectively, ε is a small, positive parameter, and the initial conditions (x_0, z_0) are not functions of ε.

The two time scale nature of (4.1) can be rigorously examined via the method of dominant balance [6], but is also apparent by considering the orders of the corresponding derivatives. The vector fields $f(\cdot)$ and $g(\cdot)$ are taken to be $\sim \mathcal{O}(1)$; see [56] for the definition of "Big-O" order notation used throughout the report. Thus $\dot{x} \sim \mathcal{O}(1)$ while $\dot{z} \sim \mathcal{O}(\varepsilon)$. As $\varepsilon \ll 1$, the actuator dynamics will evolve along a slower time scale than those of the plant. The "fast" and "slow" time scales can be defined as t and $T = \varepsilon t$, respectively. Differentiation with respect to two time scales are related by the chain rule,

$$\frac{d}{dt} = \frac{dT}{dt}\frac{d}{dT} = \varepsilon \frac{d}{dT} \tag{4.2}$$

Behavior on distinct time scales is a defining characteristic of singularly perturbed systems, for which there is a rich theoretical background supporting the generation of asymptotically valid reduced order models [56, 58, 6].

4.1.1 Reduced order model generation and validation

In the previous section it was shown that (4.1) exhibits behavior on a fast scale (t), and a slow scale $T = \varepsilon t$. In general, combining approximations valid in different time scales,

[1] Text in this section adapted from "Reduced Order Modeling with Slow Actuators with Application to Leader-Follower Satellite Constellations," presented at the 2018 American Control Conference, Milwaukee, WI, USA [37].

or generating a single approximation that is valid over multiple time scales is a nontrivial endeavor [56, Chp. 1.6-1.10]; a variety of methods from singular perturbation theory allow for such approximations to be constructed [80]. The following method generates a reduced order model of the dynamics (4.1) that approximates the full system behavior over both fast and slow time scales.

Step 1: Assume that the approximate solutions to the dynamics (4.1) are composites of the independent behaviors on each time scale,

$$x_c(t,T) = \alpha(t) + X(T) \tag{4.3}$$

$$z_c(t,T) = \beta(t) + Z(T), \tag{4.4}$$

where $\alpha(t)$ and $\beta(t)$ capture the behavior of the plant and actuator on the fast time scale and $X(T)$ and $Z(T)$ capture the behaviors on the slow time scale. Intuitively, if the plant behaviors over the fast time scale settle into an equilibrium, the slow influence of the actuator can drive the plant from that equilibrium into a new state.

Step 2: Substitute (4.3) and (4.4) into (4.1) by utilizing (4.2),

$$\begin{aligned} \frac{d\alpha}{dt} + \varepsilon\frac{dX}{dT} &= f(\alpha + X, \beta + Z) + \mathcal{O}(\varepsilon) \\ \frac{d\beta}{dt} + \varepsilon\frac{dZ}{dT} &= \varepsilon g(\alpha + X, \beta + Z) + \mathcal{O}\left(\varepsilon^2\right). \end{aligned} \tag{4.5}$$

Step 3: Equate first order terms in (4.5) to give the $\mathcal{O}(1)$ equations,

$$\frac{d\alpha}{dt} = f(\alpha + X, \beta + Z) \tag{4.6}$$

$$\frac{d\beta}{dt} = 0. \tag{4.7}$$

If plant dynamics on the fast time scale are unstable, then on time scales of $t \sim \mathcal{O}(1/\varepsilon)$ the fast plant dynamics (4.6) will be of $\mathcal{O}(1/\varepsilon)$ (from (4.2)), which will dominate the $\mathcal{O}(1)$ slow time scale dynamics in that regime. Thus, assume the origin is a stable equilibrium of (4.6). In this steady state, the plant dynamics are satisfied by $X(T)$ alone, $0 = f(X(T), \beta(t) + Z(T))$, and a manifold, $\phi(z_c)$, can be defined which is dependent only on the actuator state.

This manifold describes the slow time scale behavior of the plant from the equilibrium of the fast plant dynamics,

$$X(T) = \phi\left(\beta(t) + Z(T)\right). \tag{4.8}$$

The $\mathcal{O}(\varepsilon)$ term in (4.5) then gives the evolution of $Z(T)$ to be

$$\frac{dZ}{dT} = g(\alpha + X, \beta + Z). \tag{4.9}$$

Step 4: From the initial conditions on (4.1) the initial conditions for (4.6), (4.7), (4.9), and (4.8) can be found. For the actuator, the initial condition follows from (4.4), $z_c(t_0, \varepsilon t_0) = \beta(t_0) + Z(\varepsilon t_0)$. Using (4.7), $\beta(t) = \beta(t_0) = \beta_0$. Thus, arbitrary (but appropriate) values for β_0 and $Z(\varepsilon t_0)$ can be chosen to satisfy the equation for $z_c(t_0, \varepsilon t_0)$. It is convenient to take the initial slow scale actuator dynamics to be zero, which then specifies,

$$Z(\varepsilon t_0) = 0, \;\; \beta_0 = z_0. \tag{4.10}$$

This choice of β_0 is in line with the intuition behind (4.4): the actuator is slow, so the only influence it can exert in the fast time scale regime ($\beta(t)$) is the initial actuator state.

The initial conditions for the components of the actuator approximation also give the initial value of the plant's slow manifold, $X(\varepsilon t_0) = \phi(t_0) = \phi(\beta_0)$, which then allows the initial condition on $\alpha(t)$ to be specified from (4.3),

$$\alpha(t_0) = x_0 - \phi(t_0). \tag{4.11}$$

Step 5: To summarize, Steps 1-4 result in the following reduced order model of (4.1)

$$\dot{\alpha}(t) = f(\alpha(t) + \phi(\beta_0 + Z(T)), \beta_0 + Z(T)), \tag{4.12}$$

$$\alpha(t_0) = x_0 - \phi(t_0)$$

$$\frac{dZ}{dT} = g(\alpha(t) + \phi(\beta_0 + Z(T)), \beta_0 + Z(T)), \tag{4.13}$$

$$Z(\varepsilon t_0) = 0$$

$$X(T) = \phi(\beta_0 + Z(T)) \tag{4.14}$$

$$\beta(t) = \beta_0 = z_0. \tag{4.15}$$

The advantage offered by this set of equations is that they can be solved procedurally: first solve (4.13), which then gives the analytic form of $\phi(t)$ from (4.14). Finally, use the previous results to solve for $\alpha(t)$ from (4.12).

Step 6: Finally, apply (4.3) and (4.4) to construct the full approximations to the true solutions of (4.1).

Generating the reduced order model in (4.12)-(4.15) by the preceding steps allows for the behavior of both the plant and the actuator to be generated from the vector fields of the full system. Furthermore, the reduced order model generated provides information about the response on both time scales while simultaneously offering a composite approximation that is valid over both. In the following theorem, the asymptotic validity of these approximations is stated.

Theorem 3. *Like in Theorem 1, here we wish to establish that there exists a critical value of ϵ, below which we are assured that the reduced order models of the closed loop dynamics (4.1) are valid approximations to the full solution and can be used for modeling/analysis purposes. We state this formally below.*

Consider the closed loop dynamics presented in (4.1), and let $\psi(z)$ be an isolated root of the plant dynamics satisfying $0 = f(\psi(z), z)$. Assume that the following conditions are satisfied for all

$$[t, x - \psi(z), z, \varepsilon] \in [t_0, t_1/\varepsilon] \times D_x \times D_z \times [0, \varepsilon_0],$$

for some domains $D_x \subset \mathbb{R}^n$ and $D_z \subset \mathbb{R}^m$, in which D_x contains the origin:

C1) f and g, and their first partial derivatives with respect to (x, z), and the first partial of f with respect to t are continuous, and the Jacobian of f has continuous first partial derivatives with respect to its arguments. Furthermore, the Jacobian $[\partial f/\partial x]$ has bounded first derivatives with respect to (t, z), so that the Jacobian matrices $[\partial f/\partial t]$ and $[\partial f/\partial z]$ satisfy:

$$\left\| \frac{\partial f}{\partial t} \right\| \leq L_1 \|x\|, \quad \left\| \frac{\partial f}{\partial z} \right\| \leq L_2 \|x\|$$

for some constants L_1 and L_2.[2]

C2) There is a unique solution to (4.12) over D_x, and the origin is an exponentially stable equilibrium point of (4.12) uniformly in (t, z).

Then there exists a positive constant ε^* such that for $0 < \varepsilon < \varepsilon^*$, the singularly perturbed dynamics (4.1) has unique solutions $x(t)$ and $z(t)$ on $[t_0, t_1/\varepsilon]$ and

$$x(t) = x_c(t, \varepsilon t) + \mathcal{O}(\varepsilon)$$
$$z(t) = z_c(t, \varepsilon t) + \mathcal{O}(\varepsilon)$$

hold uniformly over $t \in [t_0, t_1/\varepsilon]$, where $x_c(t)$ and $z_c(t)$ are the approximate solutions defined in (4.3) and (4.4) with dynamics (4.12)-(4.15), and $D_x \subset \mathbb{R}^n$ and $D_z \subset \mathbb{R}^m$ represent the permissible plant and actuator states, respectively.

Proof. To prove Theorem 3, the conditions C1 and C2 are first used to define a Lyapunov function for the plant's fast time scale dynamics, which can be used to bound the true solutions to the plant dynamics in (4.1). From the definition of the error between the true actuator solution and its approximation, along with the true plant solution bound, the error in the actuator's state can be bounded. For the plant, an expression of the error dynamics can be found and written as a perturbation of (4.12), which allows the plant error to be bounded. For brevity, the full composite approximations are denoted as x_c and z_c as defined in (4.3) and (4.4).

Recall that in order to find the slow plant manifold (4.8), (4.6) was assumed to have a stable equilibrium at the origin *uniformly* in (t, z_c), as codified in C2. From the definition of exponential stability,

$$\|\alpha(t)\| \leq k \|\alpha(t_0)\| \exp(-\gamma(t - t_0)) \ \forall \ t \geq t_0, \tag{4.16}$$

[2]The notation $\|\cdot\|$ refers to the Euclidean norm.

for all $(\alpha, z_c) \in D_0 \times D_z$, where $D_0 = \{\alpha \in D_x | \, \|\alpha\| < \rho_0\}$ represents a region of interest. From C1 and (4.16), [58, Lemma 9.8] a Lyapunov function $V_1(\alpha, \zeta)$ exists that satisfies

$$c_1 \|\alpha\|^2 \leq V_1(\alpha, \zeta) \leq c_2 \|\alpha\|^2, \tag{4.17}$$

$$\frac{\partial V_1}{\partial \alpha} f(\alpha, \zeta) \leq -c_3 \|\alpha\|^2, \tag{4.18}$$

$$\left\| \frac{\partial V_1}{\partial \alpha} \right\| \leq c_4 \|\alpha\|, \ \left\| \frac{\partial V_1}{\partial \zeta} \right\| \leq c_5 \|\alpha\|^2, \tag{4.19}$$

where c_i, $i = 1, 2, \ldots, 5$ are positive constants independent of ζ, and the inequalities (4.17) and (4.18) hold uniformly for $\zeta \in D_z$.

Now, consider the behavior of the full system in the regime of $t \sim \mathcal{O}(1)$. The change in the actuator state in this regime is small (as stated in Step 4 in §4.1.1): for bounded (x, z) on a finite time interval, $g(x, z)$ will also be bounded. Thus, $\|\dot{z}\| \leq \varepsilon \|g(x, z)\| \leq \varepsilon\kappa$. This allows the full plant dynamics to be taken as a system with a slowly varying parameter. Now perform a coordination transformation $q = x - \psi(z)$ so that the origin is the equilibrium of the dynamics defined by $\dot{q} = f(q + \psi(z), z) + (d\psi/dz)\dot{z}$. From the continuity of f, $\phi(z)$ is locally Lipschitz, so $\|d\psi/dz\| \leq L$ for some $L > 0$. As such, at any time $t^* \geq t_0$, if $\|q(t^*)\| < \rho_0\sqrt{c_2/c_1} = \mu$, the solution $q(t)$ of the above dynamics will obey the exponential bound,

$$\|q\| \leq \mu\sqrt{\frac{c_2}{c_1}} \exp(-a_1(t - t^*)) + \varepsilon\delta_1, \text{ for } t^* \leq t \leq t_1, \tag{4.20}$$

where $a_1 = 1/2\,(c_3/c_2 - \varepsilon c_5/c_1)$ and $\delta_1 = c_4 L/2c_1 a_1$, [58, Theorem 9.3]. Now, recall that $x(t_0) = x_c(t_0)$ by (4.3) and (4.11), so that

$$\|\alpha(t_0)\| = \|x_c(t_0) - \phi(z_c(t_0))\| = \|x_0 - \phi(z_c(t_0))\| < \rho_0,$$

where the last inequality follows from (4.17). From (4.4) and (4.10), $z(t_0) = z_c(t_0)$, which implies that,

$$\begin{aligned} 0 &= f(\psi(z(t_0)), z(t_0)) \\ &= f(\psi(z_c(t_0)), z_c(t_0)) = f(\phi(z_c(t_0)), z_c(t_0)). \end{aligned}$$

Therefore, $\phi(z_c(t_0)) = \psi(z(t_0))$, which can be combined with (4.17), leading to

$$\|\alpha(t_0)\| = \|x_0 - \psi(z(t_0))\| = \|q(t_0)\| < \rho_0 \leq \rho_0 \sqrt{\frac{c_2}{c_1}}.$$

This establishes that $\|q(t^*)\| < \rho_0\sqrt{c_2/c_1}$ is satisfied for $t^* = t_0$, so (4.20) holds in the regime where $\|\dot{z}\| \leq \kappa\varepsilon$ and

$$\|q\| = \|x(t) - \psi(z)\| \leq k_1 \exp(-a_1(t - t_0)) + \varepsilon\delta_1 \tag{4.21}$$

for all $t_0 \leq t \leq t_1$ and $k_1 > 0$.

Now, define the actuator error as $\lambda(t) = z(t) - z_c(t)$, which can be written as an integral expression

$$\begin{aligned}
\lambda(t) &= z(t) - z_c(t) \\
&= z(0) - z_c(0) + \int_{t_0}^{t} \varepsilon g(x(s), z(s)) - \varepsilon g(\phi, z_c(s))\mathrm{d}s \\
&= \varepsilon \int_{t_0}^{t} \left[g(x(s), z(s)) - g(\psi(z(s)), z(s))\right] \mathrm{d}s \\
&\quad + \varepsilon \int_{t_0}^{t} \left[g(\psi(z(s)), z(s)) - g(\phi(z_c(s)), z_c(s))\right] \mathrm{d}s.
\end{aligned}$$

From this expression the norm of the error can be bounded as,

$$\begin{aligned}
\|\lambda(t)\| &\leq \varepsilon \int_{t_0}^{t} \|x(s) - \psi(z(s))\| \, L_1 \mathrm{d}s \\
&\quad + \varepsilon \int_{t_0}^{t} \|z(s) - z_c(s)\| \, L_2 + \|\psi - \phi\| \, L_3 \mathrm{d}s \\
&\leq \varepsilon \int_{t_0}^{t} \|x(s) - \psi\| \, L_1 + \varepsilon L_3 \mathrm{d}s + \varepsilon \int_{t_0}^{t} \|\lambda(s)\| \, L_2 \mathrm{d}s \\
&\leq \varepsilon \int_{t_0}^{t} \left[\varepsilon(\delta_1 + \frac{L_3}{L_1}) + k_1 \exp(-a_1(t - t_0)) \right] L_1 \mathrm{d}s \\
&\quad + \varepsilon \int_{t_0}^{t} \|\lambda(s)\| \, L_2 \mathrm{d}s \\
&\leq \varepsilon \left(\varepsilon(\delta_1 L_1 + L_3)(t - t_0) + \frac{k_1 L_1}{a_1} \right) \\
&\quad + \varepsilon \int_{t_0}^{t} \|\lambda(s)\| \, L_2 \mathrm{d}s.
\end{aligned}$$

In the mean time, by the Gronwall-Bellman lemma [58, Lemma A.1],

$$\begin{aligned}
\|\lambda(t)\| &\leq \varepsilon \frac{L_1 k_1}{a_1} \\
&\quad + \int_{t_0}^{t_1} \varepsilon^2 \frac{L_1 k_1}{a_1} \exp\left(\varepsilon L_2 (s - t_0)\right) \mathrm{d}s + \mathcal{O}\left(\varepsilon^2\right) \\
&\leq \varepsilon \frac{L_1 k_1}{a_1} + \varepsilon \frac{L_1 k_1}{a_1 L_2} \exp\left(\varepsilon (t_1 - t_0) L_2\right) + \mathcal{O}\left(\varepsilon^2\right) \\
&\leq \varepsilon \left(k_2 + k_3 \exp\left(\varepsilon (t_1 - t_0) L_2\right)\right), \qquad (4.22)
\end{aligned}$$

from which it can be concluded that the actuator error will be $\mathcal{O}(\varepsilon)$ in the regime where $\varepsilon(t_1 - t_0) = \mathcal{O}(1)$, so that $z(t) = z_c(t, \varepsilon t) + \mathcal{O}(\varepsilon)$, for $t \in [\varepsilon t_0, \varepsilon t_1]$.

Having found an acceptable bound on the actuator model, the plant error can now be bounded. First, define the plant error as $\nu(t) = x(t) - x_c(t)$, and then take its derivative

with respect to time to define the error dynamics,

$$\begin{aligned}\dot{\nu} &= \dot{x} - \dot{x}_c \\ &= \dot{x} - \frac{d}{dt}\alpha + \frac{dT}{dt}\frac{dX}{dT} \\ &= \dot{x} - \dot{\alpha} - \varepsilon\frac{d}{dT}\left(\phi\left(\beta + Z\right)\right) \\ &= \dot{x} - \dot{\alpha} - \varepsilon\frac{d\phi}{dT}\frac{dZ}{dT} \\ &= f(x,z) - f(x_c,z_c) - g(x_c,z_c)\frac{d\phi(z_c)}{dT}. \end{aligned} \tag{4.23}$$

This can be transformed to a perturbed version of the plant's fast time scale dynamics (4.12). First, add and subtract $f\left(\nu, z_c\right)$ and $f(x, z_c)$, which allows for the definition of a perturbation term,

$$\begin{aligned}\dot{\nu} &= f(\nu, z_c) \\ &\quad + f(x,z) - f(x_c,z_c) - f(\nu,z_c) - \varepsilon g(x_c,z_c)\frac{d\phi(z_c)}{dT} \\ &\quad + f(x,z_c) - f(x,z_c) \\ &= f(\nu,z_c) + \Delta f, \end{aligned} \tag{4.24}$$

where the perturbation term can be written as $\Delta f = \Delta_1 + \Delta_2 + \Delta_3$, with $\Delta_1 = \varepsilon g(x_c, z_c)\frac{d\phi(z_c)}{dT}$, $\Delta_2 = f(x,z) - f(x,z_c)$, and $\Delta_3 = f(x,z_c) - f(x_c,z_c) - f(\nu,z_c)$. The norms of Δ_1 and Δ_2 can be bounded in terms of known quantities and Lipschitz constants L_4 and L_5 as,

$$\|\Delta_1\| \leq \varepsilon L_4, \;\; \|\Delta_2\| \leq \|z - z_c\| L_5 = \|\lambda\| L_5.$$

Following from (4.22), these terms in turn can be combined into a single $\mathcal{O}\left(\varepsilon\right)$ term. For Δ_3, subtract the additional term $f(\phi, z_c)$ (this term corresponds to the equilibrium of plant

on the fast time scale so is equal to zero). Then, use (4.3) to put x_c in terms of α and ϕ,

$$\begin{aligned}
\|\Delta_3\| &= \|f(x, z_c) - f(x_c, z_c) - f(\nu, z_c) - f(\phi, z_c)\| \\
&= \|f(\nu + \alpha + \phi, z_c) - f(\alpha + \phi, z_c) \\
&\qquad - f(\nu, z_c) + f(\phi, z_c)\| \\
&= \left\| \int_{\phi}^{\alpha+\phi} \left(\frac{\partial f(s + \nu, z_c)}{\partial \alpha} - \frac{\partial f(s, z_c)}{\partial \alpha} \right) \mathrm{d}s \right\| \\
&\leq \left\| \int_{\phi}^{\alpha+\phi} M \|\nu\| \, \mathrm{d}s \right\| \\
&\leq M \|\nu\| \|\alpha\|,
\end{aligned}$$

where $M \geq 0$ since f is continuously differentiable. Thus, the entire perturbation term can be bounded by $\|\Delta f\| \leq \gamma(t) \|\nu\| + \delta_2$, where for all $t \in [t_0, \varepsilon t_1]$, $\alpha \in D_0$ and $z_c \in D_z$, $\gamma(t) \triangleq M \|\alpha\|$ is non-negative and continuous, and $\delta_2 \triangleq \varepsilon k_4$ is non-negative, continuous, and bounded. Furthermore, $\gamma(t)$ satisfies the following inequality:

$$\int_{t_0}^{t} \gamma(s) \mathrm{d}s \leq (t - t_0)\sigma + \eta,$$

where $0 \leq \sigma < c_1 c_3 / c_2 c_4$ with coefficients from (4.12) – (4.15) and $\eta > 0$. The integral of $\|\alpha\|$ can be bounded by a constant for a finite time interval as, from (4.16), it is not growing in time. Thus, the previous inequality can be satisfied by taking $\sigma = 0$ and $\eta > 0$. The nominal system of $f(\nu, z_c)$ has the Lyapunov function defined in (4.17) – (4.19), so the comparison method conditions detailed in [58, Section 9.3] are satisfied, justifying the following bound on the plant error,

$$\begin{aligned}
\|\nu\| &\leq \sqrt{\frac{c_2}{c_1}} \rho \|\nu(t_0)\| \exp(-a_2(t - t_0)) \\
&\quad + \frac{c_4 \rho}{2 c_1} \int_{t_0}^{t} \exp\left(-a_2(t - s)\right) \delta \, \mathrm{d}s \\
&\leq \left(\sqrt{\frac{c_2}{c_1}} \rho \|\nu(t_0)\| - \frac{\varepsilon k_4 c_4 \rho}{2 c_1 a_2} \right) \exp(-a_2(t - t_0)) \\
&\quad + \varepsilon \frac{k_4 c_4 \rho}{2 c_1 a_2},
\end{aligned} \tag{4.25}$$

where $a_2 = c_3/2c_2$ and $\rho = \exp(c_4\eta/2c_1)$. This bound is valid only if $\|\nu(t)\| < \rho_0$ for all $t \in [t_0, \varepsilon t_1]$, which is ensured if

$$\|\nu(t_0)\| \leq \frac{\rho_0}{\rho}\sqrt{\frac{c_1}{c_2}}$$

and

$$\sup_{t\in[t_0,\varepsilon t_1]} \delta \leq \frac{2c_1 a_2 \rho_0}{c_4 \rho}.$$

Note that by definition, $\|\nu(t_0)\| = 0$, and similarly for δ, so the previous inequalities are satisfied. For all $t \in [t_0, \varepsilon t_1]$, the norm of the plant error satisfies $\|\nu\| \leq \varepsilon(k_5 \exp(-a_2(t - t_0)) + k_6)$, which is $\mathcal{O}(\varepsilon)$, and thus $x(t) = x_c(t) + \mathcal{O}(\varepsilon)$. This completes the proof of Theorem 3. □

While Theorem 3 validates the reduced order model from §4.1.1, it is also natural to ask if the stability region for the model varies significantly from that of the full system. From Step 4 in §4.1.1 $(x_0, z_0) = (x_c(0), z_c(0))$. From this equivalence of initial conditions, if the reduced order model (4.12)-(4.15) is asymptotically stable about the equilibrium, then the region of attraction for the reduced order model is at least as large as the region of attraction for the full system. For systems in standard form, conditions ensuring the asymptotic stability of the reduced order model can be found [58, Theorem 11.3]. Given the above proof, it is expected that an equivalent theorem for (4.1) can be stated; however, such an extension is saved for future work.

4.1.2 Example Application: Leader-follower behavior in satellite constellations

To illustrate the method detailed in §4.1.1, consider a satellite constellation with communication graph given in Figure 4.1. Communication is restricted to a single leader satellite, but all satellites in the constellation need to align their in-plane heading angles. Stable attitude alignment through consensus has been discussed for a variety of network configurations and control schemes [123, 100], but here a simplified version of constellation consensus is considered. Two time scale behavior of the example system manifests from assuming a non-homogeneous constellation where the leader satellite operates on a slower time scale than the

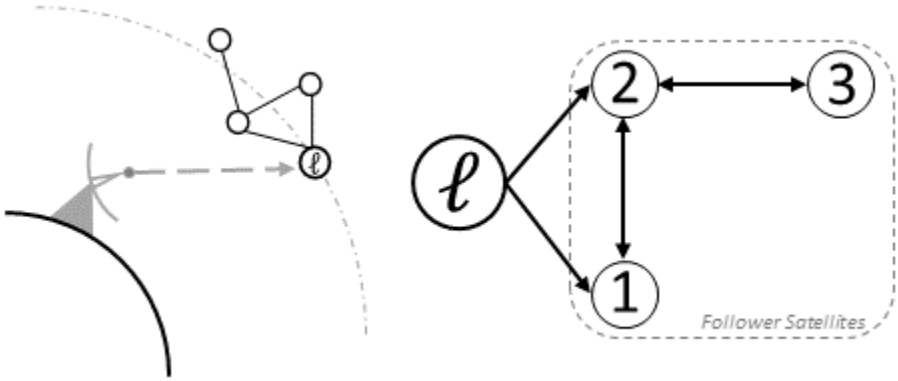


Figure 4.1: Example communications network for a satellite constellation. Control is exerted via control of the leader satellite (l), that operates on a slower time scale than the follower satellites.

followers. In the assumed leader dynamics (4.27), this corresponds to the rate at which the leader satellite can change its attitude and angular velocity being significantly slower than the followers' rate. The fast "plant" dynamics are given by the consensus protocol [75],

$$\frac{d\theta_i}{dt} = \sum_{j \in N(i)} \theta_j - \theta_i = \begin{bmatrix} (\theta_2 + \theta_l) - 2\theta_1 \\ (\theta_l + \theta_2 + \theta_3) - 3\theta_2 \\ (\theta_2 - \theta_3) \end{bmatrix}, \tag{4.26}$$

where θ_i is the heading angle of the ith follower's attitude, θ_l is the heading of the leader, and $N(i)$ is the set of neighbors of the ith follower (the satellites that can communicate with the ith satellite, as defined in Figure 4.1). The slow "actuator" dynamics are closed loop attitude dynamics for a rigid body spacecraft with quaternion-based feedback [25, Chp. 25]. The relative slowness of the leader in comparison to the follower spacecraft is encapsulated

in the small parameter ε as,

$$\begin{aligned}\frac{d}{dt}\begin{bmatrix}\nu_l\\ \eta_l\\ \omega_l\end{bmatrix} &= \varepsilon\begin{bmatrix}\frac{1}{2}\left(\eta_l\omega_l + \nu_l^{\times}\omega_l\right)\\ -\frac{1}{2}\left(\nu_l\cdot\omega_l\right)\\ J^{-1}\left(-\omega_l^{\times}J\omega_l - k_p\nu_l - k_d\omega_l\right)\end{bmatrix}\\ &\triangleq \varepsilon\begin{bmatrix}g_\nu(\nu_l,\omega_l)\\ g_\eta(\nu_l,\eta_l,\omega_l)\\ g_\omega(\nu_l,\omega_l)\end{bmatrix},\end{aligned} \tag{4.27}$$

where ω_l, ν_l, and η_l are the leader's angular velocity, attitude quaternion vector, and quaternion scalar component, respectively; $[v]^{\times}$ denotes the skew symmetric matrix representation of the cross product operation. Assuming that the leader is undergoing in-plane rotation alone to align with the inertial north, the leader's heading angle for use in the consensus dynamics is simply $\theta_l = 2\cos^{-1}(\eta)$. With the plant and actuator dynamics defined in (4.26) and (4.27), the process from §4.1.1 can now be applied. The resulting reduced order model assumes the form,

$$\begin{aligned}\frac{d\alpha_1}{dt} &= \alpha_2 - 2\alpha_1,\ \alpha_1(0) = \theta_1(0) - \theta_l(0)\\ \frac{d\alpha_2}{dt} &= \alpha_1 + \alpha_3 - 3\alpha_2,\ \alpha_2(0) = \theta_2(0) - \theta_l(0)\\ \frac{d\alpha_3}{dt} &= \alpha_2 - \alpha_3,\ \alpha_3(0) = \theta_3(0) - \theta_l(0)\\ \frac{dZ_\nu}{dT} &= g_\nu(\beta_\nu + Z_\nu, \beta_\omega + Z_\omega),\ Z_\nu(0) = 0\\ \frac{dZ_\eta}{dT} &= g_\eta(\beta_\nu + Z_\nu, \beta_\omega + Z_\omega),\ Z_\eta(0) = 0\\ \frac{dZ_\omega}{dT} &= g_\omega(\beta_\nu + Z_\nu, \beta_\omega + Z_\omega), Z_\omega(0) = 0\\ X(T) &= 2\cos^{-1}\left(\eta_{l,0} + Z_\eta\right),\end{aligned} \tag{4.28}$$

where $\alpha_i(t)$ is the fast component of the ith follower, $X(T)$ is the slow manifold (identical for each follower), $\beta_{(*)}(t)$ is the fast leader component, and $Z_{(*)}(T)$ is the slow leader component, with $(*) = [\nu, \eta, \omega]$. The dimension of each term is apparent by the context and the prior definitions of ν, η, and ω.

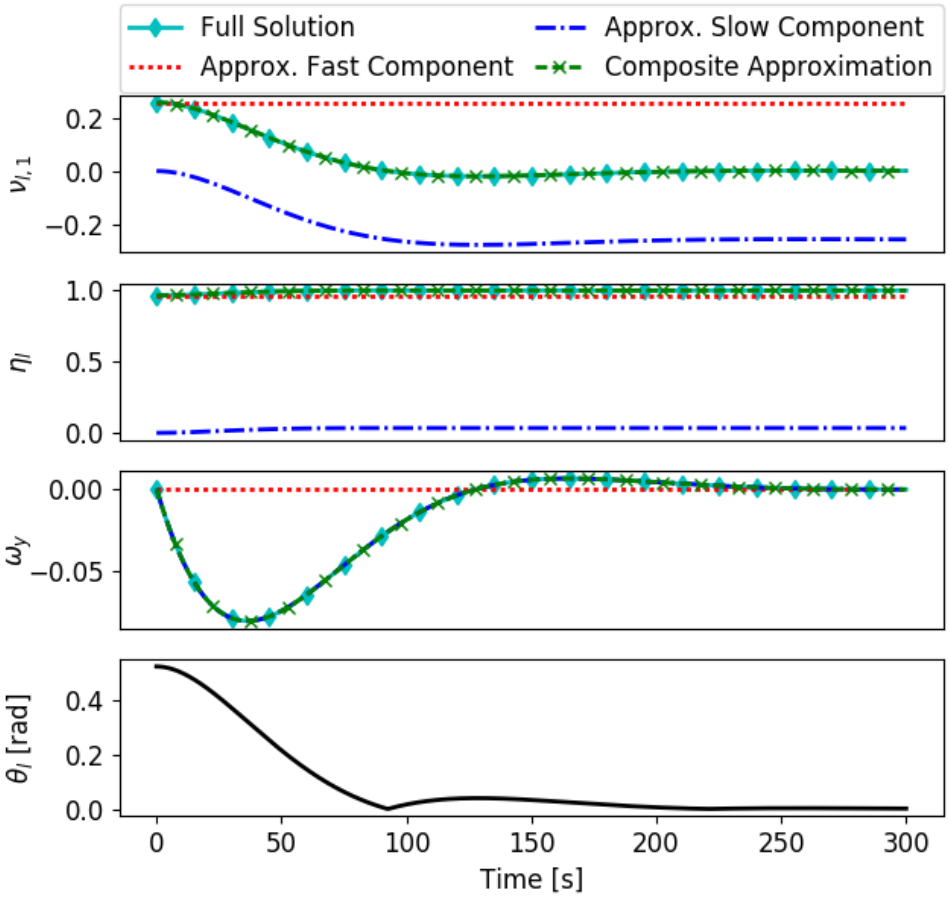


Figure 4.2: Attitude and angular velocity for the leader satellite. Leader heading angle computed using composite approximation for η_l.

For this specific example, the leader satellite was taken to be an axi-symmetric rigid body with $J = I_{3\times3}$. The controller gains were set to $k_d = 0.4$ and $k_p = 0.2$. The initial headings of the followers were randomized, and the control input to the leader was to execute a $\pi/6$ radian rotation to align with the inertial z-axis, with no initial or final angular velocity. The varying quaternion components, angular velocity, and the computed heading for the leader satellite are shown in Figure 4.2, and the heading angles for each follower are shown in Figure 4.3. The results show that the reduced order model reproduces the expected results: that the fast follower satellites quickly reorient to match the slower leader, and then track

its evolution over the slow time scale. The contribution of this method is not simply in the modeling of this type of scenarios, but in the separation of the fast and slow behaviors of each state.

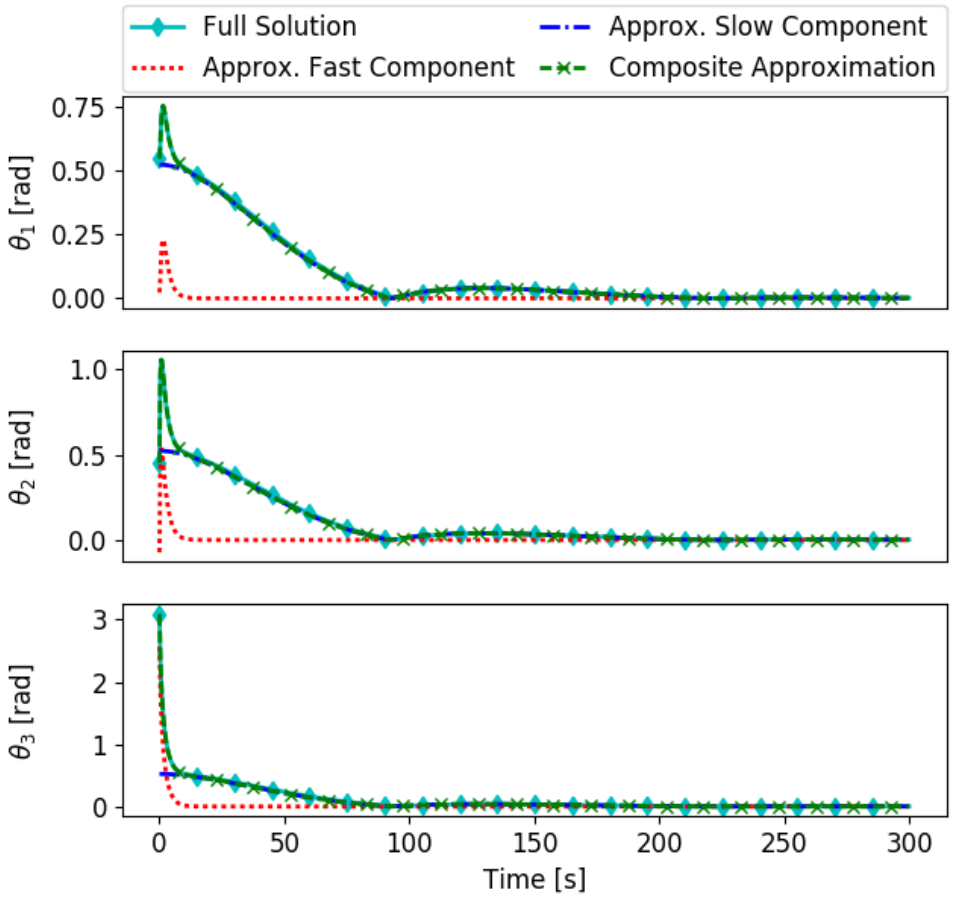


Figure 4.3: Heading angles of the follower satellites.

4.2 Performance of Noisy Scaled Networks[3]

In this section, we will describe a general formulation for consensus over a communication network with positive-definite edge weighting and agent time scaling, with a model for mea-

[3]Text in this section adapted from "Time Scale Design for Network Resilience," presented at the 2019 Conference on Decision and Control, Nice, France [35], "Performance and Design of Consensus on Matrix-Weighted and Time-Scaled Graphs" published in IEEE Transactions on Control of Networked Systems [34].

surement and process noise. The scaled consensus problem is derived from considering a group of n multi-rate integrators [86], with zero-mean Gaussian process noise, $\omega_i(t)$ such that $\mathbb{E}\left[\omega_i(t)\omega_i(t)^T\right] = \Omega_i,\ \forall i \in \mathcal{V}$,

$$\begin{bmatrix} \epsilon_{i,1}\dot{x}_{i,1} \\ \vdots \\ \epsilon_{i,k}\dot{x}_{i,k} \end{bmatrix} = E_i\dot{x}_i(t) = u_i(t) + \omega_i(t), \tag{4.29}$$

where x_i is the vector state of the i-th agent, E_i is $\text{diag}[\epsilon_{i,1}, \ldots, \epsilon_{i,k}]$, and u_i is the control input.

Suppose that communication between agents i and j is corrupted by zero-mean Gaussian noise v_{ij}, and let v_i denote the sum of all noise inputted into agent i from the connections to its neighbors in $N(i)$. Without loss of generality, we can assume that the covariance of v_i is given by $\mathbb{E}[v_i(t)v_i(t)^T] = \Gamma_i$. A weighted, decentralized feedback controller, with noise, that seeks to bring agents into consensus is given by,

$$\begin{aligned} u_i(t) &= \sum_{j \in N(i)} \left[\mathbf{W}_{ij}(x_j(t) - x_i(t)) + v_{ij}(t)\right] \\ \mathbf{u}(t) &= -\mathbf{D}(\mathcal{G})\mathbf{W}\mathbf{D}(\mathcal{G})^T\mathbf{x}(t) + \mathbf{D}(\mathcal{G})\mathbf{v}(t), \end{aligned} \tag{4.30}$$

where $\mathbf{W}$ is the block-diagonal matrix of edge weights with properties detailed in Section 2.5. The vector $\mathbf{u}$ is the stacked vector of control vectors u_i, and $\mathbf{v}$ is the stacked vector of all measurement noises. Applying (4.30) to the system (4.29) with appropriate dimensions gives the general, time scaled and matrix weighted consensus problem with process and measurement noise,

$$\begin{aligned} \dot{\mathbf{x}}(t) &= -\mathbf{E}^{-1}\mathcal{L}_w(\mathcal{G})\mathbf{x}(t) + \begin{bmatrix} \mathbf{E}^{-1} & -\mathbf{E}^{-1}\mathbf{D}(\mathcal{G}) \end{bmatrix} \begin{bmatrix} \omega(t) \\ \mathbf{v}(t) \end{bmatrix} \\ \mathbf{z}(t) &= \mathbf{D}(\mathcal{G})^T\mathbf{x}(t) \end{aligned} \tag{4.31}$$

where $\mathcal{L}_w(\mathcal{G})$ is the weighted Laplacian matrix, and $\mathbf{E} = \text{Blkdiag}(E_1, \ldots, E_n)$ is the full time scale matrix of $\mathcal{G}$. Here, we have introduced an output $\mathbf{z}(t)$ which is used to monitor the network performance which captures the differences between the node states as they evolve.

As noted by [127, 36] for scalar-valued node states over a connected graph, the zero eigenvalue (corresponding to the consensus subspace) of the Laplacian matrix precludes reasoning about the $\mathcal{H}_2$ performance of (4.31). Under matrix weighting, the zero eigenvalue will have algebraic multiplicity k (corresponding to the consensus subspace of each layer of substates) [119], so as in [127] we will appeal to a similarity transformation that separates out the zero eigenvalues. We define this transformation in the following theorem.

Theorem 4. *The time-scaled dynamics for weighted consensus can be transformed to an equivalent form that separates out the zero eigenvalue of the consensus subspace (along with the zero eigenvalues corresponding to cycles in the edge-consensus dynamics).*

Formally, the scaled and edge-weighted graph Laplacian for a connected graph with time scale matrix $\mathbf{E}$ *and weight matrix* $\mathbf{W}$*, given by* $\mathcal{L}_{w,s} = \mathbf{E}^{-1}\mathbf{D}(\mathcal{G})\mathbf{W}\mathbf{D}(\mathcal{G})^T$*, is similar to*

$$\begin{bmatrix} \mathcal{L}_{e,s}\mathbf{R}\mathbf{W}\mathbf{R}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix},$$

where $\mathcal{L}_{e,s} = \mathbf{D}(\mathcal{G}_\tau)^T\mathbf{E}^{-1}\mathbf{D}(\mathcal{G}_\tau)$ *is the edge Laplacian for a spanning tree* $\mathcal{G}_\tau$ *which is symmetrically "weighted" by the time scaling parameters, and* $\mathbf{R}(\mathcal{G}) = \begin{bmatrix} I & \mathbf{T}_\tau^c \end{bmatrix} = R \otimes I_k$*, where* R *is the basis of the cut space of* $\mathcal{G}$ *as defined as in [127], with* $\mathbf{T}_\tau = (\mathbf{D}(\mathcal{G}_\tau)^T\mathbf{D}(\mathcal{G}_\tau))^{-1}\mathbf{D}(\mathcal{G}_\tau)^T\mathbf{D}(\mathcal{G}_c)$*. Here, the* τ *and* c *subscripts on* $\mathcal{G}$ *denote the incidence matrices for a spanning tree and the complementary edges in* $\mathcal{G}$*, respectively.*

Before proving the main theorem, we prove two intermediate lemmas that will facilitate the main proof.

Lemma 1. *The following hold:* $\mathcal{T}_\tau = \mathbf{T}_\tau \otimes I_k$, $\mathbf{D} = \mathbf{D}_\tau\mathbf{R}$.

Proof. We can compute:

$$\begin{aligned} \mathbf{T}_\tau^c &= \left(\mathbf{D}_\tau^T\mathbf{D}_\tau\right)^{-1}\mathbf{D}_\tau^T\mathbf{D}_c \\ &= \left(\left(D_\tau^T D_\tau\right)^{-1} D_\tau D_c\right) \otimes I_k = T_\tau^C \otimes I_k, \end{aligned}$$

and similarly, $\mathbf{D} = D \otimes I_k = (D_\tau \otimes I_k)(R \otimes I_k) = \mathbf{D}_\tau\mathbf{R}$. □

Lemma 2. *The similarity transforms (in the proof of Theorem 4) are well-defined, in that* $S_v^{-1}S_v = I$.

Proof. Denote the product $S_v^{-1}S_v$ in block form:

$$S_v^{-1}S_v = \begin{bmatrix} \mathcal{A} & \mathcal{B} \\ \mathcal{C} & \mathcal{D} \end{bmatrix}.$$

First, note that

$$F\mathbf{E}^{-1} = \begin{bmatrix} E_1 & \cdots & E_n \end{bmatrix} \mathbf{E}^{-1} = \begin{bmatrix} I & \cdots & I \end{bmatrix} = \mathbf{1}_n^T \otimes I_k.$$

Then, we can compute each term:

$$\begin{aligned}
\mathcal{A} &= \mathbf{D}_\tau^T \mathbf{E}^{-1} \mathbf{D}_\tau \left(D_\tau^T \mathbf{E}^{-1} \mathbf{D}_\tau \right)^{-1} = I \\
\mathcal{B} &= \mathbf{D}_\tau^T \mathbf{1} = \left(D_\tau^T \mathbf{1}_n \right) \otimes I_k = 0 \otimes I_k = \mathbf{0}. \\
\mathcal{C} &= \Xi^{-1} F \mathbf{E}^{-1} \mathbf{D}_\tau \left(\mathbf{D}_\tau^T \mathbf{E}^{-1} \mathbf{D}_\tau \right)^{-1} \\
&= \Xi^{-1} \left(\mathbf{1}_n^T \mathbf{D}_\tau \otimes I_k \right) \left(\mathbf{D}_\tau^T \mathbf{E}^{-1} \mathbf{D}_\tau \right)^{-1} = \mathbf{0}.
\end{aligned}$$

Lastly, we have

$$\begin{aligned}
\mathcal{D} &= \Xi^{-1} F \mathbf{1} \\
&= \begin{bmatrix} \epsilon_{s,1}^{-1} & & \\ & \ddots & \\ & & \epsilon_{s,k}^{-1} \end{bmatrix} \begin{bmatrix} E_1 & \cdots & E_n \end{bmatrix} (\mathbf{1}_n \otimes I_k) \\
&= \begin{bmatrix} \epsilon_{s,1}^{-1} & & \\ & \ddots & \\ & & \epsilon_{s,k}^{-1} \end{bmatrix} \begin{bmatrix} \sum_{j=1}^n \epsilon_{j,1} & & \\ & \ddots & \\ & & \sum_{j=1}^n \epsilon_{j,k} \end{bmatrix} = I_k.
\end{aligned}$$

□

With those results in hand, we can now move on to the main proof.

Proof. Following [127, 35], we define a similarity transformation,

$$
\begin{aligned}
S_v(\mathcal{G}) &= \begin{bmatrix} \mathbf{E}^{-1}\mathbf{D}(\mathcal{G}_\tau)\left(\mathbf{D}(\mathcal{G}_\tau)^T\mathbf{E}^{-1}\mathbf{D}(\mathcal{G}_\tau)\right)^{-1} & \mathbf{1} \end{bmatrix} \\
S_v(\mathcal{G})^{-1} &= \begin{bmatrix} \mathbf{D}(\mathcal{G}_\tau)^T \\ \Xi^{-1}F \end{bmatrix}, \\
\mathbf{1} &= \mathbf{1}_n \otimes I_k, \;\; F = \begin{bmatrix} E_1 & \cdots & E_n \end{bmatrix} \\
\Xi &= \operatorname{diag}\left(\{\epsilon_{s,i}\}_{i=1}^k\right), \;\; \epsilon_{s,i} = \sum_{j=1}^n \epsilon_{j,i}.
\end{aligned}
$$

Note that $\epsilon_{s,i}$ is the sum of all the time scale parameters of the i-th substate over all nodes. By Lemmas 1 and 2 this transformation is well-defined.

Next, denoting for brevity $\mathbf{D}_\tau := \mathbf{D}(\mathcal{G}_\tau)$, $\mathbf{D} := \mathbf{D}(\mathcal{G})$, etc, and noting that $\mathbf{D}^T\mathbf{1} = (D^T\mathbf{1}) \otimes I_k = \mathbf{0}$, we conclude,

$$
\begin{aligned}
&S_v^{-1}\mathcal{L}_{w,s}(\mathcal{G})S_v = \\
&\begin{bmatrix} \mathbf{D}_\tau^T\mathbf{E}^{-1}\mathbf{D}\mathbf{W}\mathbf{D}^T\mathbf{E}^{-1}\mathbf{D}_\tau\left(\mathbf{D}_\tau\mathbf{E}^{-1}\mathbf{D}_\tau\right)^{-1} & A \\ \Xi^{-1}F\mathbf{E}^{-1}\mathbf{D}\mathbf{W}\mathbf{D}^T\mathbf{E}^{-1}\mathbf{D}_\tau\left(\mathbf{D}_\tau\mathbf{E}^{-1}\mathbf{D}_\tau\right)^{-1} & B \end{bmatrix} \\
&(\text{where } A = \mathbf{D}_\tau^T\mathbf{E}^{-1}\mathbf{D}\mathbf{W}\mathbf{D}^T\mathbf{1}, \;\; B = \Xi^{-1}F\mathbf{E}^{-1}\mathbf{D}\mathbf{D}^T\mathbf{1}) \\
&= \begin{bmatrix} \mathbf{D}_\tau^T E^{-1}\mathbf{D}_\tau\mathbf{R}\mathbf{W}\mathbf{R}^T\mathbf{D}_\tau^T\mathbf{E}^{-1}\mathbf{D}_\tau\left(\mathbf{D}_\tau\mathbf{E}^{-1}\mathbf{D}_\tau\right)^{-1} & \mathbf{0} \\ \Xi^{-1}\mathbf{1}^T\mathbf{D}_\tau\mathbf{R}\mathbf{W}\mathbf{R}^T & \mathbf{0} \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{L}_{e,s}\mathbf{R}\mathbf{W}\mathbf{R}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}.
\end{aligned}
$$

□

By noting that $S_v\mathbf{x}_e(t) = \mathbf{x}(t)$, the scaled, matrix weighted consensus model with noise (4.29)

is equivalent to,

$$\begin{aligned}\dot{\mathbf{x}}_e(t) &= \begin{bmatrix} -\mathcal{L}_{e,s}(\mathcal{G}_\tau)\mathbf{R}(\mathcal{G})\mathbf{W}\mathbf{R}(\mathcal{G})^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{x}_e(t) \\ &\quad + \begin{bmatrix} \mathbf{D}_\tau^T\mathbf{E}^{-1} & -\mathcal{L}_{e,s}(\mathcal{G}_\tau)\mathbf{R}(\mathcal{G}) \\ \Xi^{-1}\mathbf{1}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \omega(t) \\ \mathbf{v}(t) \end{bmatrix} \\ \mathbf{z}(t) &= \begin{bmatrix} \mathbf{R}(\mathcal{G}) & \mathbf{0} \end{bmatrix} \mathbf{x}_e(t).\end{aligned} \tag{4.32}$$

We can see that the form of (4.32) naturally suggests a partitioning of the edge state variable into a set of states in the spanning tree and those in the consensus space (span($\mathbf{1} \otimes I_k$)), $\mathbf{x}_e(t) = \begin{bmatrix} \mathbf{x}_\tau(t) & \mathbf{x_1}(t) \end{bmatrix}$. The resulting dynamics for the spanning tree states is taken from (4.32) as,

$$\Sigma^\tau[\mathcal{G}] := \begin{cases} \dot{\mathbf{x}}_\tau(t) = -\mathcal{L}_{e,s}(\mathcal{G}_\tau)\mathbf{R}(\mathcal{G})\mathbf{W}\mathbf{R}(\mathcal{G})^T\mathbf{x}_\tau(t) \\ \qquad\qquad + \mathbf{D}_\tau^T\mathbf{E}^{-1}\Omega\hat{\omega} - \mathcal{L}_{e,s}(\mathcal{G}_\tau)\mathbf{R}(\mathcal{G})\Gamma\hat{\mathbf{v}} \\ \mathbf{z}(t) = \mathbf{R}(\mathcal{G})^T\mathbf{x}_\tau(t), \end{cases} \tag{4.33}$$

where $\hat{\mathbf{v}}$ and $\hat{\omega}$ are normalized noise signals, $\Omega = \mathbb{E}\left[\omega(t)\omega(t)^T\right]$, and $\Gamma = \mathbb{E}\left[\mathbf{v}(t)\mathbf{v}(t)^T\right]$. An important note is that the chosen output for (4.31) results in the output of (4.33) containing information of the cycle states due to the fact that the cycle states are linear combinations of the tree states.

We can also consider the same edge state model with output given solely by the spanning tree states,

$$\hat{\Sigma}^\tau[\mathcal{G}] := \begin{cases} \dot{x}_\tau(t) = -\mathcal{L}_{e,s}(\mathcal{G}_\tau)\mathbf{R}(\mathcal{G})\mathbf{W} \qquad +\mathbf{D}_\tau^T\mathbf{E}^{-1}\Omega\hat{\omega} - \mathcal{L}_{e,s}(\mathcal{G}_\tau)\mathbf{R}(\mathcal{G})\Gamma\hat{\mathbf{v}} \\ \mathbf{z}(t) = \mathbf{x}_\tau(t). \end{cases} \tag{4.34}$$

The $\mathcal{H}_2$ performance of (4.33) and (4.34) are given by $\mathbf{tr}(R^T X^\star R)$ and $\mathbf{tr}(X^\star)$, respectively, where $X^\star$ is the positive-definite solution to the Lyapunov equation,

$$-\mathcal{L}_{e,s}^\tau\mathbf{R}\mathbf{W}\mathbf{R}^T X - X\mathbf{R}\mathbf{W}\mathbf{R}^T\mathcal{L}_{e,s}^\tau + \mathbf{D}_\tau^T\mathbf{E}^{-1}\Omega\Omega^T\mathbf{E}^{-1}\mathbf{D}_\tau + \mathcal{L}_{e,s}^\tau\mathbf{R}\Gamma\Gamma^T\mathbf{R}^T\mathcal{L}_{e,s}^\tau = 0. \tag{4.35}$$

In general, the addition of the matrix weighting and scaling precludes a closed form solution to (4.35) (which is desirable to find X's dependence on $\mathbf{E}, \mathbf{W}$), and numeric results yield a

nontrivial mixing of weights and scaling parameters in the entries of X. However, in the following section we will outline a case when analytic solutions to (4.35) exist, providing insights for design of edge weights and scaling parameters for optimal performance.

4.2.1 $\mathcal{H}_2$ Performance

In this section, we discuss the $\mathcal{H}_2$ performance for the models of edge consensus in the cases of nodes with time scales, and matrix weighted edges. Specifically, we identify an interesting case where explicit solutions to (4.35) can be found by an appropriate choice of noise covariances, and then discuss how the general solution can be approximated by this choice.

By inspection of (4.35), we can note that by selecting the covariance matrices $\Omega = \sigma_\omega \mathbf{E}^{1/2}$ and $\Gamma = \sigma_v \mathbf{W}^{1/2}$, we can find an analytic solution for (4.35) in line with those in [127, 35]. With this choice, (4.35) has the solution,

$$X^\star = \frac{1}{2}\left(\sigma_w^2(\mathbf{RWR}^T)^{-1} + \sigma_v^2 \mathcal{L}_{e,s}^\tau\right). \tag{4.36}$$

This solution is of particular interest because the edge and node weightings are separated in their effect on the $\mathcal{H}_2$ performance, save for the placement of the σ_ω and σ_v parameters (that is, the effective covariance parameter of the process noise is a "node" parameter, but multiplies the term containing the edge weighting in (4.36), and vice versa). Note that while this choice of Ω and Γ allows for the analytic solution (4.36), this solution is merely a proxy for the true performance of the system; for it to be useful for analysis, the error induced by the choice of covariances needs to be assessed.

Consider the scenario in which we have some given covariance matrices, Ω_T and Γ_T, which when used to solve (4.35) yield the "true" performance of the system. These matrices may or may not be known to us; in either case a relevant endeavor is the quantification of the error incurred by estimating the true performance by (4.36). The method we will use to quantify this error depends on the positive-definite ordering of performances given by covariance matrices. The following lemma provides this ordering.

Lemma 3 (Ordering of Performance)**.** *For covariance matrices satisfying,* $\underline{\Omega} \preceq \Omega \preceq \bar{\Omega}$ *and* $\underline{\Gamma} \preceq \Gamma \preceq \bar{\Gamma}$*, then denoting the solution to* (4.35) *using* $(\underline{\Omega}, \underline{\Gamma})$; (Ω, Γ); $(\bar{\Omega}, \bar{\Gamma})$ *as* $\underline{X}$; X; $\bar{X}$*, we have,* $\underline{X} \preceq X \preceq \bar{X}$*. From this we can conclude,*

$$\mathbf{tr}(R^T \underline{X} R) \leq \mathbf{tr}(R^T X R) \leq \mathbf{tr}(R^T \bar{X} R).$$

Proof. For simplicity, we adopt the following notation the state and input matrix of (4.33), $A := -\mathcal{L}_{e,s}\mathbf{RWR}^T$, and $B := B_\tau \Omega \Omega^T B_\tau^T + B_c \Gamma \Gamma^T B_c^T$. Due to the stability of $\mathcal{L}(\mathcal{G})$, we know the solution to (4.35) is positive definite, and can be written as in integral form [110],

$$X = \int_0^\infty e^{At} B_\tau \Omega \Omega^T B_\tau^T e^{A^T t} \, \mathrm{d}t + \int_0^\infty e^{At} B_c \Gamma \Gamma^T B_c^T e^{A^T t} \, \mathrm{d}t$$

Consider either term in the above solution, and generalize as,

$$X_p = \int_0^\infty e^{At} B_p Z Z^T B_p^T e^{A^T t} \, \mathrm{d}t$$

where Z and B_p are placeholders for a covariance matrix and input matrix, respectively. From this, consider the quadratic form with any u,

$$u^T X_p u = \int_0^\infty \|Z^T B_p^T e^{A^T t} u\|_2^2 \, \mathrm{d}t$$

This form is usually employed to show the positive definiteness of X_p based on the stability of A, but here it can be used to order X_p's based on the ordering of Z:

$$\begin{aligned} u^T (X_p - \bar{X}_p) u &= \int_0^\infty \|Z^T B_p^T e^{A^T t} u\|_2^2 - \|\bar{Z}^T B_p^T e^{A^T t} u\|_2^2 \, \mathrm{d}t \\ &\leq \int_0^\infty \left(\|Z^T\|_2^2 - \|\bar{Z}^T\|_2^2 \right) \|B_p^T e^{A^T t} u\|_2^2 \, \mathrm{d}t \end{aligned}$$

From $Z \preceq \bar{Z}$, it follows that $\|Z\|_2^2 \leq \|\bar{Z}\|_2^2$. Thus, this integral is always negative, which then implies $X_p - \bar{X}_p \preceq 0 \Rightarrow X_P \succeq \bar{X}_p$. By identical argument,

$$u^T (\underline{X}_p - X_p) u \leq \int_0^\infty \left(\|\underline{Z}^T\|_2^2 - \|Z^T\|_2^2 \right) \|B_p^T e^{A^T t} u\|_2^2 \, \mathrm{d}t$$

implies that $\underline{X}_p \preceq X$. This shows the ordering for the solutions of (4.35) for ordered covariances.

Now, consider the ordering of the performance, which is given by $\mathbf{tr}(RXR^T)$. From the ordering and positive definiteness of $\underline{X}, X, \bar{X}$, we have, $u^T\underline{X}u \leq u^T X u \leq u^T \bar{X} u$, for all u. Letting $u = Rx$ gives,

$$x^T R^T \underline{X} R x \leq x^T R^T X R x \leq x^T R^T \bar{X} R x.$$

Now, note that the trace of a matrix M can be written as $\sum_i e_i^T M e_i$. Taking $M = R^T X R$ and $x = e_i$ in the above equation gives the desired ordering on the performances. □

We can employ Lemma 3 to place a bound on the potential error of making the assumption that gives (4.36). Observe that it is possible to choose multiplicative factors such that,

$$\begin{aligned} \underline{\alpha}\mathbf{E}^{1/2} &\preceq \Omega_T \preceq \bar{\alpha}\mathbf{E}^{1/2} \\ \underline{\beta}\mathbf{W}^{1/2} &\preceq \Gamma_T \preceq \bar{\beta}\mathbf{W}^{1/2}. \end{aligned}$$

From Lemma 3, we know that the true performance will lie within the performances calculated using $(\underline{\alpha}\mathbf{E}^{1/2}, \underline{\beta}\mathbf{W}^{1/2})$ and $(\bar{\alpha}\mathbf{E}^{1/2}, \bar{\beta}\mathbf{W}^{1/2})$. Thus, taking the difference between the maximum performance and the minimal performance gives the worst case error of (4.36), which is given by,

$$\begin{aligned} \mathbf{tr}(R^T \bar{X}^* R) - \mathbf{tr}(R^T \underline{X}^* R) = &\frac{1}{2}(\bar{\alpha} - \underline{\alpha})\sigma_w^2 \mathbf{tr}((\mathbf{R}\mathbf{W}\mathbf{R}^T)^{-1}) \\ &+ \frac{1}{2}(\bar{\beta} - \underline{\beta})\sigma_v^2 \mathbf{tr}(\mathcal{L}_{e,s}^{\tau}). \end{aligned} \tag{4.37}$$

From this result, we can see that the relative error will be determined by the relative sizes of the multiplicative factors, which raises the question of how the factors can be found or chosen. From the definition of the necessary ordering, it is of course sufficient that,

$$\begin{aligned} \bar{\alpha} &= \frac{\lambda_{\max}(\Omega_T)}{\epsilon_{\max}^{1/2}};\ \underline{\alpha} = \frac{\lambda_{\min}(\Omega_T)}{\epsilon_{\min}^{1/2}} \\ \bar{\beta} &= \frac{\lambda_{\max}(\Gamma_T)}{\lambda_{\max}(\mathbf{W})^{1/2}};\ \underline{\beta} = \frac{\lambda_{\min}(\Gamma_T)}{\lambda_{\min}(\mathbf{W})^{1/2}}. \end{aligned} \tag{4.38}$$

The sufficiency, as opposed to necessity, of the parameters given in (4.38) results in conservative performance bounds. The bounds can be tightened by solving for the minimal/maximal

parameters via a simple (convex) optimization problem of the form (for $\bar{\alpha}$),

$$\begin{aligned} \min_{\alpha} \quad & \alpha \\ \text{s.t.} \quad & \Omega_T - \alpha E^{1/2} \preceq 0. \end{aligned} \tag{4.39}$$

The above problem can be modified to give the minimal/maximal values of $\bar{\beta}$ and $\underline{\alpha}/\underline{\beta}$. While (4.39) gives an obvious advantage over (4.38), it does require complete information about the covariances, whereas (4.38) requires only the spectral bounds of the true covariances. In either case, however, it can be seen that the worst case error is helped when the true covariances and the assumed covariances have similar spectral bounds, that is, if the minimum and maximum eigenvalues of Ω_T/Γ_T are approximately equal to those of $\sigma_w E^{1/2}/\sigma_v W^{1/2}$, the necessary multiplicative factors will be approximately unity and the possible discrepancy small (within a factor of $\simeq$5-10). We can see this numerically for random graphs on $n = 20$ nodes with $k = 1, 2$ in Figures 4.4 and 4.5, where the true covariances, edge weights, and node time scales were randomly generated then scaled to align maximum/minimum eigenvalues. Numerical results over a range of n and k suggest that as the number of substates increases, the bounds (calculated via (4.39)) become more conservative, which can be observed in the increase in the bound region of Figure 4.5 compared to the region in Figure 4.4.

While the results of Lemma 3 (along with (4.38) or (4.39)) give the ability to assess whether or not (4.36) is an acceptable proxy for the true performance, there are no explicit guarantees that the bounds are tight enough for all applications. In cases where there is large discrepancy between the spectral bounds of the true covariances and the time scale or edge weight matrices, for example, (4.36) may not be useful for performance estimation or optimization. For the cases where it is an appropriate proxy, however, we proceed with time scale and edge weight design to promote minimal $\mathcal{H}_2$ performance, starting with a remark pertaining to the special case of tree graphs.

Remark 7. *When the underlying graph topology is a tree,* $\mathbf{R} = I$*, and* (4.36) *simplifies to,*

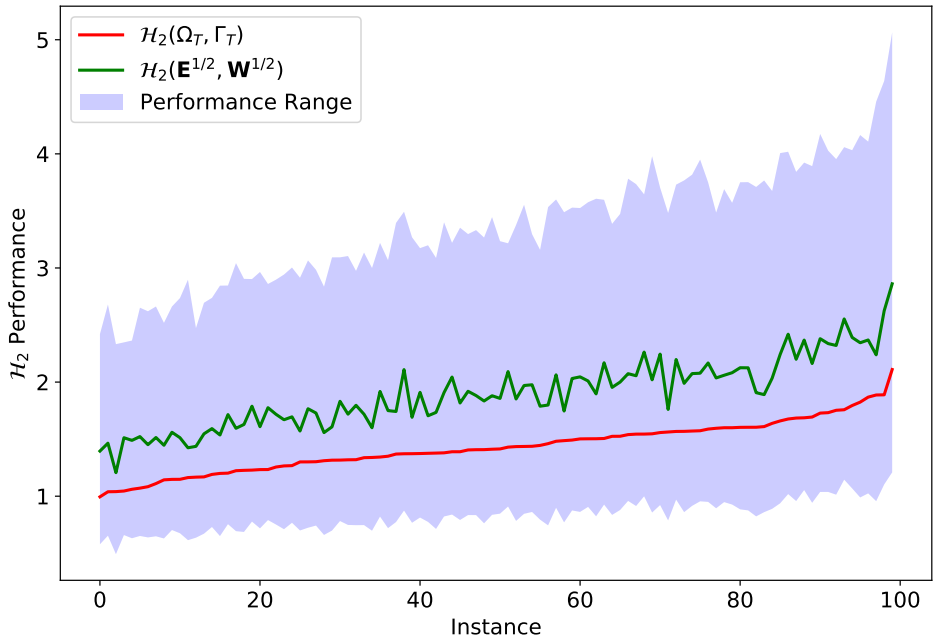


Figure 4.4: Numerically calculated $\mathcal{H}_2$ performance for random graphs ($n = 20, k = 1$) with random covariances ($\mathcal{H}_2(\Omega_T, \Gamma_T)$) and the solution to (4.36) ($\mathcal{H}_2(E^{1/2}, W^{1/2})$). The blue shaded region represents the possible range in performance, calculated using the parameters found via optimization problems given by (4.39).

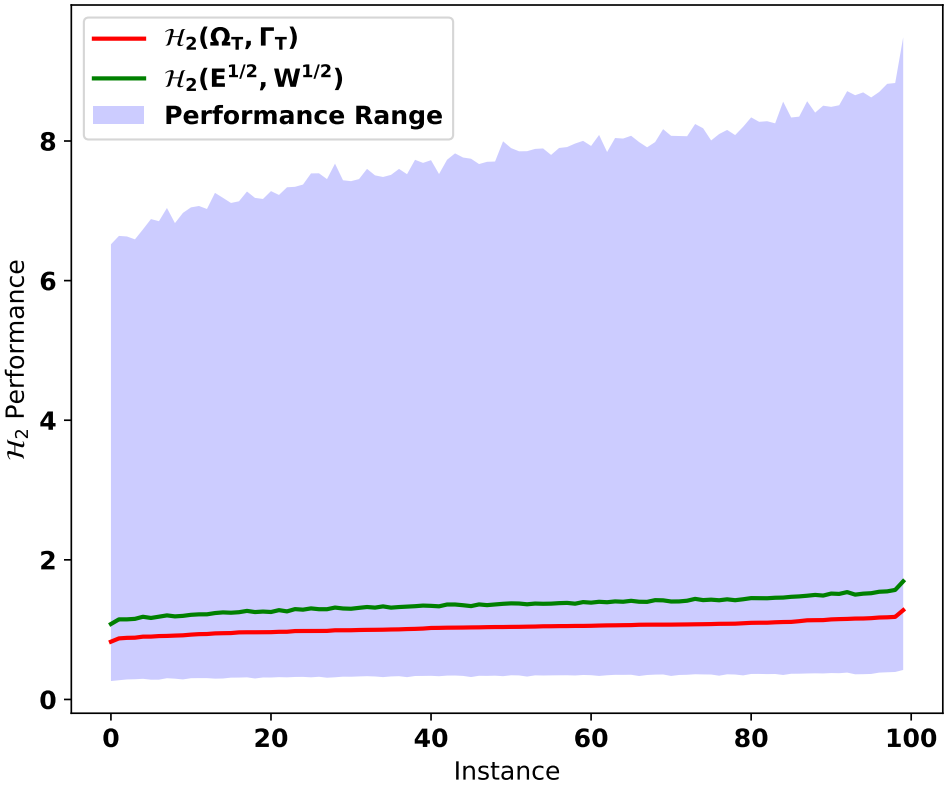


Figure 4.5: Numerically calculated $\mathcal{H}_2$ performance for random graphs ($n = 20, k = 2$) with random covariances ($\mathcal{H}_2(\Omega_T, \Gamma_T)$) and the solution to (4.36) ($\mathcal{H}_2(\mathbf{E}^{1/2}, \mathbf{W}^{1/2})$). The blue shaded region represents the possible range in performance, calculated using the parameters found via optimization problems given by (4.39).

$$X^\star = \frac{1}{2}\left(\sigma_\omega^2 \mathbf{W}^{-1} + \sigma_v^2 \mathcal{L}_{e,s}^\tau\right).$$

Furthermore, in this case $\mathcal{H}_2(\Sigma^\tau) = \mathbf{tr}(X^\star)$*. A closed form solution for the performance in this case is given in the following lemma.*

Lemma 4. *For a tree graph, the* $\mathcal{H}_2$ *norm of the* Σ^τ (4.33) *system is given by,*

$$\begin{aligned}\mathcal{H}_2(\Sigma^\tau) &= \frac{1}{2}\mathbf{tr}\left(\sigma_\omega^2 \mathbf{W}^{-1} + \sigma_v^2 \mathcal{L}_{e,s}^\tau\right) \\ &= \frac{1}{2}\left(\sigma_\omega^2 \sum_{j=1}^{n-1} \mathbf{tr}\left(W_j^{-1}\right) + \sigma_v^2 \sum_{l=1}^{k}\sum_{i=1}^{n} \frac{deg(\nu_i)}{\epsilon_{i,l}}\right),\end{aligned} \tag{4.40}$$

where $deg(\nu_i)$ *is the unweighted degree of agent* ν_i*, and* j *is the index over the edges.*

Proof. First consider the weight term. $\mathbf{W}$ is block-diagonal, so,

$$\mathbf{W}^{-1} = \text{Blkdiag}(W_1^{-1}, \ldots, W_{n-1}^{-1}).$$

Thus, its trace is the sum of the traces of the edge weight matrix inverses. Now, consider a single layer of substates, denoted by l. For the second term, consider one of the diagonal elements of $\mathcal{L}_{e,s}$,

$$[L_{e,s}^\tau]_{(ql)(ql)} = a_q^T E_l^{-1} a_q = \epsilon_{i,l}^{-1} + \epsilon_{j,l}^{-1},$$

where a_q is the edge vector corresponding to the edge between nodes i and j, that is, $q = \kappa(ij)$. Now consider a single node, ν_i. In the sum over all edges of the graph, $\epsilon_{i,l}^{-1}$ will appear once for every edge that connects ν_i to its neighbors, which is the unweighted degree of ν_i. Considering all other nodes gives the result for the second term. Finally, the preceding argument holds for all the sub-state layers, which gives the sum over all sub-states. □

From this result, we can see that there exists a trade off between the time scale parameters and the topology (in this case, the degree distribution) which determines the overall performance of the network. Also, we can contrast the influence of time scale parameters and edge weights in this case. For a given distribution of scaling parameters and edge weights,

changing the assignment of edge weights does not affect the $\mathcal{H}_2$ performance contribution from a given sub-state layer. However, the assignment of scaling parameters can have a significant effect on the performance of the network, which is in line with the similar results in the context of single-input influenced consensus [36].

We can also see that the performance contribution from the time scale parameters in the matrix weighted case is identical to considering a network with k disconnected layers, where each layer has its own time scale distribution. Thus, the evaluation of the time scale assignment in the matrix weighted case is effectively the same as considering assignment in the scalar case. This is in line with results in the context of single-input influenced consensus [36], and shown in the following example.

Example: Consider the tree graph on six nodes in Figure 4.6. Assume that we have some distribution of edge weights and node scaling parameters such that $\sum_i w_i^{-1} = 2\alpha$, $\sum_i \epsilon_i = 1$, and further, that $\epsilon_i \in (0.1, 0.2, 0.4)$. Also, let $\sigma_v = \sigma_\omega = 1$. Now, for any assignment of

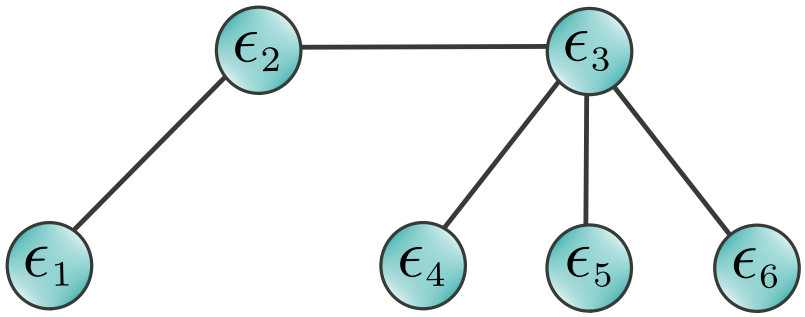


Figure 4.6: Tree graph $\mathcal{T}$ for six agents. Each agent i has an associated scaling parameter, ϵ_i, and each jth edge is labeled w_j. Agent 3 has the highest degree.

weights, the first term in (4.40) will be 2α, but the second term depends on the assignment of the scaling parameters. Consider the two following assignments: (1) set $\epsilon_{1,4,5,6} = 0.1$, $\epsilon_2 = 0.2$, $\epsilon_3 = 0.4$. In this case, the second term is equal to $\sum_{i=1}^{6} \deg(i)/\epsilon_i = 60.0$ resulting in a combined $\mathcal{H}_2 = 30 + \alpha$. Now, consider a different distribution of scaling parameters. (2) set $\epsilon_{1,3,4,5} = 0.1$, $\epsilon_2 = 0.2$, $\epsilon_6 = 0.4$. We can see that with agent 3 on a faster time scale, the

second term suffers, $\sum_{i=1}^{6} \deg(i)/\epsilon_i = 82.5$, resulting in a comparatively higher performance value of $\mathcal{H}_2 = 41.25 + \alpha$. Thus, we can see that high-degree nodes with slower time scales results in lower $\mathcal{H}_2$ performance.

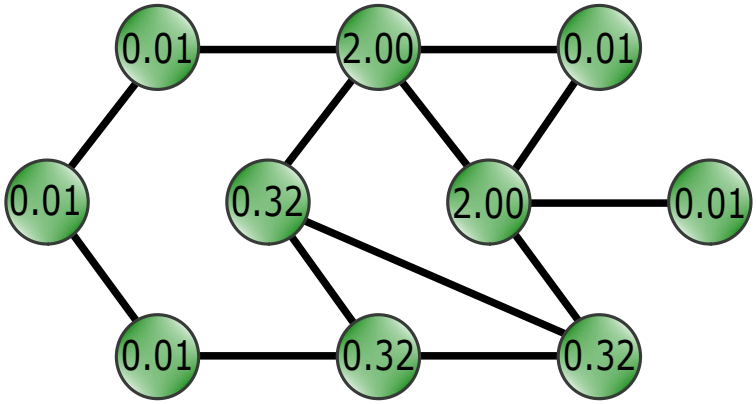
4.2.2 Time scale Design For $\mathcal{H}_2$ Resilience

The results of the previous section suggest that a heuristic for minimizing the $\mathcal{H}_2$ performance is to assign slower timescales to high-degree agents. To investigate whether this holds in the presence of cycles, note that the minimization of $\mathcal{H}_2(\Sigma^\tau)$ can be formulated as a convex problem,

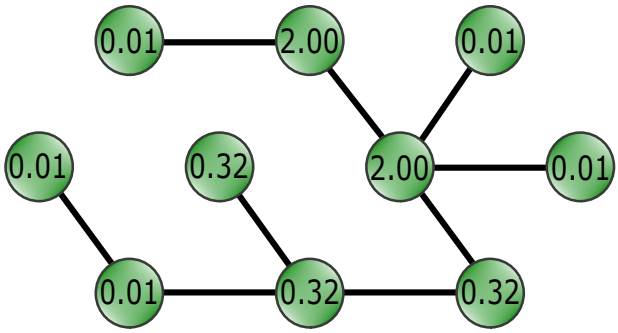
$$\begin{aligned} \min_{\epsilon_1^{-1},\ldots,\epsilon_n^{-1}} \quad & \mathbf{tr}\left(RXR^T\right) \\ \text{s.t.} \quad & \epsilon_{\max}^{-1} \leq {\epsilon_i}^{-1} \leq \epsilon_{\min}^{-1}, \ \forall i \in \mathcal{V} \\ & \mu \leq \sum_{i=1}^{n} \epsilon_i^{-1}, \\ & X = \frac{1}{2}\left((RWR^T)^{-1} + L_{e,s}^{\tau}\right), \end{aligned} \tag{P1}$$

where we have taken the effective variances, σ_ω and σ_v, to be unity. The objective is convex in the optimization variables $(1/\epsilon_1, \ldots, 1/\epsilon_n)$, and the constraints are linear. The design parameter of μ serves to ensure that not all the agents can operate on the slowest time scale (the trivial solution).

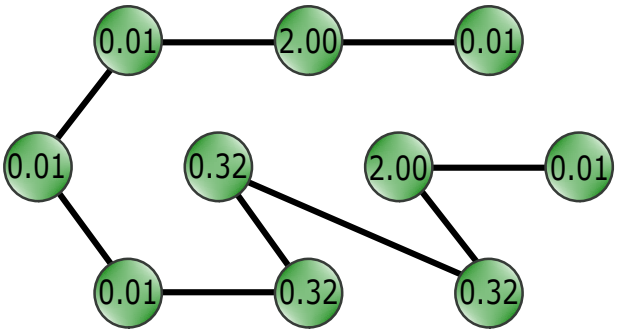
We solved Problem (P1) on random graphs (with probability of an edge between any two nodes as 0.15) that featured multiple independent cycles. For $n = 10$, $k = 1$, $\epsilon_{\min} = 0.01$, $\epsilon_{\max} = 2.0$ and $\mu = 510.5$ (which can be interpreted as the value allowing up to a third of the nodes to be slow), the results and graph topology for one such graph are presented Figure 4.7a. These results (which appear to hold over a wide range of randomly generated graphs) suggest that the presence of cycles do not detract from the heuristic developed for tree graphs; that high degree nodes should be assigned slow time scales (high scaling parameters) to minimize $\mathcal{H}_2(\Sigma^\tau)$. A note of further interest is that, due to the inclusion of the cycle



(a) Random graph topology for 10 agents.



(b) Original spanning tree.



(c) Spanning path graph.

Figure 4.7: Time scale assignment by (P1) are printed in each node, showing the slowest time scales are assigned to nodes with highest degree.

information in the output in (4.33), the optimal distribution from (P1) is independent of the selected spanning tree. We can see this by generating spanning trees with a variety of degree distributions, such as those in Figures 4.7b and 4.7c. However, we can see in (4.49) and (4.48) that the $\mathcal{H}_2$ performance for the Σ^τ system can be viewed as the performance for the $\hat{\Sigma}^\tau$ system with an additive term that encompasses the contribution of the cycle states. Consider then, the quantity

$$K = \frac{\mathcal{H}_2(\hat{\Sigma}^\tau, E)}{\mathcal{H}_2(\Sigma^\tau, E)},$$

which is a measure of how well the performance as measured by the spanning tree states

represents the graph performance including cycle information. For the spanning tree in Figure 4.7b we have $K \simeq 0.66$, and for the spanning tree in Figure 4.7c we have $K \simeq 0.24$. Intuitively, this reflects that spanning trees which more accurately reflect the true degree distribution of the parent graph will have a higher K. In line with [127], this also shows a significant portion of the $\mathcal{H}_2$ performance can come from the cycle contributions. In general, then, the performance of a given spanning tree may not be a good indicator of the full network performance, however, for graphs with few cycles, spanning trees that reproduce the full graph degree distribution closely can be a good approximation for the full network performance.

The time scale assignment problem considered here demonstrates that while the heuristic developed from results on tree graphs appears to hold for more complex graph topologies, the performance of a given spanning tree does not necessarily reflect the performance of the full tree. In the next section, we will consider a reformulation of this problem that allows for an analytic result to the optimal assignment while also reformulating the performance constraint.

4.2.3 Decentralized Updates for Optimal $\mathcal{H}_2$ Performance

Gradient Updates on Edge Weights

In the previous section, we saw that one could separate the contributions of the time scales and the edge weights on the $\mathcal{H}_2$ norm. We now present a design problem for optimizing the edge weight term of Equation (4.36). Consider Problem (P2),

$$\begin{aligned} \min_{\{W_i\}_{i=1}^{|\mathcal{E}|}} \ & \mathbf{tr}\left(\mathbf{R}^T(\mathbf{R}\mathbf{W}\mathbf{R}^T)^{-1}\mathbf{R}\right) + \frac{h}{2}\sum_{e\in\mathcal{E}} \mathbf{tr}\left[\mathbf{W}_e^T\mathbf{W}_e\right]^2 \\ \text{s.t. } & W_{\min} \preceq W_e \preceq W_{\max}, \forall e \in \mathcal{E} \\ & W = \text{blkdiag}(W_i). \end{aligned} \tag{P2}$$

We include a regularization term in the cost function to avoid the trivial solution of completely disconnecting the graph, as well as upper/lower bounds on the matrix weights. A

gradient update for solving Problem (P2) is derived in Proposition 5.

Proposition 5 (Gradient Update for Edge Weights)**.** *The gradient of the cost function with respect to the edge weight W_e in Problem (P2) is given by*

$$\nabla_{W_e} f[H] = -\text{deblk}_e^k \left[Q^T Q\right] + hW_e,$$

where $\text{deblk}_e^k[Q^T Q]$ *is the eth $k \times k$ diagonal block of $Q^T Q$, and Q is given by*

$$Q = \mathbf{R}^T \left(\mathbf{R}\mathbf{W}_H^c \mathbf{R}^T\right)^{-1} \mathbf{R}$$
$$\mathbf{W}_H^c = \text{blk}_c^k(H) + \sum_{l \in \mathcal{E}\setminus c} \text{blk}_l^k(W_l),$$

where

$$\text{blk}_c^k(H) = \mathbf{e}_c H \mathbf{e}_c^T = (e_c \otimes I_k)\, H \left(e_c^T \otimes I_k\right)$$

denotes the $kn \times kn$ matrix with H on the cth $k \times k$ diagonal block, with zeros otherwise.

A gradient update scheme for solving Problem (P2) is therefore

$$\begin{aligned} W_e^{k+1} &= W_e^k - \frac{1}{h\sqrt{k}} \nabla_{W_e} f[H] \\ &= W_e^k - \frac{1}{h\sqrt{k}} \left(hW_e - \text{deblk}_e^k \left[Q^T Q\right]\right). \end{aligned}$$

Proof. We first identify the gradient of the cost function with respect to W_c, the weight on the cth edge in $\mathcal{E}$. To this end, consider the functions

$$\begin{aligned} f(X) &= \mathbf{R} \left[\mathbf{e}_c X \mathbf{e}_c^T + \sum_{l \in \mathcal{E}\setminus c} \mathbf{e}_l W_l \mathbf{e}_l^T\right] \mathbf{R}^T \\ &= \mathbf{R} \left[\text{blk}_c^k(X) + \sum_{l \in \mathcal{E}\setminus c} \text{blk}_c^k(W_l)\right] \mathbf{R}^T \\ g(X) &= X^{-1},\ h(X) = \mathbf{tr}\left[\mathbf{R}^T X \mathbf{R}\right], \end{aligned}$$

where

$$\text{blk}_l^k(W_l) = \mathbf{e}_l W_l \mathbf{e}_l^T = (e_l \otimes I_k)\, W_l \left(e_l^T \otimes I_k\right)$$

denotes the $kn \times kn$ matrix with W_l on the lth $k \times k$ diagonal block, with zeros otherwise. Then, the cost function with W_c as the argument, is given by

$$\operatorname{tr}\left[\mathbf{R}^T\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}\mathbf{R}\right] = (h \circ g \circ f)(W_c). \tag{4.41}$$

These functions have differentials

$$\begin{aligned} df_X[H] &= \mathbf{R}\left[\operatorname{blk}_c^k(H)\right]\mathbf{R}^T,\ dg_X[H] = -X^{-1}HX^{-1} \\ dh_X[H] &= \operatorname{tr}\left[\mathbf{R}^T H \mathbf{R}\right]. \end{aligned}$$

By the chain rule, we have that

$$\begin{aligned} d(g \circ f)_X[H] &= -Y^{-1}\mathbf{R}\operatorname{blk}_c^k(H)\mathbf{R}^T Y^{-1} \\ Y \triangleq f[H] &= \mathbf{R}\left[\operatorname{blk}_c^k(H) + \sum_{l \in \mathcal{E}\setminus c} \operatorname{blk}_l^k(W_l)\right]\mathbf{R}^T \\ &\triangleq \mathbf{R}\mathbf{W}_H^c\mathbf{R}^T, \end{aligned}$$

and so

$$\begin{aligned} d(g \circ f)_X[H] &= -\left(\mathbf{R}\mathbf{W}_H^c\mathbf{R}^T\right)^{-1}\mathbf{R}\operatorname{blk}_c^k(H)\mathbf{R}^T\left(\mathbf{R}\mathbf{W}_H^c\mathbf{R}^T\right)^{-1} \\ d(h \circ g \circ f)_X[H] &= -\operatorname{tr}\left[Q\operatorname{blk}_k^c(H)Q^T\right],\ Q^T \triangleq \mathbf{R}^T\left(\mathbf{R}\mathbf{W}_H^c\mathbf{R}^T\right)^{-1}\mathbf{R}. \end{aligned}$$

Hence, we can write

$$\begin{aligned} d(h \circ g \circ f)_X[H] &= -\operatorname{tr}\left[Q\operatorname{blk}_k^c(H)Q^T\right] \\ &= -\operatorname{tr}\left[Q\mathbf{e}_c H \mathbf{e}_c^T Q^T\right] = \left\langle -\mathbf{e}_c^T Q^T Q \mathbf{e}_c, H\right\rangle, \end{aligned}$$

and so the gradient of (4.41) with respect to the cth weight W_c is identified as the cth $k \times k$ diagonal block of $-Q^TQ$.

□

Decentralized Time Scale Assignment

We saw previously in the definition of (P2) that a regularization term was included to prevent the trivial solution of disconnecting the graph. In the optimization of the time scale term of Equation (4.36), this trivial solution takes the form of all agents/substates adopting the slowest time scale parameter possible. Thus, consider (P3),

$$\begin{aligned} \min_{\epsilon_{1,1}^{-1},\ldots,\epsilon_{n,k}^{-1}} \quad & \frac{1}{2}\mathbf{tr}\left(\mathbf{R}^T\mathcal{L}_{e,s}^{\tau}\mathbf{R}\right) + \frac{h}{2}\sum_{i=1}^{n}\sum_{j=1}^{k}\epsilon_{i,j}^{r} \\ \text{s.t.} \quad & \epsilon_{\max}^{-1} \leq {\epsilon_{i,j}}^{-1} \leq \epsilon_{\min}^{-1} \ \forall i \in \mathcal{V},\ j \in [k]. \end{aligned} \tag{P3}$$

This is a minimization of the time scale portion of the separated $\mathcal{H}_2$ performance. A regularization term $2^{-1}h\sum_{i=1}^{n}\sum_{j=1}^{k}\epsilon_{i,j}^{r}$ penalizes large time scales for all nodes and their substates assuming positive, integer r. In the following proposition (proven in the Appendix), we show an analytic solution for the optimal time scale assignment which minimizes the $\mathcal{H}_2$ performance.

Proposition 6 (Analytic Optimal Time Scale Assignment)**.** *Consider* (P3)*. Let the region defined by the box constraints on* $1/\epsilon_{i,j}$ *be denoted by* $\mathcal{C}$*. Then, the minimizing assignment of time scale parameters is given by,*

$$\epsilon_{i,j}^{*} = Proj_{\mathcal{C}}\left[\left(\frac{deg(\nu_i)}{hr}\right)^{\frac{1}{r+1}}\right].$$

Proof. Consider the cost function (denoted by $f(\epsilon_{i,j}^{-1})$) without the box constraint, and note that the $\mathcal{H}_2$ portion can be rewritten as a double sum of the same form as the regularization term,

$$\mathbf{tr}\left(\mathbf{R}^T\mathcal{L}_{e,s}^{\tau}\mathbf{R}\right) = \mathbf{tr}\left(\mathbf{E}^{-1}\mathcal{L}\right) = \sum_{i=1}^{n}\sum_{j=1}^{k}\epsilon_{i,j}^{-1}\deg(\nu_i),$$

resulting in,

$$f(\epsilon_{i,j}^{-1}) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{k}\epsilon_{i,j}^{-1}\deg(\nu_i) + \frac{h}{2}\sum_{i=1}^{n}\sum_{j=1}^{k}(\epsilon_{i,j}^{-1})^{-r}.$$

Minimizing this cost alone can be achieved by setting its gradient equal to zero,

$$\frac{\partial f}{\partial \epsilon_{i,j}^{-1}} = \frac{\deg(\nu_i)}{2} - \frac{hr}{2}(\epsilon_{i,j}^{-1})^{-(r+1)} = 0,$$

implying that, $\epsilon_i^* = (\deg(\nu_i)/hr)^{1/r+1}$. Projecting this result onto the constraint set gives the result. □

Remark 8. *The assignment rule in Proposition 6 is decentralized, as the optimal assignment value depends only on the (unweighted) degree of the i-th node and the parameters h and r, which are locally known to the i-th node without global knowledge of the network topology.*

From this result we can see that for a class of regularization terms, the optimal time scale assignment is again driven by the degree distribution, which is in-line with the previous results. It is conceivable to consider using this result with online signal identification to locally adjust time scales in response to adversarial noise entering the system.

4.2.4 Example: Flocking via Second-Order Consensus

Flocking is a behaviour exhibited by certain multi-agent systems that are coordinating their motion into a cohesive formation, for example birds or stampeding buffalo. A consensus-type algorithm can be proposed that allows a system of n agents to agree on their velocity vector while maintaining a separation from their neighbours [21].

Matrix-Valued Double Integrator Consensus

In the case of vector-valued states, we can write the dynamics as

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \mathbf{E}\ddot{\mathbf{x}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & I \\ -\mathcal{L}_w & -\mathcal{L}_w \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \dot{\mathbf{x}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ I & -\mathbf{D}_{\mathcal{G}} \end{bmatrix} \begin{bmatrix} \omega \\ v \end{bmatrix}.$$

Theorem 5. *Transforming the double-integrator dynamics to the edge states results in dynamics over the spanning tree edge states. We can see this mathematically, by considering*

the double-integrator edge consensus model given by

$$\begin{bmatrix}\dot{\mathbf{x}}_e\\ \ddot{\mathbf{x}}_e\end{bmatrix} = \begin{bmatrix}\mathbf{0} & 0 & I & \\ -\mathcal{L}_e\mathbf{R}\mathbf{W}\mathbf{R}^T & \mathbf{0} & -\mathcal{L}_e\mathbf{R}\mathbf{W}\mathbf{R}^T & \mathbf{0}\\ \mathbf{0} & 0 & \mathbf{0} & 0\end{bmatrix}\begin{bmatrix}\mathbf{x}_e\\ \dot{\mathbf{x}}_e\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{D}_\tau^T\mathbf{E}^{-1} & -\mathcal{L}_e\mathbf{R}\\ \Xi^{-1}\mathbf{1}^T & \mathbf{0}\end{bmatrix}\begin{bmatrix}\omega\\ v\end{bmatrix},$$

and so the double-integrator consensus on the edge states of the chosen spanning tree τ is given by

$$\begin{bmatrix}\dot{\mathbf{x}}_\tau\\ \ddot{\mathbf{x}}_\tau\end{bmatrix} = \begin{bmatrix}\mathbf{0} & I\\ -\mathcal{L}_e\mathbf{R}\mathbf{W}\mathbf{R}^T & -\mathcal{L}_e\mathbf{R}\mathbf{W}\mathbf{R}^T\end{bmatrix}\begin{bmatrix}\mathbf{x}_\tau\\ \dot{\mathbf{x}}_\tau\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{D}_\tau^T & -\mathcal{L}_e\mathbf{R}\end{bmatrix}\begin{bmatrix}\omega\\ v\end{bmatrix}. \tag{4.42}$$

Proof. Applying the coordinate transform $\mathbf{x}_e = S_v\mathbf{x}$ and following a similar calculation as in Theorem 4 yields the result. □

We can also explicitly compute the form of the $\mathcal{H}_2$ norm for the matrix-weighted double-integrator consensus.

Theorem 6. *The performance of the time scaled, weighted, double-integrator consensus* (4.42) *can be decomposed into the performance of the position and velocity states separately, which allows for a block-diagonal controllability gramian.*

Explicitly, the controllability gramian for the time scaled double-integrator consensus (4.42) *is given by*

$$\begin{aligned}\mathbf{X}^* &= \frac{1}{2}\begin{bmatrix}\mathbf{X}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{X}_2\end{bmatrix}\\ \mathbf{X}_1 &= \sigma_w^2\left[\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}\mathcal{L}_e^{-1}\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}\right] + \sigma_v^2\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}\\ \mathbf{X}_2 &= \sigma_w^2\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1} + \sigma_v^2\mathcal{L}_e.\end{aligned}$$

Furthermore, the blocks $\mathbf{X}_1, \mathbf{X}_2$ *of* $\mathbf{X}^*$ *correspond to the position and velocity states, respectively. Hence, one can consider the* $\mathcal{H}_2$ *performance of the position and velocity states separately or aggregately by examining the* $\mathcal{H}_2$ *norms*

$$\mathcal{H}_2{}^{(1)} = \frac{1}{2}\mathbf{tr}\left(\mathbf{R}^T\mathbf{X}_1\mathbf{R}\right), \ \mathcal{H}_2{}^{(2)} = \frac{1}{2}\mathbf{tr}\left(\mathbf{R}^T\mathbf{X}_2\mathbf{R}\right)$$
$$\mathcal{H}_2{}^{(1)} = \frac{1}{2}\mathbf{tr}\left(\mathbf{R}^T(\mathbf{X}_1+\mathbf{X}_2)\mathbf{R}\right).$$

Proof. From the dynamics (4.42), the controllablity gramian is given by the positive semi-definite solution to the Lyapunov equation

$$A\mathbf{X}^* + \mathbf{X}^*A^T + BB^T = \mathbf{0}, \tag{4.43}$$

where $\mathbf{X}^*, A$, and BB^T are given by,

$$A = \begin{bmatrix} \mathbf{0} & I \\ -\mathcal{L}_e\mathbf{R}\mathbf{W}\mathbf{R}^T & -\mathcal{L}_e\mathbf{R}\mathbf{W}\mathbf{R}^T \end{bmatrix},$$
$$BB^T = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \sigma_w^2\mathcal{L}_e + \sigma_v^2\mathcal{L}_e\mathbf{R}\mathbf{R}^T\mathcal{L}_e \end{bmatrix},$$
$$\mathbf{X}^* = \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_3 \\ \mathbf{X}_3 & \mathbf{X}_2 \end{bmatrix}.$$

Solving Equation (4.43) yields

$$\mathbf{X}_3 = \mathbf{0}, \ \mathbf{X}_2 = \sigma_w^2\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1} + \sigma_v^2\mathcal{L}_e$$
$$\mathbf{X}_2 = \mathbf{X}_1\mathbf{R}\mathbf{W}\mathbf{R}^T\mathcal{L}_e,$$

and solving for $\mathbf{X}_1$ in the last display yields

$$\mathbf{X}_1 = \sigma_w^2\left[\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}\mathcal{L}_e^{-1}\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}\right] + \sigma_v^2\left(\mathbf{R}\mathbf{W}\mathbf{R}^T\right)^{-1}.$$

To measure the position, velocity, or both states for consideration in the $\mathcal{H}_2$ norm, one can choose the observation matrices

$$\begin{bmatrix}\mathbf{R}^T & \mathbf{0}\end{bmatrix}, \ \begin{bmatrix}\mathbf{0} & \mathbf{R}^T\end{bmatrix}, \ \begin{bmatrix}\mathbf{R}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{R}^T\end{bmatrix},$$

respectively. □

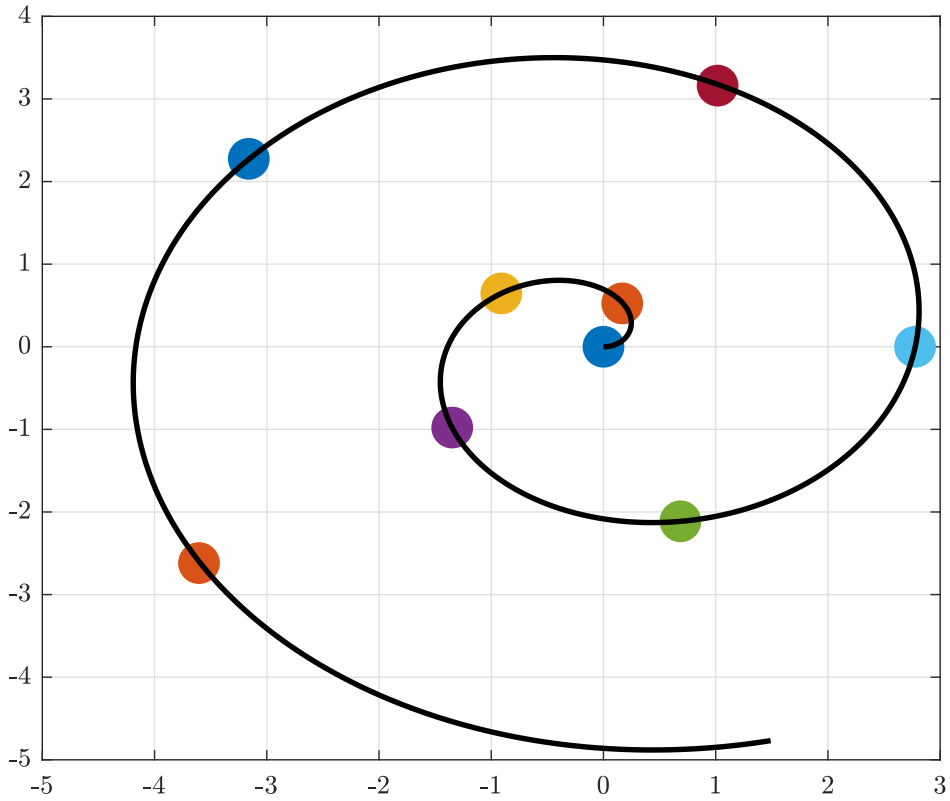


Figure 4.8: Relative formation of the agents, defined by discrete points on a spiral.

Numerical Example

The weight update scheme applied to the second-order consensus problem was implemented numerically. The task assigned to the agents was to use the second-order consensus protocol to achieve the formation shown in Figure 4.8 – a formation assigned by sampling discrete points on a 2D spiral.

Consensus on the formation is achieved by the second-order protocol with a constant signal $\mathbf{d}_i$ specifying the position in the formation:

$$\ddot{\mathbf{x}}_i = -\sum_{j\in\mathcal{N}(i)} \mathbf{W}_{ij}(\mathbf{x}_i - \mathbf{x}_j - \mathbf{d}_i) - \sum_{j\in\mathcal{N}(i)} \mathbf{W}_{ij}(\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_j). \tag{4.44}$$

Since this is a constant signal, the $\mathcal{H}_2$ performance of the edge states remains the same as in the previous section.

The initial selection of weights was chosen at random using the generator

$$W_{\{i,j\}} = \alpha \left[\frac{G_{\{i,j\}} + G_{\{i,j\}}^T}{2} + 2I_2 \right], \tag{4.45}$$

where $G_{\{i,j\}}$ is a 2×2 matrix with entries distributed according to a zero-mean standard Gaussian, and $\alpha = 0.3$ was chosen arbitrarily to yield a suboptimal initial selection of weights. The upper and lower bounds on the weights were chosen with the same generator in Equation (4.45), but with $\alpha_l = 0.05$ and $\alpha_u = 10$. The initial time scale parameters were taken to be identically unity. The penalty parameter was chosen as $h = 0.01$.

The gradient descent algorithm from Proposition 5 converges quickly, and intermediate graph weights are visualized in Figure 4.9. None of the optimal edge weights saturated the upper and lower bounds in this setup. The minimizing time scale assignment was calculated using Proposition 6.

These weights and scaling paramters were then used in a simulation of the dynamics in (4.44) over a time span of 30 seconds. At $10 \leq t \leq 20$, the formation is subject to a 'gust' of noise on the nodes and edges with covariance $\sigma_w^2 I$ and $\sigma_v^2 \mathbf{W}$, with $\sigma_v = \sigma_w = 5$. Simulations were performed with no updates (NUD), updates to edge weights (WUD) or time scales (TUD) during the wind gusts, and with both updates (BUD) during gusts. The edge states for the x and y directions for these four cases are shown in Figure 4.10, and the variance away from the consensus value $x_e(t_f)$, $\mathrm{Var}[x_e(t)] := [x_e(t) - x_e(t_f)]^2$, is shown in Figure 4.11. The updated weights and scales outperform their initial, suboptimal values.

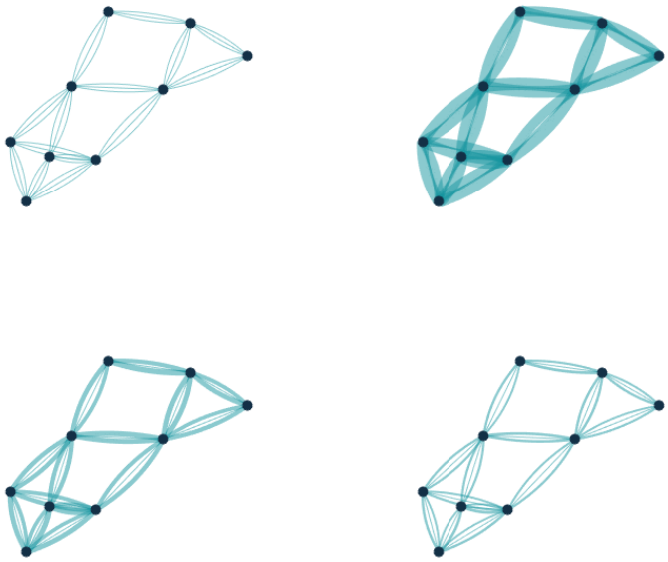


Figure 4.9: Visualization of edge weights over iterations $k = 1$ (top left), 2 (top right), 3 (bottom left) & 4 (bottom right). Each of the 3 independent parameters of the 2×2 matrix-valued weight is visualized in a multigraph.

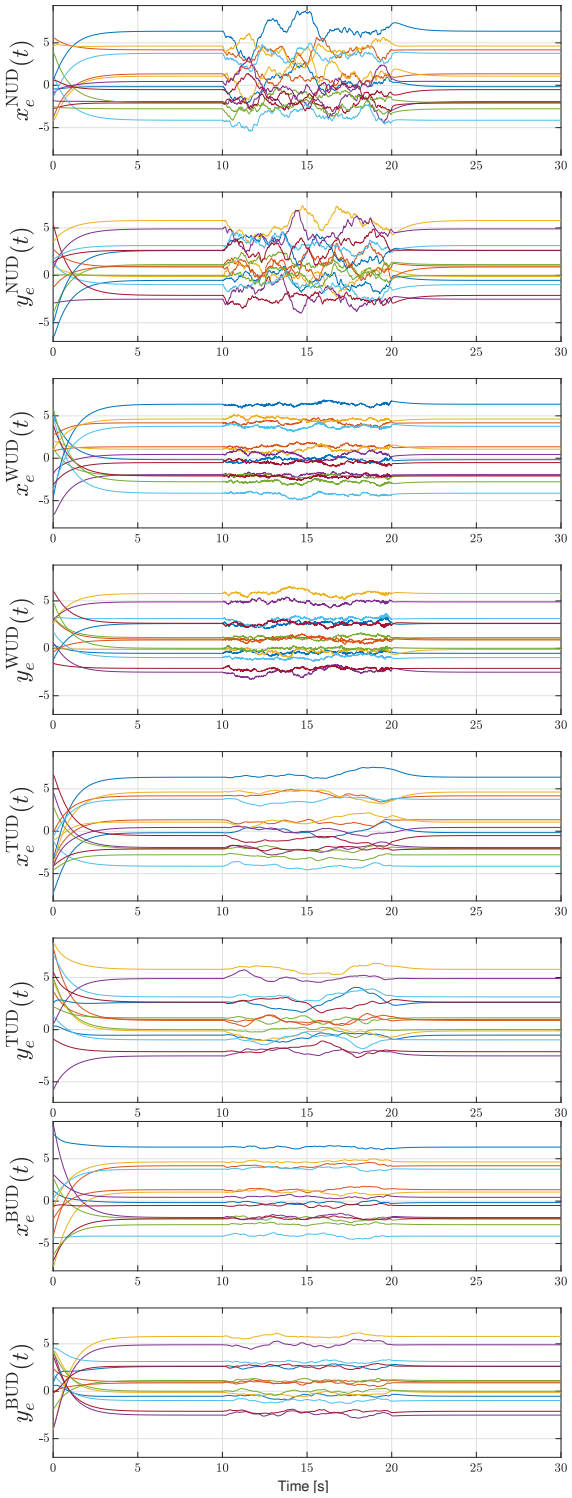


Figure 4.10: Edge states in x, y directions over time subjected to gust at $10 \leq t \leq 20$; cases from top to bottom are with no updates (NUD), weight updates (WUD), time scale update (TUD), and both updates (BUD).

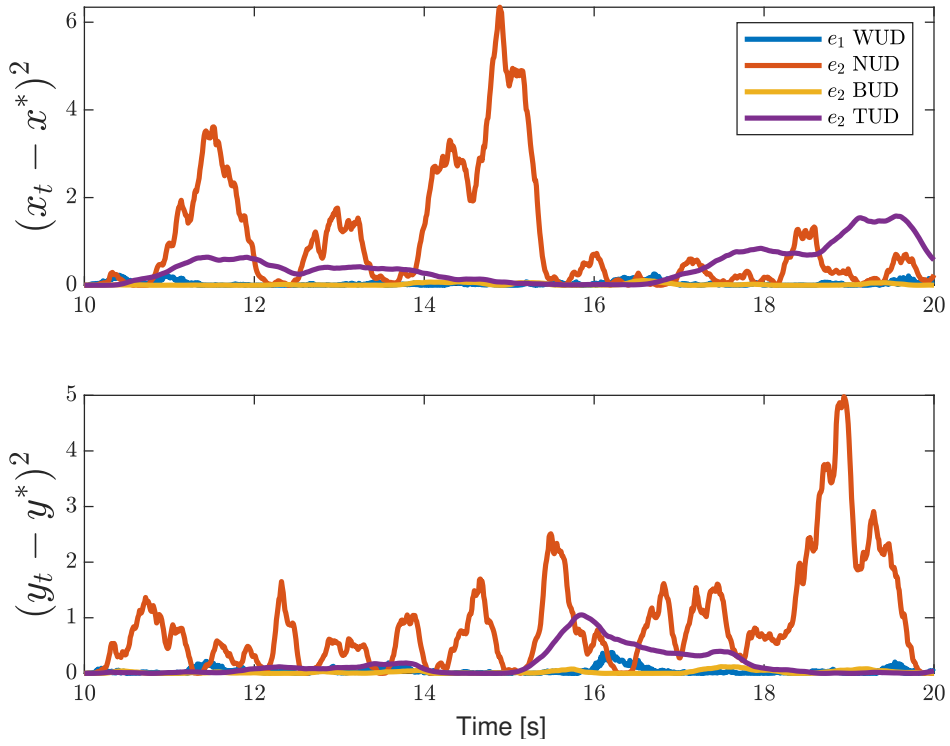


Figure 4.11: Variance from consensus for a representative edge $[x_e(t) - x_e(t_f)]^2$ for the case with the weight update (WUD), time scale update (TUD), both updates (BUD), and with no updates (NUD).

4.3 Graph-Theoretic Optimization for Edge Consensus[4]

In the previous section, we saw results that suggested, given time scaled and matrix weighted graphs, that all spanning trees were not considered equal, and selection of optimal spanning trees might be a fruitful endeavor. Here we present some minor results in this vein.

To begin, recall that it was noted in [35] that the ${\mathcal{H}_2}^2$ performances could be written as,

$$\begin{aligned}
&\mathcal{H}_2^2(\Sigma^{\mathcal{T}}(\mathcal{G})) = \mathbf{tr}\left[R_{\mathcal{G}}^{\intercal} X^* R_{\mathcal{G}}\right] \\
&= \frac{\sigma_\omega^2}{2}\mathbf{tr}\left[R_{\mathcal{G}}^{\intercal}(R_{\mathcal{G}} W_{\mathcal{G}} R_{\mathcal{G}}^{\intercal})^{-1} R_{\mathcal{G}}\right] + \frac{\sigma_v^2}{2}\mathbf{tr}\left[R_{\mathcal{G}}^{\intercal}\mathcal{L}^{\mathcal{T}} R_{\mathcal{G}}\right] \qquad (4.46) \\
&\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}(\mathcal{G})) = \mathbf{tr}\left[X^*\right] \\
&= \frac{\sigma_\omega^2}{2}\mathbf{tr}\left[(R_{\mathcal{G}} W_{\mathcal{G}} R_{\mathcal{G}}^{\intercal})^{-1}\right] + \frac{\sigma_v^2}{2}\mathbf{tr}\left[\mathcal{L}^{\mathcal{T}}\right]. \qquad (4.47)
\end{aligned}$$

for with the contributions of the weights and time scales can be separated in (4.46) & (4.47) as,

$$\begin{aligned}
\mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{G}]) &= \mathcal{H}_2^2(\Sigma^{\mathcal{T}}; W) + \mathcal{H}_2^2(\Sigma^{\mathcal{T}}; E) \\
\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}[\mathcal{G}]) &= \mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}; W) + \mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}; E)
\end{aligned}$$

where

$$\begin{aligned}
\mathcal{H}_2^2(\Sigma^{\mathcal{T}}; W) &\triangleq \frac{\sigma_\omega^2}{2}\mathbf{tr}\left[R_{\mathcal{G}}^{\intercal}(R_{\mathcal{G}} W_{\mathcal{G}} R_{\mathcal{G}}^{\intercal})^{-1} R_{\mathcal{G}}\right] \\
\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}; W) &\triangleq \frac{\sigma_w^2}{2}\mathbf{tr}\left[(R_{\mathcal{G}} W_{\mathcal{G}} R_{\mathcal{G}}^{\intercal})^{-1}\right] \\
\mathcal{H}_2^2(\Sigma^{\mathcal{T}}; E) &\triangleq \frac{\sigma_v^2}{2}\mathbf{tr}\left[R_{\mathcal{G}}^{\intercal}\mathcal{L}^{\mathcal{T}} R_{\mathcal{G}}\right] \\
\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}; E) &\triangleq \frac{\sigma_v^2}{2}\mathbf{tr}\left[\mathcal{L}^{\mathcal{T}}\right].
\end{aligned}$$

[4]Text in this section adapted from "Graph-Theoretic Optimization for Edge Consensus" accepted for 24th International Symposium on Mathematical Theory of Networks and Systems, Cambridge, UK [24]. Some results in this section are included for necessary context/discussion, but are not primary contributions of DRF (MHdB contribution). Subsection titles with astericks (*) denote primary MHdB/coauthor contributions.

Furthermore, it was shown in [35] that the $\mathcal{H}_2$ norm of Σ can be written in terms of the $\mathcal{H}_2$ norm of $\hat{\Sigma}$ as follows:

$$\mathcal{H}_2^2(\Sigma^{\mathcal{T}}; W) = \mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}; W) + \frac{\sigma_w^2}{2}\text{tr}\left[T_{\mathcal{T}}^{\intercal}(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}})^{-1}T_{\mathcal{T}}\right] \tag{4.48}$$

$$\mathcal{H}_2^2(\Sigma^{\mathcal{T}}; E) = \mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}; E) + \frac{\sigma_v^2}{2}\mathbf{tr}\left[T_{\mathcal{T}}^{\intercal}L^{\mathcal{T}}T_{\mathcal{T}}\right]. \tag{4.49}$$

We examine the behaviour of each of these components when an edge is added to a spanning tree – $\mathcal{H}_2^2(\hat{\Sigma}, W)$ and $\mathcal{H}_2^2(\hat{\Sigma}, E)$ are discussed in §4.3.2, and $\mathcal{H}_2^2(\Sigma, E)$ & $\mathcal{H}_2^2(\Sigma, W)$ in §4.3.3 and §4.3.4 respectively.

4.3.1 Minimum-$\mathcal{H}_2$ Spanning Tree

In this section, we show how a minimum-$\mathcal{H}_2$ norm spanning tree of a graph $\mathcal{G}$ can be found with a greedy algorithm. Let $\mathcal{G}$ be a weighted, time scaled graph with edge consensus dynamics (4.43) and corresponding $\mathcal{H}_2$ norm (4.47).

Suppose that we want to remove communication links from $\mathcal{G}$ until we are left with a minimally-connected graph — a tree. Such an operation can be used during a 'stealth mode' for networked systems, during which time having the minimum number of communication links while still being connected is desired. For example, a swarm of UAVs may be using consensus to agree on a formation heading, but may want to limit their chance of detection when entering a hostile area. In this section, we show that one can choose the spanning tree $\mathcal{T} \subseteq \mathcal{G}$ that minimizes the $\mathcal{H}_2$ norm in (4.47) using a greedy algorithm. This allows the network to minimize the number of communication links used in the consensus algorithm, while maintaining optimal noise rejection properties. The main result of this section is summarized in Proposition 7.

Proposition 7. *Consider a graph $\mathcal{G}$ with the edge consensus dynamics (4.43), and with corresponding $\mathcal{H}_2$ norm (4.47). Any spanning tree $\mathcal{T}' \subseteq \mathcal{G}$ has $\mathcal{H}_2$ norm*

$$\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}'}) = \frac{\sigma_\omega^2}{2}\sum_{e\in\mathcal{T}'} w_e^{-1} + \frac{\sigma_v^2}{2}\sum_{i=1}^{n}\frac{\deg(i)}{\epsilon_i}. \tag{4.50}$$

Algorithm 1 Minimum–$\mathcal{H}_2$ Spanning Tree

1: *Input:* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}, \mathcal{S})$

2: *Initialize:*

3: Choose $v \in \mathcal{V}$, set $\mathcal{Q} = \mathcal{V} \setminus \{v\}, \mathcal{P} = \{v\}$, $\mathcal{R} = \emptyset$

4: **while** $|\mathcal{Q}| > 0$ **do**

5: *Set*

$$N(\mathcal{P}) = \{j \in \mathcal{Q} \ : \ i \in \mathcal{P}, \ ij \in \mathcal{E}\}.$$

6: *Solve*

$$\text{(P1)} := \min_{j \in N(\mathcal{P})} \left\{ \sigma_v^2(\epsilon_k^{-1} + \epsilon_j^{-1}) + \sigma_\omega^2 w_{jk}^{-1} \ : \ jk \in \mathcal{E} \right\}$$

7: *From* $\arg\min$ (P1) $:= (j^*, j^*k)$, *update*

$$\mathcal{P} \mapsto \mathcal{P} \cup \{j^*\}$$
$$\mathcal{R} \mapsto \mathcal{R} \cup \{j^*k\}$$
$$\mathcal{Q} \mapsto \mathcal{Q} \setminus \{j^*\}$$

8: *Output:* $\mathcal{T}^* = (\mathcal{P}, \mathcal{R}, \mathcal{W}(\mathcal{R}), \mathcal{S})$

Algorithm 1 returns a spanning tree $\mathcal{T}^* \subseteq \mathcal{G}$ that minimizes (4.50) out of all possible spanning trees $\mathcal{T} \subseteq \mathcal{G}$.

Proof. Define an auxiliary graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{W}')$ on the same vertex and edge set as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}, \mathcal{S})$, (i.e., $\mathcal{V}' = \mathcal{V},\ \mathcal{E}' = \mathcal{E}$) but with edge weights $w(\mathcal{G}') \in \mathcal{W}'$ defined by

$$w(\mathcal{G}')_{ij} = \frac{\sigma_v^2}{\epsilon_i} + \frac{\sigma_v^2}{\epsilon_j} + \frac{\sigma_\omega^2}{w_{ij}},\ \forall\ i, j \in \mathcal{V},\ ij \in \mathcal{E}. \tag{4.51}$$

Then, define the *total cost* of the graph $\mathcal{G}'$ as the sum of the edge weights:

$$TC(\mathcal{G}') = \frac{1}{2} \sum_{e \in \mathcal{E}'} w(\mathcal{G}')_e. \tag{4.52}$$

It is clear from (4.51) that

$$TC(\mathcal{G}') = \frac{1}{2} \sum_{e \in \mathcal{E}'} w(\mathcal{G}')_e = \frac{1}{2} \sum_{ij \in \mathcal{E}} \left[\frac{\sigma_v^2}{\epsilon_i} + \frac{\sigma_v^2}{\epsilon_j} + \frac{\sigma_\omega^2}{w_{ij}} \right]. \tag{4.53}$$

For every edge ij we get one term with ϵ_i^{-1} in the sum of (4.53), and so summing over all edges yields $\deg(i)$ terms ϵ_i^{-1}. Hence, we can conclude that from (4.47), for any spanning tree $\mathcal{T}' \subseteq \mathcal{G}'$, its total cost is exactly the $\mathcal{H}_2$ norm of the corresponding tree $\mathcal{T}$ in $\mathcal{G}$.

$$TC(\mathcal{T}') = \frac{1}{2} \left[\sum_{e \in \mathcal{E}} \frac{\sigma_\omega^2}{w_e} + \sum_{i \in \mathcal{V}} \frac{\sigma_v^2 \deg(i)}{\epsilon_i} \right] = \mathcal{H}_2^2(\Sigma^{\mathcal{T}}). \tag{4.54}$$

By contruction, the minimal spanning tree $\mathcal{T}'^*$ of $\mathcal{G}'$, defined by

$$\mathcal{T}'^* := \arg\min \left\{ TC(\mathcal{T}') \ :\ \mathcal{T}' \text{ is a spanning tree of } \mathcal{G}' \right\},$$

corresponds precisely to the spanning tree $\mathcal{T}$ of $\mathcal{G}$ minimizing the $\mathcal{H}_2$ norm in (4.47).

It is well-known that a minimal-weighted spanning tree of $\mathcal{G}'$ can be obtained by the Jarník-Dijkstra-Prim, or Kruskal algorithms; see [108]. Algorithm 1 is precisely a Jarník-Dijkstra-Prim algorithm that builds the minimal-weighted spanning tree $\mathcal{T}^*$ by starting with a single node in $\mathcal{T}^*$, and at each iteration adding a new node to $\mathcal{T}^*$ by picking the neighbouring vertex k to $\mathcal{T}^*$ (but k not already in $\mathcal{T}^*$) with the smallest term $\sigma_\omega^2 w_{jk}^{-1} + \sigma_v^2(\epsilon_j^{-1} + \epsilon_k^{-1})$ for $j \in \mathcal{T}^*$. By the above argument, this is equivalent to performing Jarník-Dijkstra-Prim on $\mathcal{G}'$ with cost $TC(\mathcal{G}')$ in (4.52). □

Remark 9. *For an edge-weighted graph where the cost is the sum of the edge weights, it is known that if the edge weights are distinct, then the minimal-weighted spanning tree is unique. The minimum-$\mathcal{H}_2$ spanning tree $\mathcal{T}^*$ generated by Algorithm 1 is not unique, even if all the nodal time scales and edge weights are distinct. See §4.3.5 for an example.*

Remark 10. *Another practical algorithm would be a distributed version of Algorithm 1, which could be constructed by considering a distributed minimal-weighted spanning tree algorithm such as [4, 42, 65]. Such an algorithm would allow a consensus network to autonomously restructure itself to the $\mathcal{H}_2$-optimal tree configuration in a distributed manner.*

*4.3.2 Weighted Cycle Selection**

In the previous section, we discussed how one may find the minimum-$\mathcal{H}_2$ norm spanning tree from a connected graph $\mathcal{G}$. In this section, we examine what happens to the $\mathcal{H}_2$ norm when edges are added to a spanning tree. In particular, we show that adding weighted edges improve the $\mathcal{H}_2$ norm, and discuss what choice of edge optimizes this improvement. This can be used in the previous motivating example of a stealthy system that wants to still minimize the number of communication links, but does not reject noise well enough with just the links in the spanning tree.

[128] discussed how adding cycles to an unweighted, mono-scaled graph impacts the $\mathcal{H}_2$ performance in the presence of unit covariance noise applied to the edges and nodes. In particular, they found that the biggest improvement in $\mathcal{H}_2$ performance resulted from adding an edge to a tree $\mathcal{T}$ which maximized the length of the resulting cycle. Our first contribution is showing that in the presence of edge weights and time scales, the length of the cycle no longer matters, rather one should add an edge to form a cycle that, roughly speaking, has small weights.

First, we define the *weighted length* and *unweighted length* of a cycle $\mathcal{C}$ as,

$$l_w(\mathcal{C}) := \sum_{e\in\mathcal{C}} w_e^{-1},\ l(\mathcal{C}) := \sum_{e\in\mathcal{C}} 1.$$

Proposition 8. *Consider a graph consisting of a weighted, time scaled tree $\mathcal{T}$, and consider the dynamics $\hat{\Sigma}^{\mathcal{T}}$, with the $\mathcal{H}_2$ norm of $\hat{\Sigma}^{\mathcal{T}}$ given by Equation (4.47). Consider the task of adding edges to $\mathcal{T}$ to optimize the $\mathcal{H}_2$ norm of $\hat{\Sigma}^{\mathcal{T}}$. Then, the following hold:*

1. *Adding an edge e with weight $W_{\mathcal{C}}$ to $\mathcal{T}$ improves the $\mathcal{H}_2$ performance in the following manner:*

$$\mathcal{H}_2^2(\hat{\Sigma}[\mathcal{T} \cup e]) = \mathcal{H}_2^2(\hat{\Sigma}[\mathcal{T}]) - \frac{\sigma_\omega^2}{2l_w(\mathcal{C})} \sum_{ij \in \mathcal{T} \cap \mathcal{C}} w_{ij}^{-2}, \tag{4.55}$$

   where $\mathcal{C}$ is the unique cycle formed by adding edge e to $\mathcal{T}$. Furthermore, adding an edge to $\mathcal{T}$ to form a cycle always decreases the $\mathcal{H}_2$ norm.

2. *Adding p edge-disjoint cycles $\mathcal{C}_1, \ldots, \mathcal{C}_p$ via edges $e_1, \ldots, e_p$ improves the $\mathcal{H}_2$ performance in the following manner:*

$$\begin{aligned} &\mathcal{H}_2^2(\hat{\Sigma}[\mathcal{T} \cup \{e_i\}]) \\ &= \mathcal{H}_2^2(\hat{\Sigma}[\mathcal{T}]) - \sum_{k=1}^{p} \left[\frac{\sigma_\omega^2}{2l_w(\mathcal{C}_k)} \sum_{ij \in \mathcal{T} \cap \mathcal{C}_k} w_{ij}^{-2} \right], \end{aligned}$$

Proof. Recall that the $\mathcal{H}_2$ norm of (4.43) for a given spanning tree $\mathcal{T} \subseteq \mathcal{G}$ is given by

$$\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}}[\mathcal{G}]) = \frac{\sigma_\omega^2}{2} \mathbf{tr}\left[(R_{\mathcal{G}} W_{\mathcal{G}} R_{\mathcal{G}}^{\mathsf{T}})^{-1} \right] + \frac{\sigma_v^2}{2} \mathbf{tr}\left[\mathcal{L}^{\mathcal{T}} \right].$$

Since the graph in question is an edge added to a tree, we have $\mathcal{G} = \mathcal{T} \cup e$. This introduces one cycle, $\mathcal{C}$. Our choice of spanning tree in the cycle representation matrix $T_{\mathcal{T}}$ is the original tree $\mathcal{T}$ to which the edge e is added. Consider the trace argument in the weight matrix term,

$$\begin{aligned} (R_{\mathcal{G}} W_{\mathcal{G}} R_{\mathcal{G}}^{\mathsf{T}})^{-1} &= \left(\begin{bmatrix} I & T_{\mathcal{T}} \end{bmatrix} \begin{bmatrix} W_{\mathcal{T}} & \mathbf{0} \\ \mathbf{0} & W_{\mathcal{C}} \end{bmatrix} \begin{bmatrix} I \\ T_{\mathcal{T}}^{\mathsf{T}} \end{bmatrix} \right)^{-1} \\ &= (W_{\mathcal{T}} + T_{\mathcal{T}} W_{\mathcal{C}} T_{\mathcal{T}}^{\mathsf{T}})^{-1} = (W_{\mathcal{T}} + T' T'^{\mathsf{T}})^{-1}, \end{aligned}$$

where $W_{\mathcal{T}}$ is the top-left block of $W_{\mathcal{G}}$ corresponding to the weights of the edges in $\mathcal{T}$, and $W_{\mathcal{C}}$ is the bottom-right block of $W_{\mathcal{G}}$ corresponding to the weight of the edge being added to $\mathcal{T}$. In the last display, we made the substitution $T' \triangleq T_{\mathcal{T}} W_{\mathcal{C}}^{1/2}$.

Since we are adding a single edge e to $\mathcal{T}$, $W_{\mathcal{C}}$ is a scalar and T' is a vector. Therefore, $T'T'^{\intercal}$ is a rank-one matrix, and we can apply the Sherman-Morrison-Woodbury update, yielding

$$\begin{aligned}(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}}^{\intercal})^{-1} &= W_{\mathcal{T}}^{-1} - \frac{W_{\mathcal{T}}^{-1}T'T'^{\intercal}W_{\mathcal{T}}^{-1}}{1+T'^{\intercal}W_{\mathcal{T}}^{-1}T'} \\ &= W_{\mathcal{T}}^{-1} - \frac{W_{\mathcal{T}}^{-1}T_{\mathcal{T}}W_{\mathcal{C}}T_{\mathcal{T}}^{\intercal}W_{\mathcal{T}}^{-1}}{1+W_{\mathcal{C}}\left(T_{\mathcal{T}}^{\intercal}W_{\mathcal{T}}^{-1}T_{\mathcal{T}}\right)}. \end{aligned} \tag{4.56}$$

We recall the following lemma.

Lemma 5 (Prop 1 & 2 in [128]). *Recall the definition of the cycle representation matrix:*

$$T_{\mathcal{T}} = (D_{\mathcal{T}}^{\intercal}D_{\mathcal{T}})^{-1}D_{\mathcal{T}}^{\intercal}D_{\mathcal{G}\setminus\mathcal{T}} := \begin{bmatrix} c_1 & \cdots & c_{|\mathcal{E}(\mathcal{G}\setminus\mathcal{T})|}\end{bmatrix}.$$

Each column c_i of $T_{\mathcal{T}}$ represents a cycle $\mathcal{C}_i$ of $\mathcal{G}$. The matrices $T_{\mathcal{T}}^{\intercal}T_{\mathcal{T}}$ and $T_{\mathcal{T}}T_{\mathcal{T}}^{\intercal}$ encode the following information about the cycles of $\mathcal{G}$:

1. $[T_{\mathcal{T}}^{\intercal}T_{\mathcal{T}}]_{ii} = l(\mathcal{C}_i) - 1$
2. $[T_{\mathcal{T}}^{\intercal}T_{\mathcal{T}}]_{ij} = 0$ *if and only if the cycles c_i and c_j are edge-disjoint.*
3. $[T_{\mathcal{T}}T_{\mathcal{T}}^{\intercal}]_{ee}$ *is the number of times edge e is used to construct the cycles of $\mathcal{G}$.*

Consider the graph in question, $G := \mathcal{T} \cup e$. Adding this edge introduces a single cycle, and so by Lemma 5,

$$T_{\mathcal{T}}^{\intercal}T_{\mathcal{T}} = c_1^T c_1 = l(\mathcal{C}) - 1.$$

This holds because $T_{\mathcal{T}}^{\intercal}T_{\mathcal{T}}$ adds a '1' to the sum for each edge in the tree that ends up in the cycle $\mathcal{C}$, in other words we can write

$$T_{\mathcal{T}}^{\intercal}T_{\mathcal{T}} = \left[\sum_{e\in\mathcal{T}\cap\mathcal{C}}(1)\right] = \left[\sum_{e\in\mathcal{C}}(1)\right] - 1,$$

where the '-1' comes from excluding the edge e that is added to $\mathcal{T}$. Similarly, it can be seen that

$$\begin{aligned}(T_{\mathcal{T}})^{\intercal}W_{\mathcal{T}}^{-1}T_{\mathcal{T}} &= \sum_{ij\in\mathcal{C}\cap\mathcal{T}} w_{ij}^{-1} \\ &= \left[\sum_{ij\in\mathcal{C}} w_{ij}^{-1}\right] - W_{\mathcal{C}}^{-1} = l_w(\mathcal{C}) - W_{\mathcal{C}}^{-1}.\end{aligned}$$

From this, we can conclude that

$$\begin{aligned}1 + W_{\mathcal{C}}(T_{\mathcal{T}})^{\intercal}W_{\mathcal{T}}^{-1}T_{\mathcal{T}} &= 1 + W_{\mathcal{C}}T^{\intercal}W_{\mathcal{T}}^{-1}T \\ &= 1 + W_{\mathcal{C}}\left(l_w(\mathcal{C}) - W_{\mathcal{C}}^{-1}\right) = W_{\mathcal{C}}l_w(\mathcal{C}),\end{aligned}$$

and so from (4.56) we have that

$$(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}}^{\intercal})^{-1} = W_{\mathcal{T}}^{-1} - \frac{1}{l_w(\mathcal{C})}W_{\mathcal{T}}^{-1}TT^{\intercal}W_{\mathcal{T}}^{-1}, \tag{4.57}$$

where the length of the cycle added by edge e is given by

$$l_w(\mathcal{C}) = \sum_{e\in\mathcal{T}\cap\mathcal{C}} w_e^{-1} + W_{\mathcal{C}}^{-1}.$$

Recall from Lemma 5 that $[T_{\mathcal{T}}T_{\mathcal{T}}^{\intercal}]_{ee}$ is the number of times the edge $e \in \mathcal{T}$ is used in the cycle $\mathcal{C}$. Therefore,

$$\left[W_{\mathcal{T}}^{-1}T_{\mathcal{T}}T_{\mathcal{T}}^{\intercal}W_{\mathcal{T}}^{-1}\right]_{ee} = w_e^{-2}. \tag{4.58}$$

Taking the trace of (4.57) and using (4.58) yields

$$\begin{aligned}&\mathbf{tr}\left[(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}}^{\intercal})^{-1}\right] \\ &= \mathbf{tr}\left[W_{\mathcal{T}}^{-1}\right] + \frac{1}{l_w(\mathcal{C})}\mathbf{tr}\left[W_{\mathcal{T}}^{-1}TT^{\intercal}W_{\mathcal{T}}^{-1}\right] \\ &= \frac{1}{l_w(\mathcal{C})}\left[\sum_{e\in\mathcal{T}\cap\mathcal{C}} w_e^{-2}\right].\end{aligned}$$

For the second part, note that the improvement in $\mathcal{H}_2$ performance after adding edge c is

$$-\frac{\sigma_\omega^2}{2l_w(\mathcal{C})}\left[\sum_{e\in\mathcal{T}\cap\mathcal{C}} w_e^{-2}\right] < 0,$$

since $l_w(\mathcal{C}) > 0$ and $W_e^{-2} > 0$ for all $e \in \mathcal{E}$, and so the $\mathcal{H}_2$ norm always decreases.

In the case of adding k-edge disjoint cycles, each edge can only be in one cycle, and so by Lemma 5 we have that (4.58) still holds. The result follows by applying the preceding argument for each of the k cycles. □

4.3.3 Contributions of Time Scales to $\mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{T}])$

As discussed previously, the two models $\Sigma^{\mathcal{T}}$, $\hat{\Sigma}^{\mathcal{T}}$ differ in the outputs – the former considers only the edge states of the spanning tree, and the latter considers all edges of the graph. We noted in the proof of Proposition 7 that when adding an edge to a tree, the $\mathcal{H}_2$ norm of the dynamics $\hat{\Sigma}^{\mathcal{T}}$ is not affected by the time scales of the nodes on which the edge is placed. In this section, we show that these dynamics , adding an edge to slow nodes provides the smallest increase in $\mathcal{H}_2$ norm.

Recall that the $\mathcal{H}_2$ performance of $\Sigma^{\mathcal{T}}$ can be written as,

$$\begin{aligned}\mathcal{H}_2^2(\Sigma^{\mathcal{T}}) &= \frac{\sigma_\omega^2}{2}\mathbf{tr}[R_{\mathcal{G}}^\intercal(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}}^\intercal)^{-1}R] + \frac{\sigma_v^2}{2}\mathbf{tr}[R_{\mathcal{G}}^\intercal\mathcal{L}^{\mathcal{T}}R_{\mathcal{G}}] \\ &:= \mathcal{H}_2^2(\Sigma^{\mathcal{T}};W) + \mathcal{H}_2^2(\Sigma^{\mathcal{T}};E). \end{aligned} \tag{4.59}$$

Here we have separated the contributions of the weights and time scales into two terms. Consider the second term in (4.59), defined as

$$\mathcal{H}_2^2(\Sigma^{\mathcal{T}};E) := \frac{1}{2}\sigma_v^2\mathbf{tr}[R_{\mathcal{G}}^\intercal\mathcal{L}^{\mathcal{T}}R_{\mathcal{G}}].$$

We have the following proposition.

Proposition 9. *There exists a separation of $\mathcal{H}_2(\Sigma,E)$ into two terms, one depending on the cycle states, and the other the spanning tree states. Namely,*

$$\mathcal{H}_2^2(\Sigma;E) = \frac{\sigma_v^2}{2}\left(\mathbf{tr}[\mathcal{L}^{\mathcal{T}}] + \mathbf{tr}[\mathcal{L}^{\mathcal{G}\setminus\mathcal{T}}]\right).$$

Proof. By trace cyclicity, we can compute:

$$\begin{aligned}&\mathcal{H}_2^2(\Sigma^{\mathcal{T}}, E) = \mathbf{tr}[R_{\mathcal{G}}^{\intercal}\mathcal{L}^{\mathcal{T}}R_{\mathcal{G}}]\\&= \mathbf{tr}\left[\begin{bmatrix} I \\ (T_{\mathcal{T}})^{\intercal}\end{bmatrix}\mathcal{L}^{\mathcal{T}}\begin{bmatrix} I & T_{\mathcal{T}}\end{bmatrix}\right] = \mathbf{tr}[\mathcal{L}^{\mathcal{T}}] + \mathbf{tr}[T_{\mathcal{T}}^{\intercal}\mathcal{L}^{\mathcal{T}}T_{\mathcal{T}}].\end{aligned}$$

Via trace cyclicity, we can manipulate the argument of the second trace term in the last display as follows:

$$\begin{aligned}&\mathbf{tr}[T_{\mathcal{T}}^{\intercal}\mathcal{L}^{\mathcal{T}}T_{\mathcal{T}}] =\\&\mathbf{tr}[D_{\mathcal{T}}(D_{\mathcal{T}}^{\intercal}D_{\mathcal{T}})^{-1}D_{\mathcal{T}}^{\intercal}D_{\mathcal{G}\setminus\mathcal{T}}D_{\mathcal{G}\setminus\mathcal{T}}^{\intercal}D_{\mathcal{T}}(D_{\mathcal{T}}^{\intercal}D_{\mathcal{T}})^{-1}D_{\mathcal{T}}^{\intercal}E_{\mathcal{G}}^{-1}]\end{aligned}$$

Notice that the term $D_{\mathcal{T}}(D_{\mathcal{T}}^{\intercal}D_{\mathcal{T}})^{-1}D_{\mathcal{T}}^{\intercal}$ is the projection operator on the range space of $D_{\mathcal{T}}$. Since the cycles are linear combinations of the spanning tree edges, every column of $D_{\mathcal{G}\setminus\mathcal{T}}$ is in the range of $D_{\mathcal{T}}$. Therefore, the projection of $D_{\mathcal{G}\setminus\mathcal{T}}$ onto the range space of $D_{\mathcal{T}}$ is nothing but $D_{\mathcal{G}\setminus\mathcal{T}}$:

$$D_{\mathcal{T}}(D_{\mathcal{T}}^{\intercal}D_{\mathcal{T}})^{-1}D_{\mathcal{T}}^{\intercal}D_{\mathcal{G}\setminus\mathcal{T}} = D_{\mathcal{G}\setminus\mathcal{T}},$$

and so we can conclude that

$$\begin{aligned}&\mathbf{tr}[T_{\mathcal{T}}^{\intercal}\mathcal{L}^{\mathcal{T}}T_{\mathcal{T}}] = \mathbf{tr}[D_{\mathcal{G}\setminus\mathcal{T}}D_{\mathcal{G}\setminus\mathcal{T}}^{\intercal}D_{\mathcal{T}}(D_{\mathcal{T}}^{\intercal}D_{\mathcal{T}})^{-1}D_{\mathcal{T}}^{\intercal}E_{\mathcal{G}}^{-1}]\\&= \mathbf{tr}[D_{\mathcal{G}\setminus\mathcal{T}}^{\intercal}E_{\mathcal{G}}^{-1}D_{\mathcal{G}\setminus\mathcal{T}}] = \mathbf{tr}[\mathcal{L}^{\mathcal{G}\setminus\mathcal{T}}].\end{aligned}$$

□

From Proposition 9, we can conclude with the following corollary.

Corollary 1. *Consider the dynamics (4.33) of $\Sigma^{\mathcal{T}}$ on a graph $\mathcal{G}$. Then, consider adding an edge ij to $\mathcal{G}$ to get $\mathcal{G}\cup\{ij\}$. The time scale term of the $\mathcal{H}_2^2$ norm in (4.59) satisfies*

$$\mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{G}\cup\{ij\}]; E) = \frac{\sigma_v^2}{2}\left(\epsilon_i^{-1} + \epsilon_j^{-1}\right) + \mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{G}], E).$$

4.3.4 Contribution of Weights to $\mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{T}])^*$

In Section 4.3.2 we showed that adding an edge to a spanning tree always improves the $\mathcal{H}_2$ norm of $\hat{\Sigma}$. In this section, we discuss what happens to $\mathcal{H}_2$ norm of Σ by examining (4.48).

Recall that we can write

$$\begin{aligned}\mathcal{H}_2^2(\Sigma^{\mathcal{T}};W) = \mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}};W)& \\ + \frac{\sigma_w^2}{2}\mathrm{tr}\left[T_{\mathcal{T}}^{\intercal}(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}})^{-1}T_{\mathcal{T}}\right]&.\end{aligned} \tag{4.60}$$

When $\mathcal{G} = \mathcal{T}$, we examined what happens to $\mathcal{H}_2^2(\hat{\Sigma}^{\mathcal{T}};W)$ when an edge is added to $\mathcal{T}$ in Proposition 8. When $\mathcal{G} = \mathcal{T}$, the second term in (4.60) is zero since $T_{\mathcal{T}}$ is the empty matrix. In the next proposition, we show that this term is positive when an edge is added to $\mathcal{T}$.

Proposition 10. *Consider the edge consensus model (4.33) on a graph $\mathcal{T}$ with $\mathcal{H}_2$ norm (4.46). When an edge e with weight $W_{\mathcal{C}}$ is added to $\mathcal{T}$ forming a cycle $\mathcal{C}$, the weight portion of the $\mathcal{H}_2$ norm satisfies*

$$\begin{aligned}\mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{T}\cup\{ij\}];W) = \mathcal{H}_2^2(\Sigma^{\mathcal{T}}[\mathcal{T}];W)& \\ + \frac{\sigma_\omega^2}{2W_{\mathcal{C}}} - \frac{\sigma_\omega^2}{2l_w(\mathcal{C})}\sum_{ij\in\mathcal{C}} w_{ij}^{-2}&.\end{aligned} \tag{4.61}$$

Proof. First, note that the term in (4.60) is zero when there are no cycles in $\mathcal{G}$. Adding a single edge to the tree $\mathcal{T}$ forms a cycle, and hence $T_{\mathcal{T}}$ has one column. We can compute using the rank-one update in (4.56),

$$\begin{aligned}&\frac{\sigma_\omega^2}{2}\mathbf{tr}\left[T_{\mathcal{T}}^{\intercal}(R_{\mathcal{G}}W_{\mathcal{G}}R_{\mathcal{G}})^{-1}T_{\mathcal{T}}\right]\\ &= \frac{\sigma_\omega^2}{2}\mathbf{tr}\left[T_{\mathcal{T}}^{\intercal}\left(W_{\mathcal{T}}^{-1} - \frac{1}{l_W(\mathcal{C})}W_{\mathcal{T}}^{-1}T_{\mathcal{T}}T_{\mathcal{T}}^{\intercal}W_{\mathcal{T}}^{-1}\right)T_{\mathcal{T}}\right]\\ &= \frac{\sigma_\omega^2}{2}\left(\mathbf{tr}\left[T_{\mathcal{T}}^{\intercal}W_{\mathcal{T}}^{-1}T_{\mathcal{T}}\right] - \frac{1}{l_w(\mathcal{C})}\mathbf{tr}\left[(T_{\mathcal{T}}^{\intercal}W_{\mathcal{T}}^{-1}T_{\mathcal{T}})^2\right]\right)\\ &= \frac{\sigma_\omega}{2}\left(l_w(\mathcal{C}) - 1 - \frac{(l_w(\mathcal{C})-1)^2}{l_w(\mathcal{C})}\right)\\ &= \frac{\sigma_\omega}{2}\left(\frac{l_w(\mathcal{C})W_{\mathcal{C}}^{-1} - W_{\mathcal{C}}^{-2}}{l_w(\mathcal{C})}\right).\end{aligned} \tag{4.62}$$

Summing (4.62) with the improvement of $\mathcal{H}_2^2(\hat{\Sigma}; W)$ in (4.55) from Proposition 8 yields (4.61).

□

*4.3.5 Example: Minimum $\mathcal{H}_2$-Norm Spanning Trees**

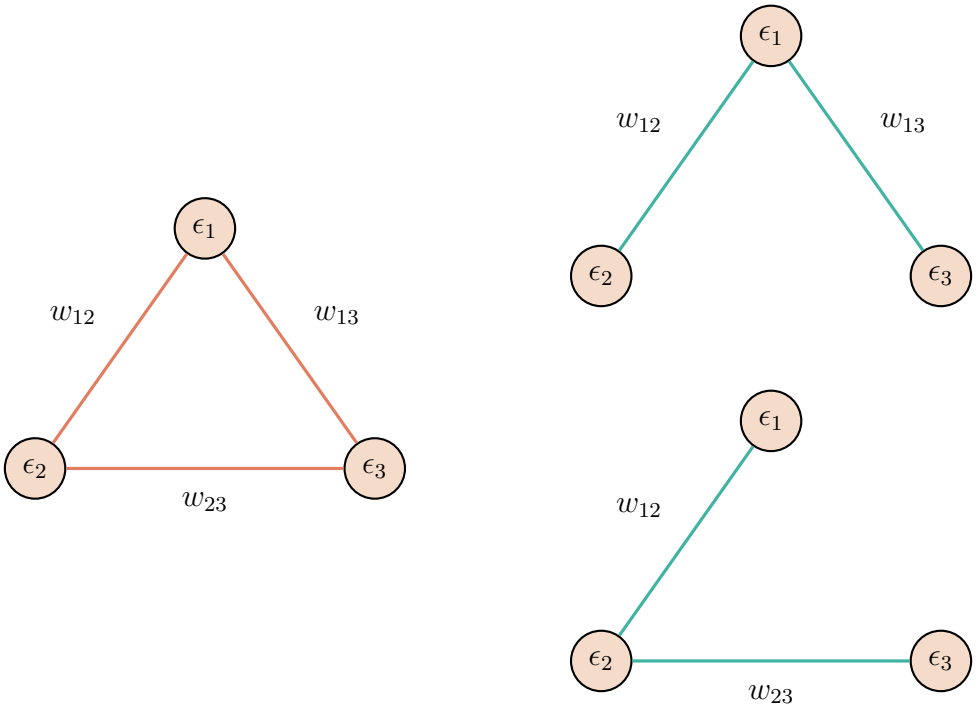


Figure 4.12: Example of a time scaled, weighted graph with distinct edge weights and time scales, but no unique minimal $\mathcal{H}_2$ spanning tree.

First, we show an example of a graph with distinct time scales on the nodes, and distinct edge weights, but no unique minimum-$\mathcal{H}_2$ spanning tree. Consider the triangle in Figure 4.12 with the edge consensus model $\hat{\Sigma}^{\mathcal{T}}$, and the following time scales and edge weights:

$$\epsilon_1 = 1,\ w_{23} = 1$$

$$\epsilon_2 = 2,\ w_{13} = 2$$

$$\epsilon_3 = 3,\ w_{12} = 3.$$

Further suppose that the noise weightings satisfy $\sigma_\omega = \sigma_v = 1$. Then, the spanning tree $\mathcal{T}_1$ of the triangle in Figure 4.12 on the top right has $\mathcal{H}_2$ norm

$$\mathcal{H}_2(\Sigma^{\mathcal{T}_1}(\mathcal{G})) = \frac{1}{2}\left[w_{12}^{-1} + w_{13}^{-1} + \epsilon_1^{-1} + \sum_{i=1}^{3}\epsilon_i^2\right] = 1.8\bar{3}$$

and the spanning tree $\mathcal{T}_2$ in Figure 4.12 on the bottom right has $\mathcal{H}_2$ norm

$$\mathcal{H}_2(\Sigma^{\mathcal{T}_1}(\mathcal{G})) = \frac{1}{2}\left[w_{12}^{-1} + w_{23}^{-1} + \epsilon_3^{-1} + \sum_{i=1}^{3}\epsilon_i^2\right] = 1.8\bar{3}.$$

Remark 11. *The reason the minimum-$\mathcal{H}_2$ spanning trees in Figure 4.12 are not unique, despite having distinct edge weights and time scales in $\mathcal{G}$, is because the edge weights of the auxilary graph $\mathcal{G}'$ defined in the proof of Proposition 7, given by*

$$w'_{ij} = \sigma_v^2(\epsilon_i^{-1} + \epsilon_j^{-1}) + \sigma_\omega^2 w_{ij}^{-1},$$

are not distinct in $\mathcal{G}'$.

*4.3.6 Example: Adding Edges to a Tree Graph**

Consider the path graph in Figure 4.13 denoted by the solid edges, where all the edges in the path have weight $w_e = 1$. Suppose $W_1 = 10$ and $W_2 = 5$, and $\sigma_\omega = 1$ Then, the improvement of adding either edge W_1 to $\mathcal{G}$ to create cycle $\mathcal{C}_1$, or adding W_2 to $\mathcal{G}$ to create cycle $\mathcal{C}_2$ is

$$-\frac{1}{2l_w(\mathcal{C}_1)}\left[\sum_{e\in\mathcal{T}\cap\mathcal{C}_1} w_e^{-2}\right] \approx -0.9524$$
$$-\frac{1}{2l_w(\mathcal{C}_2)}\left[\sum_{e\in\mathcal{T}\cap\mathcal{C}_2} w_e^{-2}\right] \approx -0.9375,$$

and so the shorter (but more highly weighted) cycle $\mathcal{C}_1$ offers the better improvement.

4.4 Performance of Single Influenced Consensus[5]

A common scheme to exert control over a networked system is through partitioning the node set into leader(s) and followers. The leader node(s) are then directly controlled or influenced

[5]Text in this section adapted from "Influenced Consensus for Multi-Scale Networks," presented at the 2019 American Control Conference, Philadelphia, PA, USA [36].

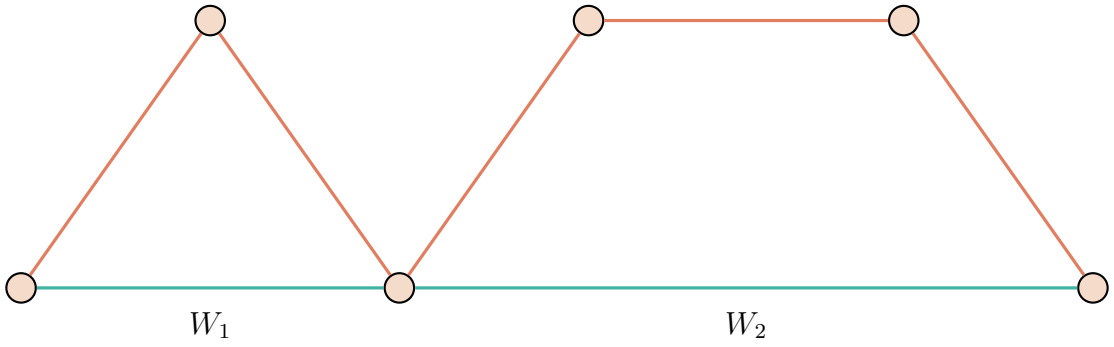


Figure 4.13: Example of a weighted graph where a cycle with fewer edges has a lower $\mathcal{H}_2$ norm for $\hat{\Sigma}^{\mathcal{T}}$ than a cycle with more edges.

as if a directly controlled node was attached to the leader node. These two formulations are equivalent, but here we will adopt the formulation of *single input influenced consensus* for which an influencing node is attached to a native node in the graph, $v_i \in \mathcal{V}$ [15]. This influencing node injects a control signal $u(t) \in \mathbb{R}$ to v_i which is then propagated over the rest of the network via the consensus protocol. This can be written in terms of a continuous, time-invariant linear system by defining system and input matrices, $A(\mathcal{G}, i) = -\left(L(\mathcal{G}) + e_i e_i^T\right)$ and $B(i) = e_i$, respectively. The matrix $-A(\mathcal{G}, i)$ is called the *Dirichlet matrix* or the *grounded Laplacian*, where $\mathcal{G}$ denotes the graph topology and i corresponds to the influenced node $v_i \in \mathcal{V}$. Just as the scaled consensus problem was defined as the mono-scale consensus problem over agent-based time scaling, we extend the mono-scale influenced consensus to a multi-scale case defined as,

$$E\dot{x}(t) = A(\mathcal{G}, i)x(t) + B(i)u(t). \tag{4.63}$$

From this formulation we can define the scaled, grounded Laplacian as,

$$\begin{aligned}\tilde{A}(\mathcal{G}, i) &= -E^{-1}\left(L(\mathcal{G}) + e_i e_i^T\right) \\ &= -\left(\tilde{L}(\mathcal{G}) + E^{-1} e_i e_i^T\right),\end{aligned}$$

as well as $\tilde{B}(i) = E^{-1}e_i$, where we adopt the convention of denoting scaled quantities with a tilde, and mono-scale quantities without. With the scaled, influenced model now defined, let us consider some necessary and useful properties of the scaled Laplacian/scaled consensus problem.

Properties of the Scaled Grounded Laplacian

An important and often exploited characteristic of a mono-scale Laplacian matrix is the well-defined eigenvalue structure [75]. Here, we will show that despite not being PSD, the scaled Laplacian retains some of these advantageous characteristics. We can note that the scaled Laplacian's definition is akin to the system matrices used in random walks for discrete systems or a normalized Laplacian. Neither of these comparisons are exact, however, due to the lack of assumptions on the values of the scaling parameters. However, we can take inspiration from these formulations to note that the scaled Laplacian is similar to a symmetric matrix, which we formalize in the following proposition.

Proposition 11. *The scaled Laplacian, $E^{-1}L(\mathcal{G})$ is similar to a symmetric matrix,*

$$L' = E^{-1/2}L(\mathcal{G})\,E^{-1/2}, \tag{4.64}$$

which will be subsequently referred to as the auxiliary matrix for the scaled Laplacian.

Proof. Define a transformation matrix $T = E^{1/2}$, then observe,

$$\begin{aligned} T\tilde{L}(\mathcal{G})\,T^{-1} &= E^{\frac{1}{2}}\left(E^{-1}L(\mathcal{G})\right)E^{-\frac{1}{2}} \\ &= E^{-\frac{1}{2}}L(\mathcal{G})\,E^{-\frac{1}{2}}. \end{aligned}$$

The existence of $E^{\pm 1/2}$ is ensured by a diagonal E with positive entries [47, Theorem 7.2.6]. □

This allows us to state an important characteristic of the auxiliary matrix.

Proposition 12. *The auxiliary matrix, $E^{-1/2}LE^{-1/2}$, is PSD.*

Proof. The Laplacian matrix, L, is PSD [75]. From [47, Observation 7.1.6], for PSD matrix L, Q^TLQ is also PSD; now let $Q = E^{-1/2}$. □

The auxiliary matrix being symmetric and PSD ensures that all of its eigenvalues are real and nonnegative. By (4.64), this holds for $\tilde{L}(\mathcal{G})$ thus it retains a key characteristic of the mono-scaled Laplacian despite not being PSD nor symmetric. Further, the results of Propositions 11 and 12 can be applied to the scaled, grounded Laplacian, $\tilde{A}(\mathcal{G}, i)$.

Next, we note that under Assumption 1 (that the underlying graph is connected), we can state definitively that the grounded Laplacian is invertible.

Proposition 13. *When $\mathcal{G}$ is connected, $\tilde{A}(\mathcal{G}, i)$ is an invertible matrix.*

Proof. For mono-scale networks, $A(\mathcal{G}, i)$ is negative definite (and thus invertible) so long as $\mathcal{G}$ is connected [19, Proposition 3]. Then, the inverse for $\tilde{A}(\mathcal{G}, i)$ is given by right-multiplying $A(\mathcal{G}, i)^{-1}$ by E. □

The invertibility of $\tilde{A}(\mathcal{G}, i)$ allows us to recover another useful property of the grounded Laplacian, $-\tilde{A}(\mathcal{G}, i)^{-1}\tilde{B}(i) = \mathbf{1}$, which is necessary to find the convergence cost for a constant influence signal presented in the next section. A final proposition relates the ordered eigenvalues of the scaled Laplacian with the unscaled Laplacian spectra.

Proposition 14. *For eigenvalues in increasing order and $k \leq n$, $\epsilon_{\max}\lambda_k(\tilde{L}(\mathcal{G})) \leq \lambda_k(L(\mathcal{G}))$. Furthermore, for $k = 1$, this relation can be refined to $\epsilon_{\min}\lambda_1(\tilde{L}(\mathcal{G})) \leq \lambda_1(L(\mathcal{G}))$.*

Proof. The first relation follows directly from the singular value/eigenvalue inequalities in [28, Theorem 2]. Since the eigenvalues of $L(\mathcal{G})$ and $\tilde{L}(\mathcal{G})$ are real and non-negative, the inequalities generalize to the eigenvalues, that is, for $j, k \geq 1$, $j + k \leq n + 1$,

$$\lambda_{j+k-1}(E^{-1}L(\mathcal{G})) \leq \lambda_j(E^{-1})\lambda_k(L(\mathcal{G})). \tag{4.65}$$

Now, take $j = 1$, and note that $\lambda_1(E^{-1}) = 1/\epsilon_{\max}$. Substituting these into (4.65) gives the proposed relation,

$$\epsilon_{\max}\lambda_k\left(E^{-1}L(\mathcal{G})\right) \leq \lambda_k(L(\mathcal{G})),$$

for $k \leq n$. Next, let $j = n$ and $k = 1$ in (4.65) giving,

$$\lambda_n\left(E^{-1}L(\mathcal{G})\right) \leq \frac{1}{\epsilon_{\min}}\lambda_1(L(\mathcal{G})).$$

Noting that $\lambda_1\left(E^{-1}L(\mathcal{G})\right) \leq \lambda_n\left(E^{-1}L(\mathcal{G})\right)$ gives the second relation. □

The eigenvalue bounds presented in Proposition 14 also hold for $-A(\mathcal{G}, i)$ and $-\tilde{A}(\mathcal{G}, i)$, which allows for the performance of scaled influenced networks to be related back to properties of mono-scale influence networks.

4.4.1 Constant Input Influence

The combined results from Section 4.4 have the consequence that under Assumption 1, there exists a consensus subspace spanned by $\mathbf{1}$ for the scaled, influenced system (4.63). For a constant input signal, $u(t) = u_c$, all agents will converge to the input value, though all nodes will now be responding to this input/change in the adjacent nodes on their distinct time scale. The input of a constant signal is referred to as *anchor* influence. The corresponding state cost necessary to steer all agents to this value from an arbitrary initial condition can be found by first defining a relative state vector $\tilde{x} = x(t) - u_c\mathbf{1}$, then considering the following integral over an infinite time horizon,

$$\tilde{J}(\mathcal{G}, i, \tilde{x}(0)) = 2\int_0^\infty \tilde{x}^T\tilde{x}\ \mathrm{d}t,$$

where the leading factor of 2 is an arbitrary scaling included to ensure that the coefficient of the scaled state cost is unity. It then follows from (4.63) and Proposition 13 that,

$$\tilde{J}(\mathcal{G}, i, \tilde{x}(0)) = -\tilde{x}(0)(\tilde{A}(G, i))^{-1}\tilde{x}(0). \tag{4.66}$$

As in the mono-scale case, the inverse of the scaled, grounded Laplacian is an important indicator of how the scaled consensus network responds to anchor influence. In [15], three

metrics based on the cost $\tilde{J}$ are defined; a minimum, maximum, and average performance cost. We reproduce them here for completeness and to put in terms of our notation,

$$\begin{aligned}
\tilde{J}_{\min}(\mathcal{G}, i) &= \inf_{\|\tilde{x}(0)\|=1} \tilde{J}(\mathcal{G}, i, \tilde{x}(0) \\
&= \lambda_1(-\tilde{A}(\mathcal{G}, i)^{-1}) = 1/\lambda_n(-\tilde{A}(\mathcal{G}, i)) \qquad (4.67) \\
\tilde{J}_{\max}(\mathcal{G}, i) &= \sup_{\|\tilde{x}(0)\|=1} \tilde{J}(\mathcal{G}, i, \tilde{x}(0) \\
&= \lambda_n(-\tilde{A}(\mathcal{G}, i)^{-1}) = 1/\lambda_1(-\tilde{A}(\mathcal{G}, i)) \qquad (4.68) \\
\tilde{J}_{\text{avg}}(\mathcal{G}, i) &= \mathbf{E}_{\|\tilde{x}(0)\|=1} \tilde{J}(\mathcal{G}, i, \tilde{x}(0) \\
&= \frac{1}{n}\mathbf{tr}\left(-\tilde{A}(\mathcal{G}, i)^{-1}\right) = \frac{1}{n}\sum_{i=1}^{n} 1/\lambda_i(-\tilde{A}(\mathcal{G}, i)), \qquad (4.69)
\end{aligned}$$

where **tr** denotes the trace operator.

Just as in [15], we can show that the state cost (4.66) is bounded by the maximum performance cost of a path graph from above, and the minimum performance cost of a complete graph from below. However, we can also observe that lower and upper bounds must be adjusted to account for the fastest and slowest nodes present in the network, respectively.

Proposition 15. *For the n-node graph, $\mathcal{G}$, with minimum agent scaling of $\epsilon_{\min}$, the minimum and maximum performance costs (4.67), (4.68) of attaching to node $v_i \in \mathcal{V}$ is bounded as,*

$$\inf_{(\mathcal{G},i)} \tilde{J}_{\min}(\mathcal{G}, i) = 2\epsilon_{\min}(1 + n + \sqrt{n^2 + 2n - 3})^{-1} \qquad (4.70)$$

and

$$\sup_{(\mathcal{G},i)} \tilde{J}_{\max}(\mathcal{G}, i) = \frac{\epsilon_{\max}}{2}\left(1 + \cos\frac{2\pi n}{2n+1}\right)^{-1}. \qquad (4.71)$$

Proof. First, let us consider the upper bound. We start by considering a graph $\mathcal{G}_s$ with the same topology as $\mathcal{G}$, but where each node is scaled by an identical scaling parameter, ϵ_s. Furthermore, suppose we select ϵ_s such that,

$$\lambda_1(-\tilde{A}(\mathcal{G}, i)) \geq \lambda_1(-A(\mathcal{G}_s, i)) = \epsilon_s\lambda_1(-A(\mathcal{G}, i)). \qquad (4.72)$$

Combining (4.68) and (4.72) gives

$$\tilde{J}_{\max}(\mathcal{G}, i) = \frac{1}{\lambda_1(-\tilde{A}(\mathcal{G}, i)} \leq \frac{1}{\epsilon_s \lambda_1(-A(\mathcal{G}, i)\,)}. \tag{4.73}$$

In [15, Proposition 2.2] it was established that the minimum eigenvalue of $A(\mathcal{G}, i)$ was bounded from below by the minimum eigenvalue of a n-node path graph influenced from the leading node, that is $\lambda_1(-A(\mathcal{P}, 1))) \leq \lambda_1(A(\mathcal{G}, i)\,)$. Applying this to (4.73), provides a bound that is dependent on the value of ϵ_s,

$$\begin{aligned}\tilde{J}_{\max}(\mathcal{G}, i) &\leq \frac{1}{\epsilon_s \lambda_1(-A(\mathcal{G}, i)\,)} \\ &\leq \frac{1}{\epsilon_s \lambda_1(-A(\mathcal{P}, 1))} = \frac{1}{\epsilon_s} J_{\max}(\mathcal{P}, 1).\end{aligned} \tag{4.74}$$

To find the necessary value for ϵ_s that satisfies our original supposition (4.72), we can consider Proposition 14. For the arbitrarily scaled, grounded Laplacian,

$$\lambda_1(-\tilde{A}(\mathcal{G}, i)\,) \leq \frac{1}{\epsilon_{\min}} \lambda_1(-A(\mathcal{G}, i)\,).$$

Imposing the requirement from (4.72) on the above equation yields

$$\epsilon_s \lambda_1(A(\mathcal{G}, i)\,) \leq \frac{1}{\epsilon_{\min}} \lambda_1(A(\mathcal{G}, i)\,),$$

which is satisfied when $\epsilon_s \leq 1/\epsilon_{\min}$. By considering the total distribution of time scale parameters present we can note, $1/\epsilon_{\min} \geq 1/\epsilon_{\max}$, so $\epsilon_s = 1/\epsilon_{\max}$ satisfies the necessary relation with the smallest ϵ_s. Taking this value of ϵ_s along with the closed form solution for $J_{\max}(\mathcal{P}, 1)$ derived in [15, Proposition A.5], leads to the relation presented in (4.71).

Now let us consider the lower bound. Define a complement graph of $\mathcal{G}$, denoted by $\mathcal{G}_c$, such that $\tilde{L}(\mathcal{K}) = \tilde{L}(\mathcal{G}) + \tilde{L}(\mathcal{G}_c)$. Here, $\mathcal{K}$ is a scaled, complete graph that has $\mathcal{G}$ embedded within it. From the definition of the scaled, grounded Laplacian,

$$\tilde{L}(\mathcal{G}) + \tilde{L}(\mathcal{G}_c) + E^{-1} e_i e_i^T = \tilde{L}(\mathcal{K}) + E^{-1} e_i e_i^T,$$

it follows that,

$$\lambda_n \left(\tilde{L}(\mathcal{G}) + E^{-1} e_i e_i^T \right) \leq \lambda_n \left(\tilde{L}(\mathcal{K}) + E^{-1} e_i e_i^T \right). \tag{4.75}$$

Additionally, from Proposition 14 we can relate the minimum performance cost for the scaled, complete graph and the mono-scale graph,

$$\frac{1}{\epsilon_{\max}\lambda_n(\tilde{L}(\mathcal{K}) + E^{-1}e_i e_i^T)} \geq \frac{1}{\lambda_n(L(\mathcal{K}) + e_i e_i^T)},$$
$$\tilde{J}_{\min}(\mathcal{K}, i) \geq \epsilon_{\max} J_{\min}(\mathcal{K}, i) \geq \epsilon_{\min} J_{\min}(\mathcal{K}, i). \tag{4.76}$$

Combining (4.75) and (4.76) with (4.67) gives,

$$\begin{aligned}\epsilon_{\min} J_{\min}(\mathcal{K}, i) &\leq \frac{1}{\lambda_n(-(\tilde{L}(\mathcal{K})) + E^{-1}e_i e_i^T)} \\ &\leq \frac{1}{\lambda_n(-\tilde{L}(\mathcal{G}) + E^{-1}e_i e_i^T)} = \tilde{J}(\mathcal{G}, i),\end{aligned}$$

which establishes the desired bound, (4.70), when combined with the closed form solution of $J_{\min}(\mathcal{K}, i))$ derived in [15, Proposition A.1]. □

We can also find a closed form solution for the average performance cost for scaled tree graphs, which provides some insights into how the scaling parameters affect the notion of centrality for a given node, and how the distribution of scaling parameters over the nodes impacts the average performance.

Proposition 16. *The average performance cost for constant influence through node $v_i \in \mathcal{V}(\mathcal{T})$ where $\mathcal{T}$ is an n-node tree graph is*

$$\tilde{J}_{avg}(\mathcal{T}, i) = \frac{1}{n}\sum_{j=1}^{n} \epsilon_j (d(v_i, v_j) + 1), \tag{4.77}$$

where $d(v_i, v_j)$ denotes the shortest path length between nodes v_i and v_j, which for a tree graph is simply the number of edges separating the nodes.

Proof. From Proposition 13, we have $-\tilde{A}(\mathcal{G}, i)^{-1} = -A(\mathcal{G}, i)^{-1}E$, and Lemma 2.3 from [15] provides the closed form for $-A(\mathcal{G}, i)^{-1}$. Combining these results, as well as making the assumption that the nodes are numbered such that the influencing node attaches to node $v_1 \in \mathcal{V}(\mathcal{T})$, gives,

$$-\tilde{A}(\mathcal{G}, i)^{-1} = \begin{bmatrix} \epsilon_1 & \mathbf{1}^T E_{11} \\ \epsilon_1 \mathbf{1} & \tilde{A}_{11}^{-1} - \mathbf{1}\mathbf{1}^T E_{11} \end{bmatrix}, \tag{4.78}$$

where A_{11} and E_{11} are the principle submatrices (formed by deleting the first row and column) of $-A(\mathcal{G}, i)$ and E, respectively. From here, the proof follows exactly from the proof of Lemma 2.3 in [15]. For completeness, we summarize the relevant ideas here: The influenced node, v_1 can either have a single neighbor or multiple neighbors. If it has a single neighbor, the principle submatrix of $\tilde{A}(\mathcal{G}, i)$ is given by the grounded Laplacian of the subgraph of $\mathcal{G}$ formed by removing v_1 and considering the neighbor node as the new influenced node. When v_1 has multiple neighbors, the principle submatrix can be represented as a block diagonal matrix of the sub-trees headed by each neighbor node. These two cases can be applied for each link in the tree, each time reducing the dimension of the principle submatrices by one. This, along with the first entry of (4.78) establishes that the diagonal of $\tilde{A}(\mathcal{G}, i)^{-1}$ represents the minimum distances between v_1 and all other nodes weighted by the scaling parameter of the intermediate nodes plus the scaling parameter of the influenced node. Generalizing (4.78) to any influenced node completes the proof and gives (4.77). □

This result illustrates that the average performance cost given a constant input will be determined both by the topology of the tree-graph (which determines the shortest path between nodes), as well as the distribution of scaling parameters over the nodes. The following example highlights how this can diverge from results for mono-scale networks.

Constant Input Example

In Figure 4.14, we show an example tree graph on six nodes. For the mono-scale influenced consensus problem, the node with the minimum average state cost is node 3 with an average state cost of $J_{\text{avg}}(\mathcal{T}, 3) = 2$ [15, Lemma 2.3]. However, we can observe that any node can be made to have the lowest average cost by introducing time scaling.

Nodes 4, 5, or 6: If nodes 1-3 are taken to have a scaling parameter equal to 0.5, then scaling any single remaining node (4, 5, or 6) to be significantly slower can result in the slow node having a lower average performance cost, despite having longer paths to all other nodes. For example, for $\epsilon_4 = 3.0$, with $\epsilon_{1,2,3,5,6} = 0.5$, $\tilde{J}_{\text{avg}}(\mathcal{T}, 3) = 1.83$ but $\tilde{J}_{\text{avg}}(\mathcal{T}, 4) = 1.75$. Similar

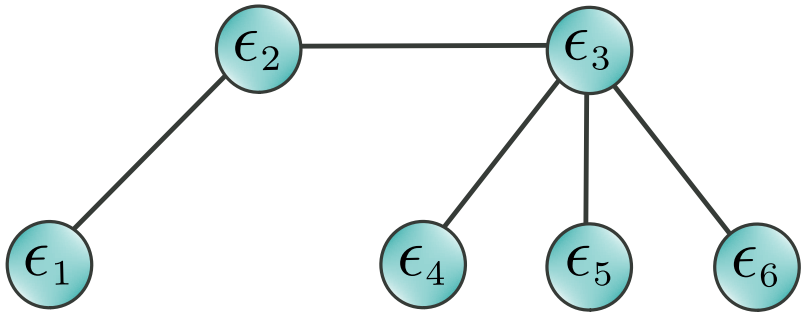


Figure 4.14: Tree graph on six nodes, $\mathcal{T}$. Each node has an associated scaling parameter, ϵ_i, with the subscript denoting the node number. Node 3 is the most "central" node, as it has the shortest possible distance to all other nodes.

responses can be seen when adjusting the scaling of nodes 5 or 6.

Nodes 1 or 2: In the previous case, the scaled node with lowest average cost still had short path lengths to the remaining nodes. The following distribution of scaling parameters grants the lowest state cost to the least central node (1): for $\epsilon_1 = 2.5$, $\epsilon_{2,3} = 0.5$, and $\epsilon_{4,5,6} = 0.2$ we find $\tilde{J}_{\text{avg}}(\mathcal{T}, 1) = 1.23$, $\tilde{J}_{\text{avg}}(\mathcal{T}, 2) = 1.38$, $\tilde{J}_{\text{avg}}(\mathcal{T}, 3) = 1.70$, and $\tilde{J}_{\text{avg}}(\mathcal{T}, [4, 5, 6]) = 2.32$. Similar results can be found for node 2.

4.4.2 Variable Input Influence

In the previous section, we evaluated the network response to a constant influence signal. In practice, however, a network may be exposed to an arbitrary input signal. To investigate the scaled network's response to varying input, we can utilize the controllability gramian for the scaled system (4.63) [46], which is defined in the integral form (for stable $\tilde{A}$),

$$\tilde{P}(\mathcal{G}, i) = \int_0^\infty e^{\tilde{A}(\mathcal{G},i)\tau} \tilde{B}(i)\tilde{B}(i)^T e^{\tilde{A}(\mathcal{G},i)^T\tau} \, \mathrm{d}\tau.$$

It is also the unique solution to the continuous algebraic Lyapunov equation,

$$\tilde{A}(\mathcal{G}, i)\, \tilde{P}(\mathcal{G}, i) + \tilde{P}(\mathcal{G}, i)\tilde{A}(\mathcal{G}, i)^T = -\tilde{B}(i)\tilde{B}(i)^T. \tag{4.79}$$

Beyond providing a binary measure of state controllability, the properties of the gramian, such as its minimum/maximum eigenvalues, determinant, and trace of its inverse have been utilized in the literature for selection of control nodes, network optimization, as well as quantification of network fragility [89, 88]. Here, however, we do not utilize the gramian as a measure of controllability; instead we use the fact that $\mathcal{H}_2^2(\tilde{A}(\mathcal{G},i)\,,\tilde{B}(i)) = \mathbf{tr}(\tilde{P}(\mathcal{G},i))$ (under the assumption of full state output [110, 4.10.1]). The $\mathcal{H}_2$-norm is a measure of the network's energy response to an influencing signal, as well as being related to the effective resistance of a network [75]. In [15, Lemma 3.1], it was shown that the $\mathcal{H}_2$-norm for all connected graphs is independent of the choice of influencing node. Due to the asymmetry of the scaled Laplacian, a closed form solution for the controllability gramian is difficult to derive for arbitrary graph topologies. However, we can bound the minimum $\mathcal{H}_2$ performance of the scaled network in the following proposition.

Proposition 17. *For a connected graph $\mathcal{G}$ with influenced node v_i,*

$$\mathbf{tr}(\tilde{P}(\mathcal{G},i)) \geq \frac{1}{\left(2\epsilon_i^2\left(\frac{1}{\epsilon_i}+\sum_{j=1}^{n}\frac{deg(v_j)}{\epsilon_j}\right)\right)}, \tag{4.80}$$

where $deg(v_j)$ represents the degree of node v_j.

Proof. From the continuous, algebraic Lyapunov equation (4.79), [90] determines the following bound on the trace of the controllability gramian,

$$\mathbf{tr}(\tilde{P}(\mathcal{G},i)) \geq -\frac{\mathbf{tr}(\tilde{B}(i)\tilde{B}(i)^T)}{2\mathbf{tr}(\tilde{A}(\mathcal{G},i))}. \tag{4.81}$$

The trace of $\tilde{B}(i)\tilde{B}(i)^T = E^{-1}e_ie_i^TE^{-1} = 1/\epsilon_i^2$ and the trace of $\tilde{A}(\mathcal{G},i) = -(\tilde{L}(\mathcal{G}) + E^{-1}e_ie_i^T)$ is the sum of the scaled degrees of the nodes in $\mathcal{G}$ plus $1/\epsilon_i$ to account for the grounding. Substituting these expressions into (4.81) gives the intended result. □

We can see from this bound that the performance will be a function of the scaling parameters, and the choice of influenced node can impact the relative performance of the network. In fact, the bound in (4.80) shows that the relative lower-bound of the performance from

influencing faster nodes is always greater than the case where a slower node had been chosen. This suggests that, given a choice of influenced node, the fastest node available should be chosen to maximize the $\mathcal{H}_2$-norm performance. However, there is no guarantee that the performance for a scaled graph exceeds that of a mono-scale graph of the same topology. In the following example, we highlight this dependence and lack of performance guarantee.

Varying Input Example

Consider a graph with topology given by Figure 4.15, under the case where only the influenced node is scaled to a fast time scale. Let the scaled, influenced node be v_1. If $\epsilon_1 \lesssim 0.19$, then $\mathbf{tr}(\tilde{P}(\mathcal{G}, 1)) \geq 0.5$. This shows that the existence of scaled nodes can allow for greater $\mathcal{H}_2$-norm performance than mono-scale networks (for which $\mathbf{tr}(P(\mathcal{G}, i)) = 1/2$ for any choice of influenced node [15]), but that improved performance over the mono-scale network is not guaranteed for all distributions of scaling parameters.

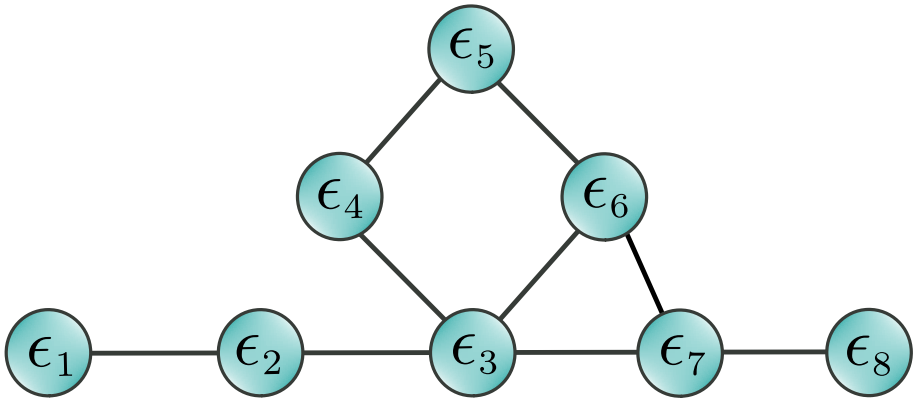


Figure 4.15: Example graph topology.

Now, retain the topology in Figure 4.15 but let $\epsilon_{1,7} = 0.1$, $\epsilon_{2,5,8} = 1.0$, $\epsilon_3 = 0.4$, $\epsilon_4 = 0.2$, and $\epsilon_6 = 0.6$. Influencing either node 1 or 7, for which $\mathbf{tr}(\tilde{P}(\mathcal{G}, [1, 7])) \simeq 0.62$, guarantees higher performance than the mono-scale network on the same topology. This suggests that either of these nodes are good candidates for a point of influence, however, the lower bound

in (4.80) does not allow us to further differentiate between their actual performance. Node 4 has the next best performance bound, $\mathbf{tr}(\tilde{P}(\mathcal{G}, 4)) \simeq 0.17$, which we can see does not provide the performance guarantees of nodes 1 and 7. Thus, the lower bound given in Proposition 17 can offer insights into candidate nodes for influence, but does not differentiate between the performance for identically scaled nodes.

4.5 Chapter Summary

In this chapter we have presented the prior work that the proposed work presented in the next chapter will be built upon. We've observed that reduced order modeling techniques can be applied to complex dynamics operating over a network, but there remains the question of the graph-theoretic interpretation of those reduced order models. We've also seen that the addition of agent-based time scales fundamentally changes some of the notions of performance for consensus networks, but a graph-theoretic interpretation of such systems is still possible, even without resorting to reduced order modeling techniques. However, we've also seen some of the limitations and hardships introduced by the inclusion of time scales; much of the results that leverage the symmetry and special spectral qualities of the graph Laplacian must be hard won yet again. In the next chapter, we will seek to further explore multi-time scale consensus problems, while also seeking to bring the tools from singular perturbation into play to assist in performance analysis.

Chapter 5

CLOCK BIASED CONSENSUS ERROR MITIGATION AND CLOCK SYNCHRONIZATION

In the previous chapters, we have observed that clock-based measurement and implementation errors for receptive agents with inertial navigation (RAIN) can lead to a failure to achieve consensus in the double integrator case. This is analogous to the well-known initialization errors and integration drift in inertial navigation systems, where small errors in the initial state of a system or errors are integrated forward and grow with time [10], though distinct in the sense that while the velocity states do not form consensus, neither do they diverge (which is the case for long-duration noise errors in inertial systems).

A natural remedy to this phenomenon is to periodically update the inertial/dead reckoned states with updated state measurements, which resets the integration errors to within the sensor limits. This is a realistic addition, as well, as accurate sensors may have a non-negligible sensing time, between which the system evolves with inertial measurements. In the next section, Section 5.1, we investigate via simulation the effect that this has on the double integrator RAIN case that we identified previously.

In Section 5.2, we also approach the problem by proposing a method of continuous clock synchronization which utilizes consensus-based inputs to virtual clock parameters. This allows for an effective clock synchronization without knowledge of the local clock parameters. We consider both leaderless and leader-based implementations, the latter of which allows for synchronization to a selected agent's clock. This is useful in scenarios where one agent is known to have an accurate clock (i.e. a heterogeneous multi-agent system with a mix of instrumentation and sensors).

5.1 Periodic Measurement Correction

It is a fair question of why, when the RAIN case failed to achieve consensus on either true or inertially tracked states, we would expect periodically updating to have any positive effect? The answer of course, is that we cannot expect periodic updates to induce consensus in the RAIN system. However, inspecting Figures 3.5 and 3.6 do indicate that the inertial velocities do tend to cluster together, apparently *near* some consensus value. If successive state updates can iteratively reduce the spread in the tracked states, then potentially the error in the true states can also be reduced. Before proceeding, however, let us be explicit about one detail about the following simulations.

Assumption 2. *Denote the cadence of accurate state updates as δ. We assume that δ is the same for all agents regardless of agent clock.*

The simplifying part of Assumption 2 is that all of the updates happen simultaneously at the same cadence. This is reasonable if the update is taken to be true state information from some centralized source that is on it's own clock, but less so if the update is from an "on-board" instrument that may be triggered each $\tau_i(k\delta)$. However, even in this case, the result would be some agents begin their update before others, which should not drastically change the behavior described here. However, relaxing this assumption would be a goal of future work.

In Figure 5.1 we see a simulation of a RAIN network periodically receiving accurate state measurements. What we can observe is that it appears that there is a minimum spread between the velocity states that the periodic updates can assist in getting to, after it has been achieved no further improvement is achieved. Moreover, we can observe that this behavior appears to be stable over many updates, which can be seen in Figure 5.2.

Finally, in Figure 5.3, we see that we can drive the system extremely divergent (even under the periodic updates), by allowing very fast clocks. This is to be expected as the implementation errors from fast clocks along with the scaled inertial dynamics should result in large errors. However, again, we can see that the clock biased internal estimate dynamics

are ultimately to blame in this scenario as well, simulations with no true implementation error also exhibit this phenomenon - as can be seen in Figure 5.4.

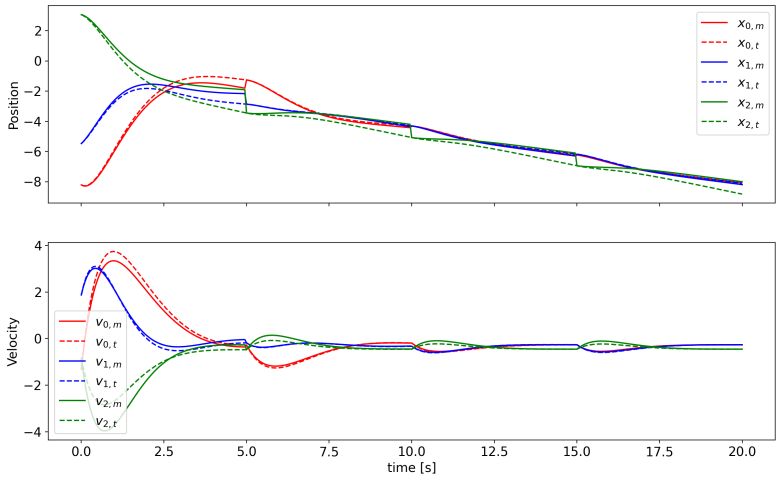


Figure 5.1: Simulation of periodic measurement update for graph on 3 nodes with $\delta = 5$. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed. Clock rates are randomly generated in the range $[0.5, 1.5]$

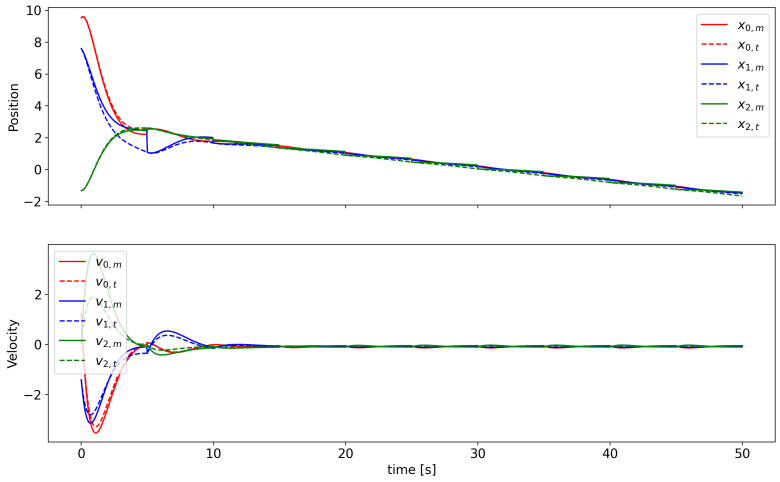


Figure 5.2: Simulation of periodic measurement update for graph on 3 nodes with $\delta = 5$ over a duration of 10δ. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed. Clock rates are randomly generated in the range $[0.5, 1.5]$.

5.2 Simultaneous Clock Synchronization

We saw in the previous section that consensus problems spanning multiple applications could be parameterized in terms of the relative skew rates of the individual agent clocks. In this section, we propose a simple clock synchronization algorithm, and then propose including these shared clock dynamics in some of the previously identified problem formulations to consider the advantages of simultaneous consensus formation and clock synchronization. This is not necessarily a novel contribution; as there have been a variety of algorithms proposed for synchronizing discrete clock dynamics of the form (2.1) with similar virtual clock augmentations, but here we consider a simpler continuous model so that we can seamlessly integrate simultaneous clock synchronization into the continuous problems considered in the previous section.

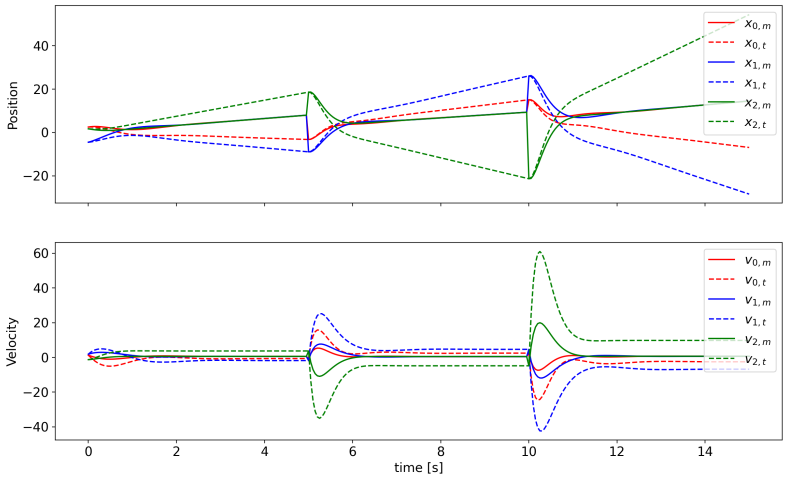


Figure 5.3: Simulation of unsuccessful periodic measurement update for graph on 3 nodes with $\delta = 5$. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed. Clock skews are equal to $[2.74, 2.85, 2.65]$ (red,blue,green).

5.2.1 Shared Clock Model and Synchronization

One potential mitigation for the failed-consensus situations that we observed in the previous section is to have synchronized clocks between all agents. However, synchronization cannot occur on the individual clock parameters, as individual agents realistically do not have knowledge of their individual clock parameters. Furthermore, accurately determining their clock parameters (via system identification or an online learning algorithm), requires access to a true clock, which may be unavailable for some applications. Thus, the procedure presented here does not require any knowledge of individual clock parameters, just adjusted measurements of agents' clocks.

As described previously, each agent is assumed to have an internal clock of the form (2.1). While we want to continue under the further assumption that agents do not have access to their clock parameters, we can note that agents can perform arbitrary resets of their internal

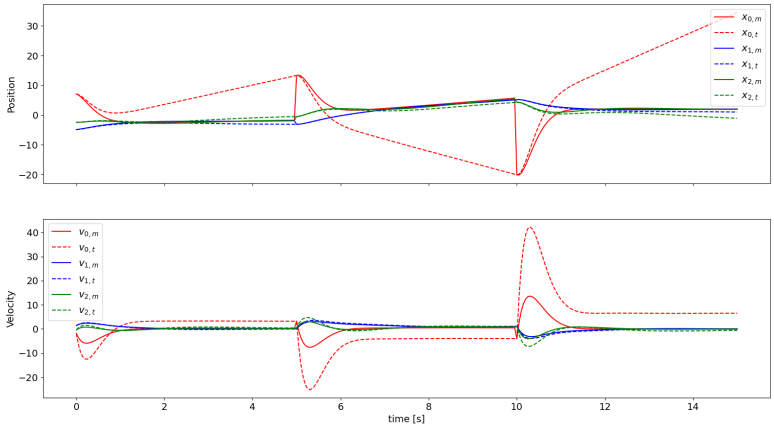


Figure 5.4: Simulation of unsuccessful periodic measurement update for graph on 3 nodes with $\delta = 5$ and no implementation error. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed. Clock skews are equal to $[2.62, 1.17, 1.64]$ (red,blue,green).

clock. That is, at any time, denoted by t_r, agents can reset their clock offset by storing their current clock reading and tracking,

$$\tau_{r,i}(t) = \tau_i(t) - \tau_i(t_r).$$

For times $t \geq t_r$, $\tau_{r,i}(t)$, the reset clock evolves with the same skew rate as agent i, but with zero offset. Then, note that we can artificially alter the skew rate and offset of the reset clock by applying a multiplicative and additive factors, such as,

$$\tau_i'(t) = \alpha_{c,i}\tau_{r,i}(t) + \beta_{c,i}. \tag{5.1}$$

In the virtual clock model, each agent locally has known clock parameters $\alpha_{c,i}$ and $\beta_{c,i}$ which translate a local reset clock reading ($\tau_{r,i}(t)$) to an adjusted clock reading. As the reading of the local clock and reset can be done without knowledge of the local parameters, and

multiplying/adding that measurement to known parameters is an allowable operation, the computation of the virtual clock can be done by each agent. The goal is for each agent to self-adjust their local reset clock reading such that they all track a shared clock model,

$$\tau_s(t) = \bar{\alpha}t + \bar{\beta}_c. \tag{5.2}$$

We can note that, in the case where all local clock parameters are known along with a desired shared clock (which could simply be GPS time, e.g. $\bar{\alpha} = 1$, $\bar{\beta}_c = 0$), then the local skews parameters, adjusted skew parameters, and the shared skew parameter are related by the following expressions,

$$\begin{aligned} \bar{\alpha} &= \alpha_{c,i}\alpha_i \ \forall i \in \ 1, /dots, n \\ \bar{\beta}_c &= (\alpha_{c,i}\alpha_i - \bar{\alpha})\, t + \beta_{c,i}. \end{aligned} \tag{5.3}$$

Thus, the goal of clock synchronization without knowledge of the local clock parameters is for agents to share information that allows for consensus on the shared clock parameters to give rise to a single shared clock of the form (5.2). The consensus algorithm below performs a continuous update to the local parameters $\alpha_{c,i}$ and $\beta_{c,i}$ such that (5.3) find a stable, steady state value. We allow the virtual parameters to be updated locally, so we take the following dynamics,

$$\frac{d}{d\tau}\begin{bmatrix} \tau' \\ \alpha_c \\ \beta_c \end{bmatrix} = \begin{bmatrix} 0 & I & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \tau' \\ \alpha_c \\ \beta_c \end{bmatrix} + \begin{bmatrix} 0 \\ u_\alpha \\ u_\beta \end{bmatrix}$$
$$\begin{bmatrix} \tau'(0) \\ \alpha_c(0) \\ \beta_c(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

where τ', α_c, and β_c are the stacked vectors of the individual agents' virtual times, virtual skews, and virtual offsets, respectively. Before introducing the control inputs u_α and u_β, we

can note that for inputs equal to zero, the above virtual clock model simply reproduces the dynamics of each agent's reset clock. We have also abused the derivative notation here, such that we have overloaded the $\frac{d}{d\tau}$ notation to signify that each agent's time and parameter dynamics are evolving at their individual clock rate - analogous to the inertial navigation calculations in previous chapters.

Now, we would like to update the local, shared clock parameters to allow these virtual clocks to agree on a shared clock reading. To that end, consider the following control inputs,

$$u_\alpha = \sum_{j\sim i} \tau_j' - \tau_i' + \sum_{j\sim i} \alpha_{c,j} - \alpha_{c,i}$$
$$u_\beta = \sum_{j\sim i} \tau_j' - \tau_i' + \sum_{j\sim i} \beta_{c,j} - \beta_{c,i},$$

which can also be combined with the vector version of the clock dynamics in the following form,

$$\frac{d}{d\tau}\begin{bmatrix} \tau' \\ \alpha_c \\ \beta_c \end{bmatrix} = \begin{bmatrix} 0 & I & 0 \\ -\mathcal{L} & -\mathcal{L} & 0 \\ -\mathcal{L} & 0 & -\mathcal{L} \end{bmatrix}\begin{bmatrix} \tau' \\ \alpha_c \\ \beta_c \end{bmatrix} \tag{5.4}$$

Before continuing, we must address an implicit assumption in 5.4 and the preceding setup.

Assumption 3. *All agents can reset their local clocks simultaneously.*

On initial consideration, this assumption would seem to imply that all agents have access to a global clock, or a single entity has global knowledge of all clock parameters and thus is able to inform agents of the local clock reading that corresponds to t_r. However, we can note that t_r (and the corresponding local clock readings) is arbitrary, thus it doesn't necessarily matter when it occurs (globally), just that it occurs at the start of the initialization of the clock synchronization routine 5.4. Thus, all that the assumption is requiring is that either a single entity can simultaneously trigger all agents to start synchronization (which may be expensive and rare, but not entirely restrictive), or that a single agent (a leader)

initiates the synchronization and the command propagates over the communication network instantaneously. This is, of course, never strictly true, but instantaneous communication between neighboring agents is a common assumption in the clock synchronization literature, so is not completely inappropriate here. In applications where the latency between agents is non-negligible, further work is necessary to achieve clock synchronization.

First, we would like to consider the stability of these dynamics in isolation.

Theorem 7. *The clock synchronization dynamics (5.4) achieve are stable, thus, promote an equilibrium. Formally, for $\mathcal{S} \succ 0$, the closed loop clock synchronization dynamics given in (5.4) are marginally stable, that is,*

$$Re\{\lambda(A_c)\} \leq 0$$

for all eigenvalues of A_c, which is defined as

$$A_c = \begin{bmatrix} 0 & I & 0 \\ -\mathcal{L} & -\mathcal{L} & 0 \\ -\mathcal{L} & 0 & -\mathcal{L} \end{bmatrix}.$$

Proof. We start by quantifying the spectrum of A_c in terms of the eigenvalues of $\mathcal{L}$. This is done by considering,

$$\begin{aligned} 0 &= \det\left(\lambda I_{3n} - A_c\right) \\ 0 &= \det\left(\begin{bmatrix} \lambda I & -I & 0 \\ \mathcal{L} & \lambda I + \mathcal{L} & 0 \\ \mathcal{L} & 0 & \lambda I + \mathcal{L} \end{bmatrix}\right). \end{aligned}$$

From [97, Theorem 2.1], the determinant of a block matrix with this structure can be computed by,

$$\det\left(\lambda I_{3n} - A_c\right) = \det\left(\lambda I + \mathcal{L}\right)\det\left(\lambda I + \mathcal{L}\right) \times \det\left(\lambda I + \left(\lambda I + \mathcal{L}\right)^{-1}\mathcal{L}\right). \tag{5.5}$$

Recall that $\mathcal{L} = U\Lambda U^T$. Denoting the eigenvalues of $\mathcal{L}$ by μ_i, such that $0 = \mu_1 < \mu_2 \leq \cdots \leq \mu_n$, then allows,

$$\begin{aligned}\left(\lambda I + \mathcal{L}\right)^{-1}\mathcal{L} &= \left(U^T\lambda I U + U^T\Lambda U\right)^{-1}U^T\Lambda U \\ &= U^T\left(\lambda I + \Lambda\right)^{-1}\Lambda U \\ &= U^T\mathrm{diag}\left[\frac{\mu_1}{\lambda+\mu_1},\ldots,\frac{\mu_n}{\lambda+\mu_n}\right]U.\end{aligned}$$

Combining this with the remaining factor of λI_n, and recalling that the determinant of a matrix is equal to the product of its eigenvalues, we have,

$$\det\left(\lambda I + \left(\lambda I + \mathcal{L}\right)^{-1}\mathcal{L}\right) = \prod_{i=1}^{n}\lambda + \frac{\mu_i}{\lambda+\mu_i}.$$

Via similar machinations, we can also see that,

$$\det\left(\lambda I + \mathcal{L}\right) = \prod_{i=1}^{n}(\lambda+\mu_i).$$

Combining the previous results with (5.5), allows us to set the characteristic equation for A_c equal to zero and simplify,

$$\begin{aligned}0 &= \prod_{i=1}^{n}\left(\lambda+\mu_i\right)^2\left(\lambda + \frac{\mu_1}{\lambda+\mu_1}\right) \\ &= \prod_{i=1}^{n}\lambda^3 + 2\lambda^2\mu_i + \lambda(\mu_i+\mu_i^2) + \mu_i^2 \\ &= \prod_{i=1}^{n}(\lambda+\mu_i)(\lambda^2+\lambda\mu_i+\mu_i)\end{aligned}$$

So that we can see that every eigenvalue of $\mathcal{L}$, determines three eigenvalues of A_c,

$$\lambda_{i:1} = -\mu_i$$
$$\lambda_{i:2,3} = \frac{-\mu_i \pm \sqrt{\mu_i^2 - 4\mu_i}}{2}$$

Thus, there are three zero eigenvalues from $\mu_1 = 0$, and then, from $\mu_i > 0$ for $i > 1$, we can see that all non-zero eigenvalues of A_c have negative real parts. This satisfies the definition for marginal stability. □

The previous theorem considered the clock synchronization dynamics in the agents' time scales (generated by the clock skew rate). This does not change the stability result, as we saw in Chapter 4 (Proposition 12), left multiplication by a positive-definite scaling matrix does not alter the stability of the Laplacian matrix. In Figure 5.5 we see a simulation for 20 agents with random skews and initial offsets subject to (5.4).

Next, we can consider the equilibrium of these dynamics.

Theorem 8. *The consensus-based clock synchronization algorithm in (5.4) promotes an equilibrium that corresponds to synchronization of each agent's clock to a shared clock of the form (5.2).*

The shared clock parameters ($\bar{\alpha}, \bar{\beta}_c$ in (5.2)) have the following equilibrium values,

$$\bar{\alpha} = \frac{\sum_{i=1}^{n} \frac{1}{\alpha_i}}{\sum_{i=1}^{n} \frac{1}{\alpha_i^2}}$$
$$\bar{\beta}_c = \bar{\tau}\mathbf{1} - \tau'(t)$$

where $\bar{\tau}$ is the average virtual time at time t.

Proof. First, from inspection, we can find a left eigenvector associated with an zero eigenvalue,

$$v_0 = [\mathbf{0}\ \mathbf{1}^T\mathcal{S}^{-1}\ \mathbf{1}^T\mathcal{S}^{-1}].$$

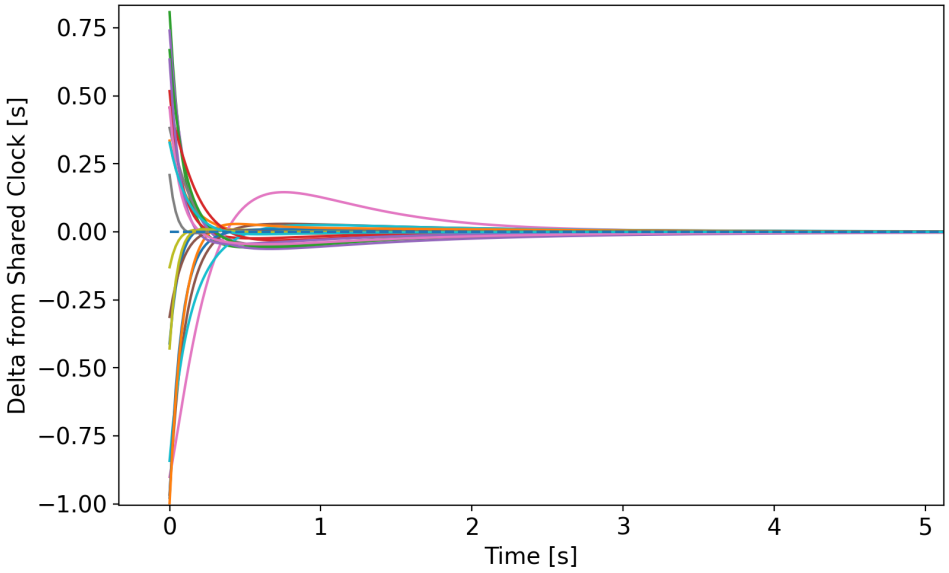


Figure 5.5: Separation between individual agents' shared clock reading and the resulting shared clock after synchronization. Agent skews were taken from a uniform random distribution from 0.5 to 1.5, and initial offsets were taken from a uniform random distribution from 0.0 to 2.0.

Denoting the state vector in 5.4 as $s_c(t)$. We can note that $v_0^T \dot{s}_c(t) = 0$, thus, $v_0^T s_c(t)$ is an invariant quantity. From the initial conditions, we can further note that,

$$\mathbf{1}^T \mathcal{S}^{-1} \alpha_c(t) = \sum_{i=1}^{n} \frac{1}{\alpha_i}$$
$$\mathbf{1}^T \mathcal{S}^{-1} \beta_c(t) = 0.$$

With those invariant quantities in place, first let us consider the skew equilibrium. We know that in equilibrium, the intrinsic skews and the virtual skew parameters obey,

$$\mathcal{S}\alpha_c = \bar{\alpha}\mathbf{1},$$

that is, the local skew parameters correct the skews to a common value. The matrix of skew parameters is invertible, so we can solve for the vector of local skew parameters,

$$\alpha_c = \bar{\alpha}\mathcal{S}^{-1}\mathbf{1}.$$

However, to find the exact values of the entries of α_c, we must know $\bar{\alpha}$, which we can use the invariance quantity of the initial condition to solve for,

$$\mathbf{1}^T -^{-1} \bar{\alpha}\mathcal{S}^{-1}\mathbf{1} = \mathbf{1}^T \mathcal{S}^{-1}\alpha_c$$
$$\bar{\alpha} = \frac{\sum_{i=1}^{n} \frac{1}{\alpha_i}}{\sum_{i=1}^{n} \frac{1}{\alpha_i^2}},$$

which is the desired result for the skew parameter. For the offsets, we can follow a similar process. We know that in equilibrium, the adjusted reset times plus the local offsets should give us a uniform shared clock reading. That is,

$$\bar{\beta}_c = \bar{\tau}(t)\mathbf{1} - \tau'(t)$$

where $\bar{\tau}(t)$ is the shared clock reading at time t, and $\bar{\beta}_c$ is the vector of final local offsets that bring the skew-adjusted reset clocks into agreement. Once again, left multiplying by $\mathbf{1}^T\mathcal{S}^{-1}$ allows us to invoke the invariant quantity from the initial condition, which in this case show,

$$\begin{aligned} 0 &= \bar{\tau}(t)\mathbf{1}^T\mathcal{S}^{-1}\mathbf{1} - \mathbf{1}^T\mathcal{S}^{-1}\tau'(t) \\ \bar{\tau}(t) &= \frac{\mathbf{1}^T\mathcal{S}^{-1}\tau'(t)}{\mathbf{1}^T\mathcal{S}^{-1}\mathbf{1}} \\ &= \frac{\mathbf{1}^T\mathcal{S}^{-1}\mathcal{S}\alpha_c(t-t_r)}{\mathbf{1}^T\mathcal{S}^{-1}\mathbf{1}} \\ &= \frac{\mathbf{1}^T\alpha_c(t-t_r)}{\sum_{i=1}^{n}\frac{1}{\alpha_i}}, \end{aligned}$$

that is, that the shared clock time is simply the average of the skew-corrected local clocks, corrected for the bulk clock rates in the network. Then, the result for the local values of $\bar{\beta}_c$ follows directly. □

It is important to note that in practice, the values of α_c, $\bar{\alpha}$ and $\bar{\beta}_c$ can not be calculated *a priori*. If they could be, this would imply that there is a global entity with knowledge of all clock parameters - a scenario under which clock synchronization is not necessary as all agents could be assigned local corrections to place all agents on any arbitrary clock. Instead, the results of Theorem 8 serve as an abstraction that allows us to reason about the results of using an algorithm such as (5.4). First, the equilibrium values solidifies our intuition about the development of (5.4): the skew parameter control input corrects all clocks to a uniform skew. However, as the control law uses only local information, this necessarily results in clock trajectories that diverge slightly before achieving a uniform skew. When the clocks are on the same skew, offset control input then finds the difference between the corrected reset clocks and the average of all clocks, which allows for the computation of a true shared clock for all agents. From the result for the shared skew rates, we can also note the following limits.

Corollary 2. *The equilibrium shared skew rate is bounded by the minimum and maximum*

agent skew rates,

$$\frac{\alpha_{\min}^2}{\alpha_{\max}} \leq \bar{\alpha} \leq \frac{\alpha_{\max}^2}{\alpha_{\min}}.$$

Proof. This follows directly from noting that $\frac{n}{\alpha_{\max}} \leq \sum_{i=1}^{n} \frac{1}{\alpha_i} \leq \frac{n}{\alpha_{\min}}$ and $\frac{n}{\alpha_{\max}^2} \leq \sum_{i=1}^{n} \frac{1}{\alpha_i^2} \leq \frac{n}{\alpha_{\min}^2}$ along with the definition of $\bar{\alpha}$. □

In light of the discussion of limitations clock skew rates in Section 5.1, we can note that this result potentially allows for very large consensus values of the shared clock parameter, potentially leading to instances where the shared clock will exhibit the growth of error behavior that is associated with fast-clocked agents.

Remark 12. *In light of the relation between the consensus values of the shared clock from Theorem 8, as well as the relations between the local adjusted parameters and the shared clock parameters (5.3), it could be proposed that after synchronization has occured, one could propose (if there is access to a true clock for sequential measurements), a simple "system identification" routine that would allow the shared clock rate to become known. From there, each agent could use the stored value of their local adjusted parameter to find an estimate of their true clock parameters. As noted before, knowledge of these parameters allows for arbitrary clock adjustment/correction. This highlights an intrinsic choice between synchronization without true parameter knowledge, or undertaking system identification before clock synthesis. The trade off between the two is that to find the true values of local parameters, all agents need access to a true clock reading periodically, however once this is done, adjustment to any desired shared clock rate or model is possible. On the other hand, blind synchronization does not require access to a true clock, but precludes further adjustment.*

We can also note (again), that the clock parameters for each agent may be slowly varying with time, and thus a single instance of system identification would eventually no longer accurately describe the agent clock, and the agents would desynchronize with time. Continually running (5.4) could mitigate these issues, as the shared clock rate would continually evolve with the changing local parameters. However, if there is some (potentially) non-linear form

of the local clocks (for example, a sinusoidal drift in skew rate as a function of time, possibly due to changes in operating temperature over an day/night cycle), then including system identification to ascertain the nonlinear clock model would be worth undertaking.

5.2.2 Combined Clock Synchronization and Periodic Updates

We'd now like to simulate the effects of combining both the periodic true state updates from the beginning of the chapter, with simultaneous clock synchronization and see if we can improve the consensus achievement. In Figures 5.6 and 5.7 we can see the same initial conditions and skews ($[1.49, 0.95, 1.44]$) but for the system with periodic state updates with and without simultaneous synchronization, respectively. The results are fairly clear that the simultaneous clock synchronization allows for a marked improvement in velocity consensus. There is still some positional drift between the true and measured positions, however, that is fairly easily explained by the non-unity shared clock parameter causing integration errors of the measured states. However, this does not pull the velocities out of near-consensus as the positions are near consensus even as they propagate slightly off from the true clock rate. The periodic updates, then, achieve their goal of periodically resetting the accumulated error.

We can also test this combination when large skews are present. In Figures 5.8 and 5.9 we can see the no-synchronization and synchronization cases for clock skews equal to ([2.94, 0.67, 2.65]). We can see that, as expected, the simulation with no synchronization exhibits the growth in errors of the periodic updates. However, when the clocks are allowed to synchronize, the system does not exhibit the growth in errors and achieves near-consensus as for the smaller skew case in the previous simulations.

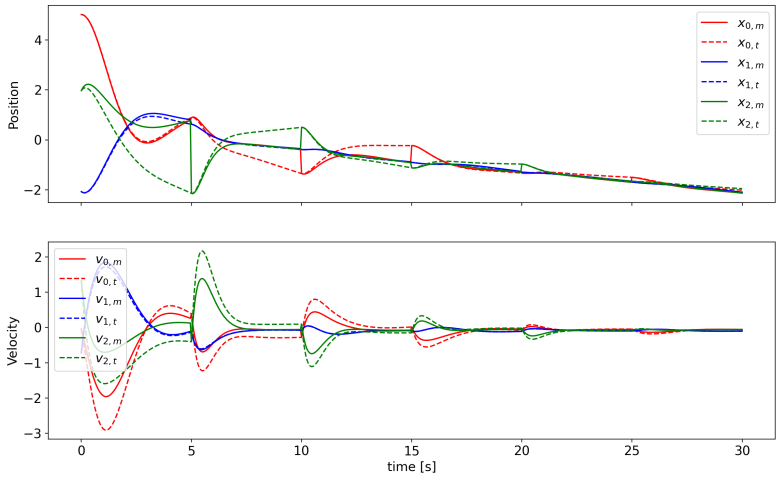


Figure 5.6: Simulation of periodic measurement update (no simultaneous clock synchronization) for graph on 3 nodes with $\delta = 5$. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed. Clock rates are randomly generated in the range $[0.5, 1.5]$

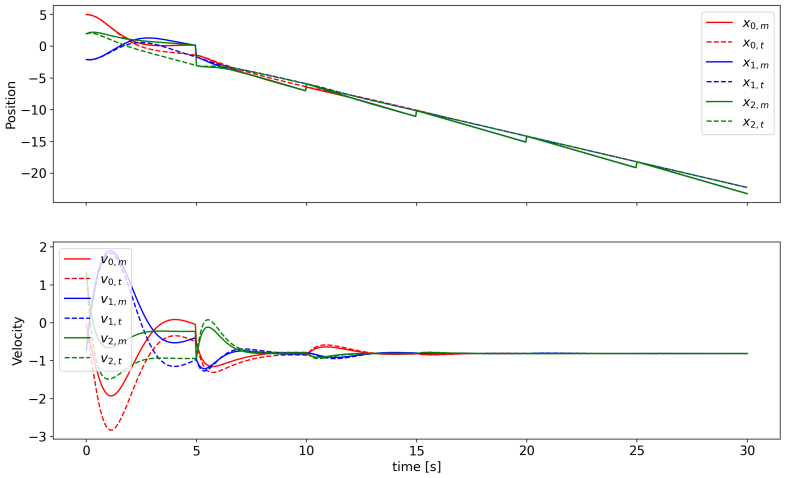


Figure 5.7: Simulation of periodic measurement update combined with simultaneous clock synchronization for graph on 4 nodes with $\delta = 5$. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed. Clock rates are randomly generated in the range $[0.5, 1.5]$.

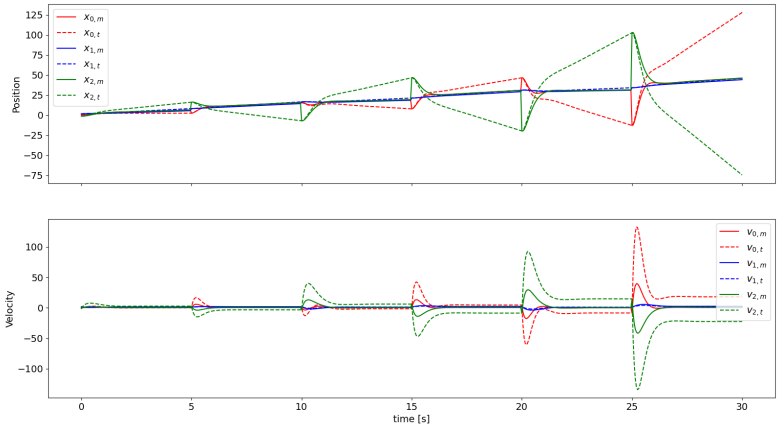


Figure 5.8: Simulation of periodic measurement update (no simultaneous clock synchronization) for graph on 3 nodes with $\delta = 5$ for large skews. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed.

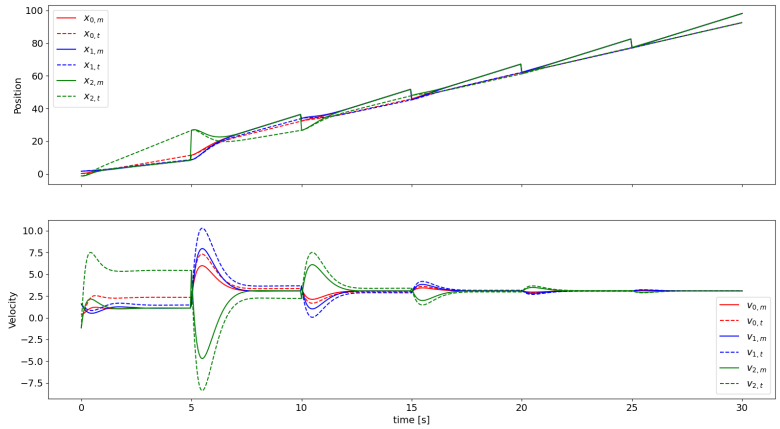


Figure 5.9: Simulation of periodic measurement update combined with simultaneous clock synchronization for graph on 4 nodes with $\delta = 5$ for large skews. Internal (inertial) state estimates are plotted in solid, with true state values plotted in dashed.

5.3 Chapter Summary

In this chapter, we have presented some very preliminary investigations into utilizing periodic state updates and simultaneous clock synchronization to mitigate the failed consensus behavior that we observed for RAIN systems as a result of internal clock biases. Future work suggested by this section is a quantification of these effects along with guarantees related to the initial skew distribution, as well as methods for leader-follower clock synchronization which could promote synchronization to a true clock rate for networks containing one-to-a-few agents with access to a true clock.

Chapter 6

MISCELLANEOUS RESULTS AND APPLICATIONS

6.1 Graph Theoretic Interpretations of Agent-Based Skew Rates

6.1.1 Weighted Digraphs

A natural question when first considering a scaled Laplacian is whether or not the time scale parameters can be interpreted as edge weights in some fashion. Obviously, the asymmetry of the scaled Laplacian precludes it being a weighted, undirected graph, but directed graphs are naturally represented by asymmetric Laplacian matrices, so they are a natural candidate for an alternative representation. To determine the appropriate representation, consider a row of an arbitrary scaled Laplacian, which can be written as,

$$[\tilde{L}(\mathcal{G})]_i = \frac{1}{\epsilon_i}\begin{bmatrix} w_{i1} & w_{i2}\dots & \deg(i) & \dots & w_{in} \end{bmatrix}.$$

where w_{ij} is the weight of the edge in the undirected graph between agents i and j. Here, we can allow $w_{ij} = 0$ if $i \nsim j$ to allow for the above row to be completely generic, and $\deg(i)$ here is the weighted degree of agent i equal to,

$$\deg(i) = \sum_{j=1}^{n} w_{ij}.$$

The original graph in question being undirected ensures that the in and out degrees are equal and simply the sum of the adjacent edge weights, however, we'll see that a distinction between the two is necessary in the directed interpretation. More exactly, define the following edge weighting,

$$\tilde{w}_{ij} = \frac{w_{ij}}{\epsilon_i},$$

which is applied to any edge incoming to node i. Then, the diagonal element of the ith row of the scaled Laplacian corresponds to the weighted *in-degree* of agent i. Interpreting each row of $\tilde{L}(\mathcal{G})$ in this fashion gives the weighted, directed graph interpretation of a scaled Laplacian: each undirected edge in $\mathcal{G}$ is replaced by two directed edges in opposing directions, with weights equal to $\tilde{w}_{ij}$ and $\tilde{w}_{ji}$. Pictorially, we can see this in Figure 6.1.

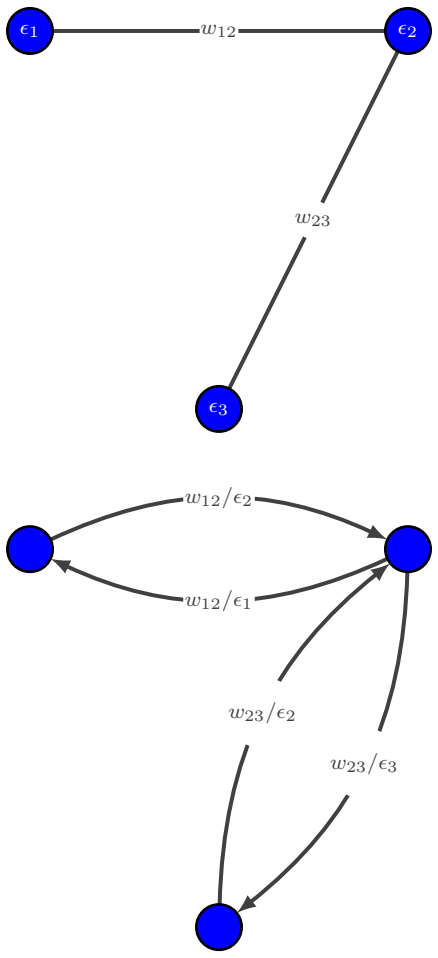


Figure 6.1: Simple scaled graph (top) with the directed graph interpretation (bottom).

So we can see that any scaled graph can be reinterpreted as a directed graph with weighting adjusted by the time scale parameters on edges incoming to the nodes. We'll reference this equivalence later in this section in Section 6.3, but in general, arbitrarily weighted di-

graphs do not enjoy the same graph-theoretic interpretations and results as those found for undirected graphs (primarily due to the fact that any linear system can be interpreted as representing a weighted directed graph), thus, we do not explicitly call this interpretation into use often.

6.1.2 Asymptotically Valid Reduced Order Models of Graphs with Sparsely Connected Complete Subgraphs

In most of the presented main results, we have focused on the performance and analysis of scaled graphs, but the prime mover of our analyses have been graph and linear systems theory, and we have not made use of singular perturbation theory. This has been in part due a desire to retain the abundance of information and interpretation of networked problems based on the properties of the graph Laplacian. Blindly applying hierarchical model reduction via nested Tikhonov's theorem will not necessarily preserve any meaningful network structure. Additionally, forgoing any explicit time scale separation allows for a relaxation of some of the necessary assumptions on the smallness and relative order of the scaling parameters. When imposed, these assumptions do not hinder the previous graph-theoretic-centric analysis so the results there still apply even with the restrictions imposed.

However, to completely ignore methods from singular perturbation theory would be to deny ourselves an incredibly deep and powerful set of tools for modeling and analysis. Thus, in this section, we detail a minor result delving into asymptotically valid lifting models that allow us to approximate a multi-scale graph as a mono-scale graph where the time scales manifest as a result of the topology. At first, this may seem like a step backwards based on the results of the previous chapters. However, we have also seen the difficulty of including time scales introduces, so approximating the system as a mono-node-scale graph would allow almost all of the current graph-theoretic results for network analysis to be applied as an approximation to the full-scaled network. Singular perturbation theory provides the bridge that allows us to link these two areas together, specifically, Tikhonov's theorem and the asymptotic guarantees that it provides.

Equivalent Model

We are principally interested in systems of the following form,

$$\begin{aligned} E\dot{x}(t) &= -\mathcal{L}x(t) \\ h(t) &= x(t), \end{aligned} \tag{6.1}$$

where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ defines a communication topology for $|\mathcal{V}| = n$ agents, undirectedly connected by $|\mathcal{E}| = m$ identically weighted edges. The graph Laplacian, $\mathcal{L}$ defines the agent dynamics, with the agent ν_i's dynamic rate determined by the associated time scale parameter $\epsilon_i > 0$, such that $E = \text{diag}[\epsilon_1, \ldots, \epsilon_n]$. We can also consider this system under an extraneous signal, $u(t)$:

$$\begin{aligned} E\dot{x}_f(t) &= -A(\mathcal{G}, l)x_f(t) + \text{B}u(t) \\ h(t) &= x_f(t), \end{aligned} \tag{6.2}$$

where $x_f(t)$ denotes the follower nodes, $A(\mathcal{G}, l)$ is the Laplacian that is grounded by some input into a set of nodes, l, and B is the input matrix, and $h(t)$ is the output of the system, taken here to be the full agent states. The signal $u(t)$ can be taken to represent additional noise being injected into the system, a designed control input (either to promote a desired consensus value, or to subvert the natural consensus value via nefarious action). For both (6.1) as well as (6.2), we make our standard assumption that the graph is undirected and connected (Assumption 1). Note that the consequence of Assumption 1 is that the (grounded) Laplacian matrices in (6.1) and (6.2) have a single zero eigenvalue and the remaining eigenvalues are real and positive. Also recall that multiplication by the inverse scaling matrix E^{-1} does not alter these base properties [36].

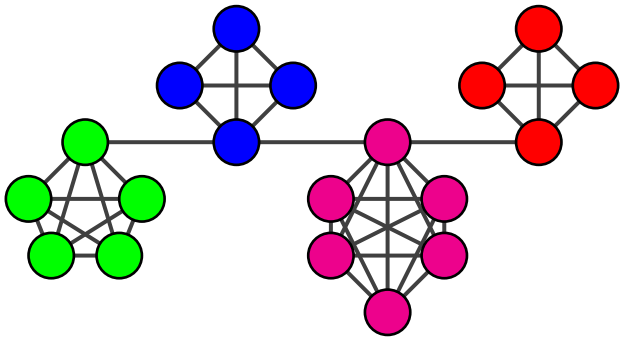


Figure 6.2: Example lifted graph with $\mathcal{P}_4$ backbone graph.

6.1.3 Reduced Order Modeling of Lifted Graphs

With these reference systems in mind, consider the following consensus system over a graph $\mathcal{G}'$: n complete graphs, of m_i nodes in the ith complete graph. Connecting these complete graphs are the edges from $\mathcal{G}$, such that only one agent in each K_{m_i} has external connections. We'll refer to the set of edges from $\mathcal{G}$ as the *backbone* topology of the graph $\mathcal{G}'$, and the interconnection of the n complete graphs in total as the *lifted graph*. An example of such a lifted graph is shown in Figure 6.2.

Following [23, 7], we can consider the node and area parameters for an arbitrary lifted graph, with the natural choice of partitioning being each complete graph as an individual area. Thus, we have n areas over the $N = \sum_{i=1}^{n} m_i$ nodes in the lifted graph. The node parameters are defined as in [23], with $c^I_{\alpha,i}$ being the number of internal connections of node i in area α, and $c^E_{\alpha,i}$ as the number of external connections. Here, internal and external denote an edge connecting a node to a neighbor within or outside of the area α. Given that each area is composed of a complete graph, and we are given an auxiliary graph that represents the external connections between areas, we can see that for all $\alpha \in \mathcal{I}_n$ and $i \in \mathcal{I}_{m_\alpha}$,

$$c^I_{\alpha,i} = m_\alpha - 1$$
$$c^E_{\alpha,i} = \begin{cases} \deg(\nu_\alpha, \mathcal{G}) & i \in \mathcal{I}_{m_\alpha},\ i = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Thus, we can define the node parameter for $\mathcal{G}'$ as,

$$d = \frac{\max_{\alpha,i} c^E_{\alpha,i}}{\min_{\alpha,i} c^I_{\alpha,i}} = \frac{\max \deg(\nu_\alpha, \mathcal{G})}{\min_\alpha m_\alpha - 1}.$$

Likewise, the area parameters are defined as the total number of internal or external connections for each area, denoted γ^I_α and γ^E_α, respectively. Again, the fact that all areas are represented by complete graphs with known interconnections allows us to represent these parameters succinctly,

$$\gamma^I_\alpha = m_\alpha(m_\alpha - 1)$$
$$\gamma^E_\alpha = \deg(\nu_\alpha, \mathcal{G}).$$

The bulk area parameter is then defined as the ratio of the maximal external and minimal internal area parameters,

$$\delta = \frac{\max \deg(\nu_\alpha, \mathcal{G})}{\underline{\mathrm{m}}(\underline{\mathrm{m}} - 1)}.$$

Finally, we can note that due to the fact that only one node in each area has external connections, we can relate the node and area parameters,

$$\delta = \frac{d}{\underline{\mathrm{m}}}. \tag{6.3}$$

In order for the graph to be characterized as having dense areas with sparse interconnections, and thus, able to be reduced in the method that follows, it must satisfy $d \ll 1$ and $\delta \ll 1$, with the area parameter being the more critical parameter [23]. From (6.3) combined

with $\underline{\mathrm{m}} \geq 1$, $d \ll 1$ ensures $\delta \ll 1$. In the remainder of this paper, we will assume this condition is satisfied by $\mathcal{G}'$, which we codify in Assumption 4

Assumption 4. *Given the backbone graph $\mathcal{G}$, the lifted graph $\mathcal{G}'$ satisfies*

$$\max \deg(\nu_\alpha, \mathcal{G}) \ll \underline{m},$$

where $\underline{m}$ is the number of nodes in the smallest complete graph area in $\mathcal{G}'$.

For agents connected with topologies that satisfy Assumption 4, we would like to consider consensus dynamics, that is,

$$\dot{x}_l(t) = -\mathcal{L}(\mathcal{G}')\, x_l(t). \tag{6.4}$$

Now, we would like to consider the reduced order modeling of consensus on the lifted graph (6.4). To begin, we can define the slow and fast variables. The fast variables are the relative states within each area, defined by,

$$z_\alpha = Q_\alpha x^\alpha,$$

where Q_α is an $(m_\alpha - 1) \times m_\alpha$ "differencing" matrix that defines $z_{\alpha,i}$ as the difference between an arbitrary reference node in area α and x_i. This difference can be an unweighted difference as in [23],

$$Q_\alpha = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 \\ -1 & 0 & 1 & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ -1 & 0 & 0 & \cdots & 1 \end{bmatrix},$$

or a weighted difference of all area nodes as in [7],

$$Q_\alpha = \begin{bmatrix} -1+(m_\alpha-1)\nu & 1-\nu & -\nu & \cdots & -\nu \\ -1+(m_\alpha-1)\nu & -\nu & 1-\nu & \cdots & -\nu \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ -1+(m_\alpha-1)\nu & -\nu & -\nu & \cdots & 1-\nu \end{bmatrix},$$

where $\nu = (m_\alpha - \sqrt{m_\alpha})/m_\alpha(m_\alpha - 1)$. The advantage of a more complex weighting of the inter-area node differencing in the definition of the fast variable is that when Q_α has orthonormal rows perpendicular to $\mathbf{1}$, it can be observed that,

$$Q_\alpha \mathbf{1}_{m_\alpha} = 0, \quad Q_\alpha Q_\alpha^T = I_{m_\alpha - 1},$$

as well as $z_\alpha = 0$ when all nodes in an area are synchronized. When Q_α does not satisfy the above criteria, we can still find an inverse transformation from the fast variables back to the agent states by noting that $Q_\alpha^\dagger = Q_\alpha^T(Q_\alpha Q_\alpha^T)^{-1}$, resulting in,

$$Q_\alpha^\dagger = \frac{1}{m_\alpha} \begin{bmatrix} -1 & -1 & \cdots & -1 \\ m_\alpha - 1 & -1 & \cdots & -1 \\ -1 & m_\alpha - 1 & \cdots & -1 \\ \cdot & \cdot & \cdot & \cdot \\ -1 & -1 & \cdots & m_\alpha - 1 \end{bmatrix}.$$

The slow motion is taken as the aggregate of all nodes within an area,

$$y_\alpha = \sum_{i=1}^{m_\alpha} \frac{x_i^\alpha}{m_\alpha} = \frac{1}{m_\alpha} u_\alpha^T x^\alpha,$$

where $u_\alpha = \mathbf{1}_{m_\alpha}^T$. We can stack all of the node states in vector form and express the slow and fast variables as,

$$y = M^{-1} U^T x$$

$$z = Qx,$$

and $M = \text{diag}(m_1, \ldots, m_n)$, $U = \text{blockdiag}(\mathbf{1}_{m_1}, \ldots, \mathbf{1}_{m_n})$, and $Q = \text{blockdiag}(Q_1, \ldots, Q_n)$. Using the definitions of y_α along with the inverse transformations to the fast variables, we can write the slow-fast system as a linear system in y, z alone,

$$\begin{bmatrix} \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix}. \tag{6.5}$$

The constituent matrices are defined as follows,

$$\begin{aligned} \bar{A}_{11} &= -CL(E, \mathcal{G}')\, U \\ \bar{A}_{12} &= -CL(E, \mathcal{G}')\, Q^* \\ \bar{A}_{21} &= -QL(E, \mathcal{G}')\, U \\ \bar{A}_{22} &= -Q(L(I, \mathcal{G}') + L(E, \mathcal{G}'))Q^* \end{aligned}$$

where $C = M^{-1}U^T$, and $L(E, \mathcal{G}')$ and $L(I, \mathcal{G}')$ are defined as the Laplacian matrices containing all of the external connections and internal (for all areas) in $\mathcal{G}'$, respectively. We can see that $L(E, \mathcal{G}')$ is simply the Laplacian of the backbone topology, buffered with zeros for all rows and columns corresponding to nodes with only internal connections. Finally, we use Q^* to denote the inverse of Q, which could be Q^T or $Q^\dagger$ depending on the convention of Q chosen.

Of predominant interest is the dynamics of the slow system, which we can write as,

$$\dot{y} = \bar{A}_{11} y + \bar{A}_{12} z.$$

As noted in [23, 7, 103], the order of these matrices ($\bar{A}_{11}, \bar{A}_{12}, \bar{A}_{21} \in \mathcal{O}\left(c^I \delta\right)$ and $\bar{A}_{22} \in \mathcal{O}\left(c^I\right)$) suggests multi-scale behavior in the system, and allows for the definition of two relevant time scales to consider dynamics on, a fast time scale,

$$t_f = c^I t,$$

as well as a slow time scale,

$$t_s = c^I \delta.$$

The system matrices in (6.5) can then be rescaled. In the slow time scale, the fast dynamics of the inter-area relative states will converge to a slow manifold. Solving for this manifold and substituting into (6.5) gives the following reduced systems,

$$\begin{aligned}
\frac{dy_s}{dt_s} &= \left(A_{11} - dA_{12}A_{22}^{-1}A_{21}\right) y_s, \quad y_s(0) = y(0) \\
\frac{dz_f}{dt_f} &= A_{22} z_f, \quad z_f(0) = z(0) + dA_{22}^{-1}A_{21}y(0).
\end{aligned}$$

In the previous literature, the following result, following from Tikhonov's Theorem is relied upon.

Theorem 9. *The two-time scale behavior of the inter-area dynamics along with the slow dynamics between area can be written in singularly perturbed standard form. Thus, the reduced order models for those dynamics are asymptotically valid below critical values of the area and node parameters.*

Formally, There exists δ^* and d^* such that for all $0 < \delta \leq \delta^*$ and $0 < d \leq d^*$, the reduced systems approximate the full system solutions, such that,

$$\begin{aligned}
y(t) &= y_s(t_s) + \mathcal{O}\left(\delta d\right) \\
z(t) &= -dA_{22}^{-1}A_{21}y_s(t_s) + Z_f(t_f) + \mathcal{O}\left(\delta d\right)
\end{aligned}$$

Proof. As noted, this result follows from Tikhonov's Theorem (1) and is fully laid out in multiple places in the literature [58, 7, 103]. □

Now, we can note that due to the rescaling, $A_{11} \in \mathcal{O}\left(1\right)$, but also that $dA_{12}A_{22}^{-1}A_{21} \in \mathcal{O}\left(d\right) < \mathcal{O}\left(1\right)$. Thus, as Assumption 4 is much sufficiently satisfied, the contribution from the inter-area relative dynamics becomes more separated from the slow aggregate motion. Thus, we can further propose a reduced slow system of,

$$\frac{dy_s}{dt_s} = A_{11} y_s + \mathcal{O}(d), \quad y_s(0) = y(0). \tag{6.6}$$

Substituting the definition of A_{11} into (6.6), yields,

$$\dot{y}_s = -M^{-1} L(\mathcal{G})\, y_s,$$

which we can observe is identical to the agent dynamics in (6.1), with the diagonal matrix of area agent numbers taking the place of the time scale parameters. Thus, if we are interested in assessing characteristics of the system in (6.1), via Theorem 9, we can consider a lifted graph with the appropriate backbone topology, and number of agents per area determined by the associated time scale parameters in the scaled consensus problem of interest. We can note, however that in (6.1), we did not specify that ϵ_i had to be greater than unity, or an integer. We address these concerns in the following remark.

Remark 13. *The dynamics in* (6.1) *can always be considered on a reference time scale,*

$$t_r = \frac{t}{\epsilon_r}.$$

Expressing the scaled consensus dynamics derivatives with respect to this time scale is equivalent to dividing each agent time scale parameter by ϵ_r*. Thus, we can always choose to evaluate a scaled consensus system on a faster reference time scale (thus,* $\epsilon_r < \epsilon_i \forall i$*), which results in the equivalent areas in the lifted graph having more than one node. Equivalently, this could also be interpreted as uniformly applying an edge weighting of* $(1/\epsilon_r)$ *to all edges in* $\mathcal{G}$*.*

While the resulting effective time scale parameters are still not ensured to be integer values (and thus correspond to a particular raised graph exactly), we can see that these discrepancies will be small.

A consequence of choosing a reference time scale under which to analyse (6.1) is that we can always choose ϵ_r small enough that d is sufficiently small such that (6.6) has arbitrarily

small error. However, the trade off is clear; improving the approximation comes at the cost of having to consider potentially large number of nodes in the lifted graph.

Finally, we must note that to be completely in line with (6.1), the output from (6.6) must be the aggregate node states,

$$h_s(t) = y_s = Cx,$$

which we can see is similar to the full-state output, but in the lifted case requires a priori knowledge of the areas and total number of agents in each.

6.1.4 Choice of Atomic Graph

In the previous section, from the beginning of the consideration of the lifted graphs, we considered each area to be a complete graph with a single node with external connections. This naturally raises the question of whether or not the "atomic" topology of the area matters in the ability of the lifted graph to approximate the scaled system and vice versa. The question is whether the atomic graph choice has any bearing on a lifter graph's ability to reproduce the consensus value of the scaled consensus model. From [83], we know that (6.1) will converge to,

$$x_c = \frac{\omega^T x(0)}{\omega^T \mathbf{1}}. \tag{6.7}$$

where $w = [\epsilon_1, \ldots, \epsilon_n]$ (the vector of time scale parameters). Now consider (6.1) after transforming to a suitably fast reference time scale, and consider the replacement of a slow node i with m nodes (connected, but assuming no specific graph topology), all with an identical initial value c. Without loss of generality, let $i = n$. So, we seek a choice of m and c that will allow us to recover the value given by (6.7). With $i = n$, we have,

$$\tilde{\omega} = [1, \sigma_1, \ldots, \sigma_{n-1}, \underbrace{1, \ldots, 1}_{m}]^T$$

$$\tilde{x}_0 = [x_0(0), x_1(0), \ldots, x_{n-1}(0), \underbrace{c, \ldots, c}_{m}]^T,$$

where $\sigma_i = \epsilon_i/\epsilon_r$. The consensus value of this new network is given by applying (6.7),

$$\tilde{x}_c = \frac{\tilde{\omega}^T \tilde{x}_0}{\tilde{\omega}^T \mathbf{1}} = \frac{\omega^T x_0 - \sigma_n x_n(0) + mc}{\omega^T \mathbf{1} - \sigma_n + m} \tag{6.8}$$

From (6.8) we can see that if we take $m = \sigma_n$ and $c = x_n(0)$, then $\tilde{x}_c = x_c$. However, m must be an integer, and even though $\sigma_i > 1$ is ensured, it is not a good assumption that all scaling parameters are integer multipliers of each other (though, we can see from (6.8) that in that case the consensus value of the approximate network reproduces that original consensus value exactly). Furthermore, if m is chosen to be some integer approximation to σ_n, then from (6.8), c can be chosen to compensate for this approximation to minimize the error between x_c and $\tilde{x}_c$. Thus, we can see that the topology of the atomic graph is not important for recovering the consensus value. The number of nodes in the atomic graph, however, is determined by the scale of the node being approximated.

As the consensus value can be recovered solely by ensuring that each atomic graph has the correct number of constituent agents, we must consider the dynamical response of an arbitrary agent in order to differentiate between candidate atomic graphs. Consider the response to a constant unity control signal for a scaled single integrator, $\sigma_i x_i' = u_i$. We know that the spectra of a graph Laplacian for a complete graph has $n-1$ repeated eigenvalues of value n. The rate of convergence (and hence the rate at which the graph agents coalesce about the unity input) is governed by $\lambda_2(\mathcal{L})$. For the complete graph, we know $\lambda_2 = n$, which matches the interpretation of the reduced slow model in the previous section. Thus, the complete graph is the natural choice for the atomic graph.

Finally, as a sanity check, we can see in Figure 6.3, the complete graph (where the input and output nodes are not the same node in the atomic graph) mimics the slow response

very well. Based on this, we will choose the complete graph with number of nodes chosen to match the level of scaling as our atomic graph for approximating the slow nodes.

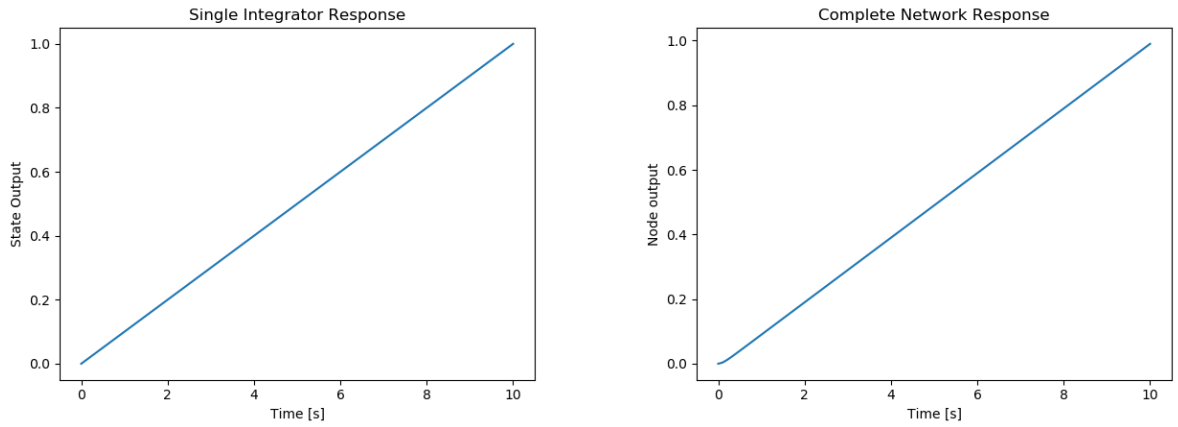


Figure 6.3: Comparison between scaled single integrator dynamics (left) and complete graph (right) to unity input.

Supporting Simulations - No Input Case

Here we present a simple example to show that the approximate graph (with complete components replacing slow nodes) is a valid approximation of the full consensus dynamics. Consider a path graph on four nodes with three distinct time scale parameters, which we display in Figure 6.4.

The fastest time scale is $\tau = t/0.1$, so after transforming the system to that time scale, we can see that the "blue" and "green" nodes should be approximated by K_5 and K_{10}, respectively. The mono-scale approximation graph is shown in Figure 6.5.

Simulating consensus for both the multi-scale graph and the mono-scale approximation can be seen in Figure 6.6. We can see that we recover the consensus value (as this example satisfies the condition at the scales are integer multiples of each other), as well as approximating the individual node dynamics (the aggregate of the approximating graph nodes tracks

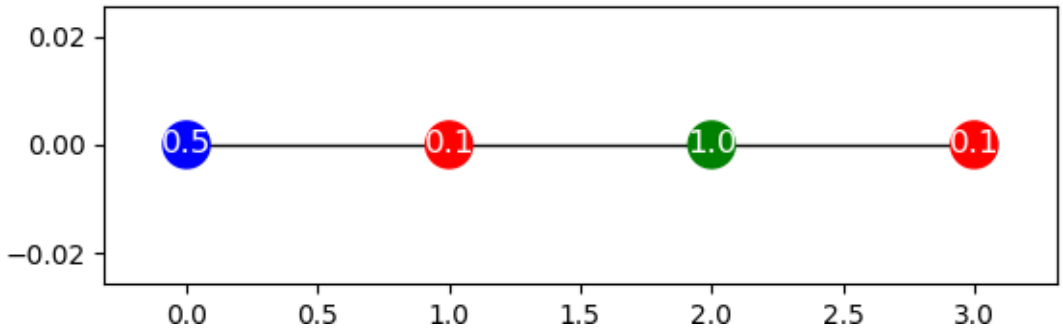


Figure 6.4: Original path graph. Time scale parameters are labeled in white on individual nodes.

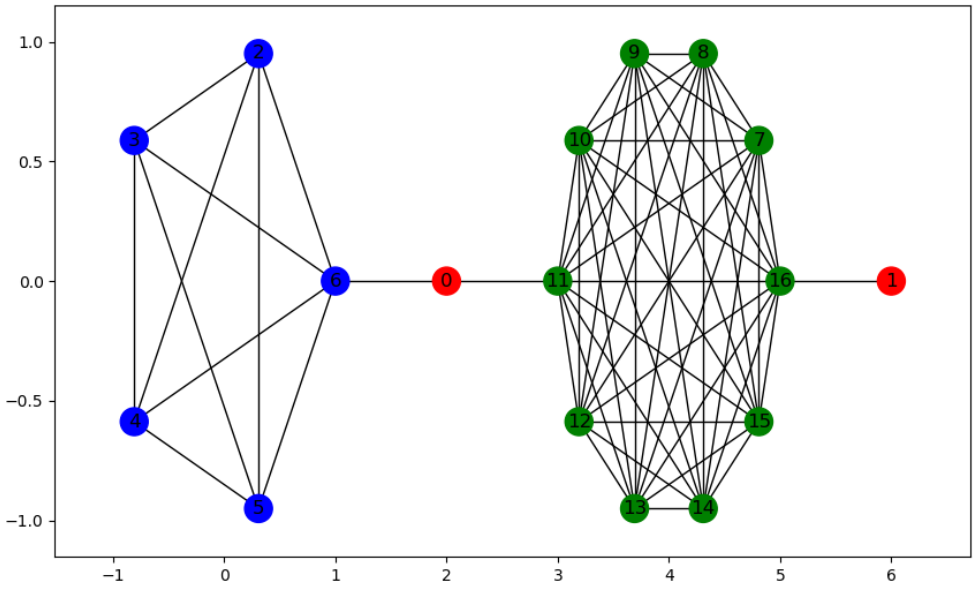


Figure 6.5: Approximation path graph.

the exact node trajectory with bounded error).

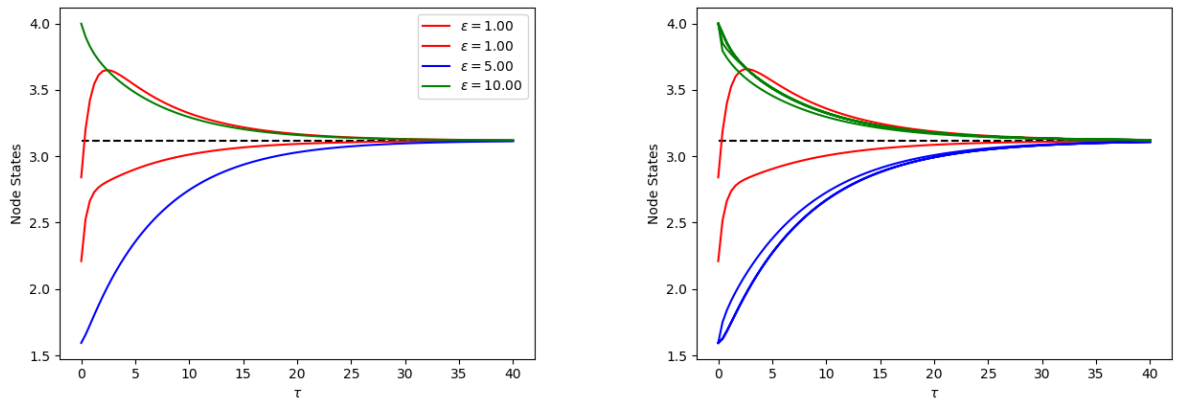


Figure 6.6: Comparison between multi-scale consensus (left), and mono-scale approximation (right). Colors denote the time scale/node approximation group from previous figures.

Input to the Lifted Graph

Now, we would like to extend the thrust of the previous sections and consider the case where we have extraneous input being injected into one or more native nodes, such that the resulting reduced order model will correspond to (6.2). To do this, consider a lifted graph and apply the transformation as before,

$$\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} C \\ Q \end{bmatrix} x$$
$$x = \begin{bmatrix} U & Q^* \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix},$$

which gives,

$$\begin{bmatrix} \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} + \begin{bmatrix} CB \\ GB \end{bmatrix} u(t). \tag{6.9}$$

In the same process as the case without input, we can scale the constituent matrices, resulting the definition of fast and slow time scales. Then, the fast inter-area dynamics can be solved for the slow manifold, which results in a reduced slow model of,

$$\frac{dy_s}{dt_s} = (A_{11} + A_{12}A_{22}^{-1}A_{21})y_s + (A_{12}A_{22}^{-1}Q + C)Bu. \tag{6.10}$$

Again, we can note that if we would like the reduced order model to have full state output for the aggregate nodes, the output of the lifted graph will be an aggregate of all nodes,

$$h_l(t) = Cx,$$

Finally, we can apply the order arguments from the previous section to further reduce the reduced slow system to that of a full separation principle, resulting in,

$$\begin{aligned} \dot{y}_s &= M^{-1}A\mathcal{G}, ly_s + M^{-1}U^TBu \\ h(t) &= y_s, \end{aligned}$$

which, again, corresponds to a lifted graph with dynamics and output,

$$\begin{aligned} \dot{x} &= -A(\mathcal{G}', l)x + Bu \\ h(t) &= Cx. \end{aligned} \tag{6.11}$$

For a desired U^TB in the scaled formulation, we can choose an appropriate B for the lifted graph.

Performance Calculation

A relevant question, now, is the approximation of system-level metrics for the scaled system using the mono-scale lifted graph dynamics. In particular, we are interested in the $\mathcal{H}_2$ performance, due to its relevance in networked dynamics disturbance rejection and network security considerations. It is an open question how the $\mathcal{H}_2$ performance for consensus systems

influenced by arbitrary signals changes in the presence of time scales, thus, we seek insight from the lifted case that is applicable to the scaled case.

The $\mathcal{H}_2$ performance for the lifted graph is given by,

$$\mathcal{H}_2^2 = \operatorname{tr}\left(\int_0^\infty C\exp\left(-A(\mathcal{G},l)\tau\right)BB^T\exp\left(-A(\mathcal{G},l)\tau\right)C^T\mathrm{d}\tau\right) \tag{6.12}$$

For mono-scale consensus systems with full-state output, the symmetry of $A(\mathcal{G},l)$ along with the ability to permute the matrix operations with the trace operation allows for the performance to be reformulated as,

$$\mathcal{H}_2^2 = \operatorname{tr}\left(BB^T\int_0^\infty \exp\left(-2A(\mathcal{G},l)\tau\right)\mathrm{d}\tau\right)$$

which has analytic solutions, or at least a form that can be manipulated algebraically, even for arbitrary inputs and graph topologies. In (6.12), the only way to achieve a similar result is to have $B = I$, which is not an interesting case, or especially realistic, as it also corresponds to input to all aggregate nodes in the reduced models.

What we can see from (6.12) is that the integrand of the performance can be permuted so that the either the input or output matrices are between the matrix exponentials of a symmetric grounded laplacian. Depending on the form of those input/output matrices, either analytic forms of the performance (or bounds) can be made - however, without a strong motivating example or problem informing the decision to select a form for the B or C matrices, in many cases these results are potentially not useful for any realistic system.

6.2 Multi-valued Saturated Actuators with Consensus Inputs

In any physical system, the actuator that is exerting the control effort will be subject to either physical or practical limits. For example, when we seek to control the attitude of a satellite, we are accomplishing angular velocity control via applying torques to the body of the spacecraft, either via some sort of magnetorquer, reaction wheel system, or thruster. Each of those actuators, though, will have a minimum and maximum torque it can apply; the

magnetorquer is limited by the available power and magnitude of the ambient magnetic field, the reaction wheel has a maximum angular velocity and fixed mass, and the thrusters have a nominal thrust value (that is also dependent on the availability of propellant and degrades over repeated use). Thus, it behooves us to consider formulations that impose limits on the control input to a system.

A common way of accomplishing this is to simply subject any control law output to a saturation function with limits that are appropriate for the system in question. However, in most applications in the literature, this limit is a single-valued limit for all inputs into the system [68, 74, 114, 115, 113]. Another method, though not one we explore here, is to define the control input as a non-linear consensus function (that is, state differences as the argument of a nonlinear function [84]) that naturally saturates. Typically a hyperbolic tangent function has been employed for this purpose [101]. This approach has the appeal of smoothly approaching the saturation limit, however, comes at the cost of being explicitly nonlinear in all regimes as well as only asymptotically approaching the maximum control signal as the non-saturated signal approaches infinity.

For heterogeneous multi-agent systems, the control limitations could vary from agent to agent, and imposing a limit based on the most constrained agent is leaving control authority on the table for the other agents. Thus, we would like to consider formulations that allow us to apply multi-level actuator saturation limits on agents. What we will see is that for consensus based inputs, the resulting saturation function relies on a scaled Laplacian matrix, allowing us to use the properties we've found in investigations of clock-biased consensus systems to show some basic properties of systems under multi-level saturation (where, here the scaling will be due to agent-based saturation limits, instead of agent-based clocks).

To begin, let us define the multi-level saturation function that we will be subjecting the control inputs considered in later discussions to. For a constant, positive scalar ν, and scalar

u,

$$\sigma_\nu(u) = \begin{cases} -\nu, & u < -\nu \\ u, & -\nu \leq u \leq \nu \\ \nu, & u > \nu \end{cases} \tag{6.13}$$

Furthermore, we will extent this notation to denote the application of $\sigma_{(\cdot)}(\cdot)$ to vector quantities. For $u \in \mathbb{R}^n$, and $V = \text{diag}(\nu_1, \ldots, \nu_n)$, $\sigma_V(u) = [\sigma_{\nu_1}(u_1), \ldots \sigma_{\nu_n}(u_n)]$. Thus, this function in the application to vector quantities allows for each entry in a vector to be bounded by a potential unique limit with respect to the remaining entries.

With that definition in hand, we'll consider the cases of single and double integrator agent dynamics and in each case show that the saturated consensus dynamics will still converge under the presence of multi-level saturation.

6.2.1 Single Integrator Agent Dynamics

The simplest case of saturated consensus-based input is the single integrator dynamics case, which for the unsaturated case was derived in Chapter 2 as (2.8) but is restated here for completeness,

$$\dot{x} = -\mathcal{L}x.$$

The focus here is to consider the case where we would like to apply inputs to a systems of the form (2.8), but have symmetric actuator limits which are potentially unique to each agent,

$$-\nu_i \leq u_i \leq \nu_i$$

where ν_i is a known, positive scalar that corresponds to the input limits for agent i, which we codify in the following assumption.

Assumption 5. *All agent specific saturation levels are strictly positive, $\nu_i > 0$ $\forall i \in$.*

We can impose the assumed actuator limitations by wrapping the consensus dynamics (2.8) in the multi-level saturation function $\sigma(\cdot)$,

$$\dot{x} = \sigma_V\left(-\mathcal{L}x\right) \tag{6.14}$$

where $V = \text{diag}\left(\nu_i, \ldots, \nu_n\right)$, and as discussed in the previous section, we are taking the application of $\sigma(\cdot)$ to a vector to signify entry-wise application of the function. In the following proposition, we show that this formulation is equivalent to a formulation utilizing the more common unity-saturation function.

Proposition 18. *For saturation limits on each agent's input given by ν_i with $V = diag(\nu_i, \ldots, \nu_n)$, the multi-level saturation problem (6.14) is equivalent to,*

$$\dot{x} = V\, sat\left(-V^{-1}\mathcal{L}x\right) \tag{6.15}$$

where $sat(\cdot)$ is the unitary saturation function.

Proof. Let $z = \mathcal{L}x$, and consider (6.15) entry-wise. It follows directly that $[V^{-1}z]_i = z_i/\nu_i$, thus,

$$\dot{x}_i = [V\text{sat}(V^{-1}z)]_i = \nu_i \text{sat}\left(\frac{z_i}{\nu_i}\right).$$

Now, consider the available possibilities for the value of z_i and the resulting saturation, or lack thereof. If $|z_i/\nu_i| \leq 1$ $(z_i \leq \nu_i)$, then $\text{sat}(z_i/\nu_i) = z_i/\nu_i$ corresponding to $\dot{x}_i = z_i = -\mathcal{L}x$. If $|z_i/\nu_i| > 1$ $(z_i > \nu_i)$, then $\text{sat}(z_i/\nu_i) = \text{sign}(z_i)1$, corresponding to $\dot{x}_i = \text{sign}(z_i)\nu_i$. Compiling these scenarios gives,

$$\dot{x}_i = \begin{cases} -\nu_i & z_i < -\nu_i \\ z_i & -\nu_i \leq z_i \leq \nu_i \\ \nu_i & \nu_i < z_i \end{cases}$$

which is the exact form that we expect from (6.14). □

With the previous proposition in hand, we will now establish the basic convergence property of saturated consensus problems holds in the multi-level saturation case. To begin, we establish our base assumptions as to the nature of the coupling topology of our agents.

Assumption 6. *Zero is a single eigenvalue of the Laplacian matrix representing the communication coupling between agents.*

This assumption can be interpreted as the graph having a single connected component and, in the case of a directed communication topology, that there is a spanning tree of $\mathcal{G}$. With this assumption, and the previous proposition, we can now state the following result.

Theorem 10. *The multi-level saturated consensus problem (6.15) can be globally solved if and only if Assumption 6 is satisfied. That is, for $V \succ \mathbf{0}$ (Assumption 5), $x(0) \in \mathbb{R}^n$,*

$$\lim_{t\to\infty} x(t) = x_c\mathbf{1},$$

where $x_c \in \mathbb{R}$ is a constant in the convex hull of $x(0)$, if and only if the underlying graph is connected (granting a single zero eigenvalue).

Proof. To begin, we can note that as stated, Theorem 10 is satisfied for the case,

$$\dot{x} = \nu_s \text{sat}\left(-w\mathcal{L}x\right)$$

where $\nu_s \in \mathbb{R}$ is a single saturation level, and $w > 0$ is a uniform edge weighting on the edges in $\mathcal{G}$ [68, Theorems 1 & 2]. We will rely on this previous result to bound the convergence of the multi-level saturated system from above and below. To begin, we can note that,

$$\text{sat}\left(V^{-1}\mathcal{L}x\right) \leq \text{sat}\left(\nu_{\min}^{-1}\mathcal{L}x\right).$$

where the $\leq$ is applied entry-wise to the vectors in question. To see this, let $z := V^{-1}\mathcal{L}x$. Entry-wise we have,

$$\begin{aligned} z_i &= \frac{1}{\nu_i}[\mathcal{L}]_{ii}x_i + \sum_{j\sim\mathcal{N}(i)} \frac{[\mathcal{L}]_{ij}}{\nu_i}x_j \\ &= \frac{1}{\nu_i}\left([\mathcal{L}]_{ii}x_i + \sum_{j\sim\mathcal{N}(i)} [\mathcal{L}]_{ij}x_j\right) \\ &\leq \frac{1}{\nu_{\min}}\left([\mathcal{L}]_{ii}x_i + \sum_{j\sim\mathcal{N}(i)} [\mathcal{L}]_{ij}x_j\right) = \frac{1}{\nu_{\min}}[\mathcal{L}x]_i. \end{aligned}$$

Note that applying the saturation function to z_i does not change this inequality, as if $1/\nu_{\min}[\mathcal{L}x]_i$ is not saturating, then so is z_i. If $1/\nu_{\min}[\mathcal{L}x]_i$ is saturating, then z_i is either not saturating (and the inequality holds), or it is saturating (and the equality holds). By the same logic, we also have, $\text{sat}\left(\nu_{\max}^{-1}\mathcal{L}x\right) \leq \text{sat}\left(V^{-1}\mathcal{L}x\right)$, leading to

$$\text{sat}\left(\nu_{\max}^{-1}\mathcal{L}x\right) \leq \text{sat}\left(V^{-1}\mathcal{L}x\right) \leq \text{sat}\left(\nu_{\min}^{-1}\mathcal{L}x\right). \tag{6.16}$$

Now, under Assumption 6, we have that for any $q \in \mathbb{R}^n$,

$$\nu_{\min} q \leq Vq \leq \nu_{\max} q \tag{6.17}$$

where, again, the inequalities are assessed entry-wise. These follow directly from the multiplication by diagonal V being an entry-wise weighting of q. Combining (6.16) and (6.17), for any $x \in \mathbb{R}^n$ gives,

$$\begin{aligned}\underbrace{\nu_{\min}\text{sat}\left(\frac{1}{\nu_{\max}}\mathcal{L}x\right)}_{:=\underline{\dot{x}}} &\leq V\text{sat}\left(V^{-1}\mathcal{L}x\right) \\ &\leq \underbrace{\nu_{\max}\text{sat}\left(\frac{1}{\nu_{\min}}\mathcal{L}x\right)}_{:=\bar{\dot{x}}}.\end{aligned}$$

Now, they systems of $\underline{\dot{x}}$ and $\bar{\dot{x}}$ are single-saturation limit systems for uniformly weighted inputs, which from [68, Theorems 1 & 2] are proven to synchronize. Thus, since $\dot{x}$ for the multi-level saturation limit system that we are considering is bounded both above and below by convergent systems, it must also converge. □

Remark 14. *In [68, Theorem 1], the method of proving the solvability of the single-level saturation problem can be summarized as follows: an invariant set (Ω) equivalent to the region for which the input is saturating is defined. Within Ω, the problem is unsaturated, so a solution under Assumption 6 is ensured. It is then shown that trajectories from any initial condition outside of Ω enter the set in finite time. This method cannot be applied analogously to the multi-level saturation problem because for any specified set, the boundary can be*

rendered repulsive by selecting initial conditions based on the ratios of saturation parameters of neighboring agents, rendering it not useful in the aforementioned proof structure.

Next, we consider a slightly more complex set-up, where the agent dynamics are identically equal to double integrators.

6.2.2 Double Integrator Agent Dynamics

In the case of each agent possessing double-integrator dynamics, we can assume the following agent dynamics.

Assumption 7. *The system matrices are given by the following forms,*

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Furthermore, denote the substates of each agent states as,

$$x_i = \begin{bmatrix} r_i \\ q_i \end{bmatrix}$$

Under the above assumption, the individual agent dynamics are given by,

$$\dot{r}_i = q_i$$
$$\dot{q}_i = u_i$$

where u_i will be subject to a multi-level saturation function $\sigma_V(\cdot)$, that is, $\dot{q}_i = \sigma_{\nu_i}(u_i)$. The leader dynamics are taken to be an uninfluenced double integrator, that is, $\dot{x}_l = Ax_{l,0}$, which is equivalent to $\dot{r}_l = q_{l,0}$ and $\dot{q}_l = 0$. Under Assumptions 5 and 6, we propose the following distributed control input to bring the double integrator system into consensus with the leader agent,

$$u = A(\mathcal{G}, l)\tilde{r} + A(\mathcal{G}, l)\tilde{q} \tag{6.18}$$

where $A(\mathcal{G}, l)$ is the grounded Laplacian matrix describing the communication network between agents as well as taking into consideration the connection to the leader node by the

followers, and $\tilde{r}_i = r_i - r_l$ and $\tilde{q} = q_i - q_l$ are measures of each agents' substates relative to the leader's states.

Remark 15. *The definition of the relative variables $\tilde{r}$ and $\tilde{q}$ at first seem to imply that each agent requires knowledge of the leader's state in order to compute its control input. However, considering 6.18 component-wise shows that for follower agents not directly connected to the leader, the factors of r_l and q_l cancel out, resulting in an input that depends solely on local information.*

Theorem 11. *Under Assumptions 5, 6, and 7, the control input (6.18) achieves global consensus between the follower and leader states. That is,*

$$\lim_{t\to\infty} \tilde{r}_i(t) = 0$$
$$\lim_{t\to\infty} \tilde{q}_i(t) = 0$$

for all $i = 1, \ldots, n$, where $\tilde{r}_i(t) = r_i(t) - r_l(t)$ and $\tilde{q}(t) = q_i(t) - q_l(t)$.

Proof. First, define an auxiliary function,

$$\phi_i(s) = \int_0^s \sigma_{\nu_i}(t)\mathrm{d}t$$

which is useful for the form of its derivative. Now, consider the following Lyapunov function candidate,

$$\begin{aligned} V(\tilde{r}, \tilde{q}) = \tilde{q}^T A(\mathcal{G}, l)\tilde{q} + \sum_{i=1}^{n} \phi_i \left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j \right) \\ + \sum_{i=1}^{n} \phi_i \left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j - \sum_{j=1}^{n} a_{ij}\tilde{q}_j \right) \end{aligned} \tag{6.19}$$

where $a_{ij} = [A(\mathcal{G}, l)]_{ij}$. From Assumption 7 and (6.18), we can see that $(\tilde{r}, \tilde{q}) = (0, 0)$ is an equilibrium, and as $\tilde{r}$ or $\tilde{q} \to \infty$, $V \to \infty$, so we have the origin as an equilibrium along with radial unboundedness. Thus, we will prove the convergence of the proposed by verifying that the candidate function is negative definite along all trajectories.

$$\begin{aligned}\dot{V} &= 2\tilde{q}^T A(\mathcal{G},l)\sigma_V(u) + \sum_{i=1}^{n}\left(-\sum_{j=1}^{n} a_{ij}\tilde{q}_j\right)\sigma_V\left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j\right) \\ &+ \sum_{i=1}^{n}\left(-\sum_{j=1}^{n} a_{ij}\tilde{q}_j\right)\sigma_V(u_i) - \sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_V(u_i)a_{ij}\sigma_V(u_j) \\ &= -\sigma_V^T(u)A(\mathcal{G},l)\sigma_V(u) + \sum_{i=1}^{n}\left(-\sum_{j=1}^{n} a_{ij}\tilde{q}_j\right) \\ &\times\left(\sigma_V\left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j\right) - \sigma_V\left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j - \sum_{j=1}^{n} a_{ij}\tilde{q}_j\right)\right)\end{aligned}$$

Now, from the properties of the Laplacian under Assumption 6, the grounded Laplacian, $A(\mathcal{G},l)$ is positive definite, thus, $-\sigma_V^T(u)A(\mathcal{G},l)\sigma_V(u) < 0\ \forall \tilde{r},\tilde{q}$, and $\sigma_V^T(u)A(\mathcal{G},l)\sigma_V(u) = 0$ implies $\tilde{r},\tilde{q} = 0$. Next, from the monotonicity of $\sigma_V(\cdot)$, we can see that

$$\left(\sigma_V\left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j\right) - \sigma_V\left(-\sum_{j=1}^{n} a_{ij}\tilde{r}_j - \sum_{j=1}^{n} a_{ij}\tilde{q}_j\right)\right) < 0$$

when $\left(-\sum_{j=1}^{n} a_{ij}\tilde{q}_j\right) > 0$ (and thus the product is < 0), and vice versa. Thus, $\dot{V} < 0$ and $V(\tilde{r},\tilde{q}) = 0$ only at $\tilde{r} = 0$ and $q \tilde{=} 0$, which by LaSalle's Invariance Principle gives,

$$\lim_{t\to\infty}\tilde{r}(t) = 0,\ \lim_{t\to\infty}\tilde{q}(t) = 0$$

which indicates global convergence as desired. □

6.2.3 Section Summary

In this section, we have briefly considered the convergence of single and double integrator consensus systems under multi-level actuator saturation. We observed that the saturation function had a scaled Laplacian matrix at its heart, and thus were able to use properties of the scaled consensus problem to prove the convergence of single and double integrator consensus systems.

6.3 Clock Biased Centrality Measures

In several of our main results, we have observed instances where the performance results for scaled graphs suggests a connection to notions of network centrality. In Lemma 4 of Chapter 4.2, the $\mathcal{H}_2$ performance on tree graphs was related to the degree-centrality weighted with the time scale parameters. Similarly, in Proposition 16 of Chapter 4.4, the average state cost on tree graphs contained terms that resembled the closeness centrality. This is not necessarily surprising, as previous work for mono-scale network performance can be explicitly linked to the information centrality [19, 15, 14]. As we'll discuss further later in this section, the notion of information centrality is equivalent to the effective resistance for a resistor network representation [9, 27], so centrality is also at the core of linking performance of mono-scale networks and alternative interpretations of networked consensus systems.

Notions of centrality in networks originated in studies of social networks, where there was the desire to identify the most influential people (nodes) in a social network. Freeman and Nieminen pioneered the early formulations of *degree*, *betweenness*, and *closeness* centrality [38, 39, 40, 81]. More recently, Brandes and Fleiescher [9] rigorously derived the closeness centrality for networks in terms of current flow over an equivalent resistor network, which explicitly linked the effective resistance to centrality. These measures, outside of effective resistance, were for the most part used for identification purposes - that is, the dynamics of the networks such concepts were being applied to were often not considered. This was the driving motivation for the concept of *dynamic influence* which considers the dynamics of a system driven by "collective dynamics" and seeks to quantify the influence of a node on the overall network dynamics, namely, the equilibrium of the system [61]. The resulting measure is closely related to eigenvector centrality, which is another centrality measure derived from considering the dynamics of the nodes within the network. Somewhat surprisingly, based on the amount of importance placed on the concepts of centrality in social networks, applications of centrality for purposes of controlling or influencing a dynamic network are fairly sparse in the literature. Notable exceptions include[19, 15, 27], which identified a

metric of performance that is equivalent to the information centrality (though this was presented as being related to effective resistance). Extended notions of centrality were also employed in [32, 33] to select groups of leader nodes based on information centrality. The non-information centrality measures, however, have not enjoyed as much use or analysis in the literature, however, and as we expand on in this section, are potentially an avenue for additional research.

Finally, the motivation for considering that centrality will be impacted by the existence of agent clock/time scale parameters is sufficiently captured by Borgatti [8] as he discusses the various interpretations and formulations of centrality: "*The one thing that all agree on is that centrality is a node-level construct.*" Agent-based clocks or scaling are, by definition, node-level characteristics within the network, so it should follow that centrality will be impacted by the introduction of this additional node-level construct. Thus, understanding how nodal parameters alter the notions of centrality offers both an extension of previous work that formulates specific definitions of centrality as well as work that applies centrality to analyzing or controlling networked systems to scaled formulations.

In the sections that follow, we summarize the current definitions of the most common centrality measures, and offer extensions or interpretations to scaled graphs where applicable.

6.3.1 Degree Centrality

The degree of a node in a network can be viewed as important as an index of its potential for communication activity - the more neighbors a node has, the more widely it can share its state and influence those around it [40]. The measure was first formally introduced in [81] and is simply the degree of a node within an undirected graph,

$$C_D(i) = \sum_{j=1}^{n} [A_d]_{ij}.$$

These early works also proposed a normalized version of this measure,

$$C'_D(i) = \frac{\sum_{j=1}^{n}[A_d]_{ij}}{n-1},$$

which is an acknowledgement of the fact that $C_D(i)$ alone is dependent on the size of the network; networks over larger node sets allow for potentially larger $C_D(i)$. The normalized quantity takes into account that a node can be connected to at most $(n-1)$ other nodes in the network. This allows for comparison of degree centrality between graphs of various sizes if the need arises.

Recasting with scaling matrix: From the definition of the scaled Laplacian matrix, we can defined a scaled degree matrix/adjacency matrix,

$$E^{-1}L(\mathcal{G}) = E^{-1}\left(\Delta_{\mathcal{G}} - A_{\mathcal{G}}\right)$$
$$\tilde{A}_{\mathcal{G}} := E^{-1}A_{\mathcal{G}}$$
$$\tilde{\Delta}_{\mathcal{G}} := E^{-1}\Delta_{\mathcal{G}}$$

combining this with the above, non-normalized degree centrality measure, gives

$$\tilde{C}_D(i) = \sum_{j=1}^{n}[\tilde{A}_{\mathcal{G}}]_{ij} = \sum_{j=1}^{n}\frac{[A_{\mathcal{G}}]_{ij}}{\epsilon_i} = [\tilde{\Delta}_{\mathcal{G}}]_{ii}$$

The normalized version, however, presents an interesting wrinkle. What is the maximum adjacency of a node in a scaled graph? Let $\epsilon_{\min}$ represent the smallest scaling parameter present in a scaled graph. If the node with the smallest scaling is attached to every other node in the graph, then its scaled adjacency is $(n-1)/\epsilon_{\min}$, which is the maximum adjacency possible for that network. So we could propose a normalized version as,

$$\tilde{C}'_D(i) = \frac{\sum_{j=1}^{n}[\tilde{A}_{\mathcal{G}}]_{ij}}{\frac{n-1}{\epsilon_{\min}}} = \frac{\sum_{j=1}^{n}\frac{\epsilon_{\min}}{\epsilon_i}[A_{\mathcal{G}}]_{ij}}{n-1}$$

So we can see that the $(n-1)$ factor in the denominator normalizes for the graph size, while the $\epsilon_{\min}/\epsilon_i$ factor normalizes the range of time scale factors.

6.3.2 Betweenness Centrality

Betweenness centrality was proposed [40] as a measure of the frequency of a node to fall between other pairs of nodes on the geodesic that connects them. This is a measure of centrality because nodes that fall on a large number of geodesics between surrounding nodes could influence the information that passes through them along the geodesics. This is known as either coordination (by adding to or beneficially repackaging information), or as gatekeeping (removing or restricting information flow). The mathematical definition relies on finding the total number of geodesics between two nodes, ν_i and ν_j in a graph. Denote the number of geodesics as g_{ij}. Consider the probability that information/a packet uses a geodesic to traverse the path between ν_i and ν_j is simply $1/g_{ij}$, that is, the information is equally as likely to take any of the available paths. Then, denoting the number of geodesics that ν_k lies on as $g_{ij}(\nu_k)$, the probability that ν_k is on the geodesic that the information is being transmitted over is,

$$b_{ij} = \frac{g_{ij}(\nu_k)}{g_{ij}}$$

which is the "partial betweenness" of ν_k found by considering nodes i, j. To find the full betweenness for node k, we sum the partial betweenness over all pairs of nodes in the graph:

$$C_B(\nu_k) = \sum_{i \neq k}^{n} \sum_{j \neq k}^{n} b_{ij}(\nu_k)$$

In an earlier paper [38], it was determined that the maximum value for $C_B(v_k)$ is achieved by the central node in a star graph, so the relative betweenness centrality can be given by,

$$C'_B(\nu_k) = \frac{2 \sum_{i \neq k}^{n} \sum_{j \neq k}^{n} b_{ij}(\nu_k)}{n^2 - 3n + 2}$$

Recasting with scaling matrix: The betweenness centrality measure defined above conveniently is agnostic of the choice of geodesic. That is, for the scaled network, one could imagine that longer path lengths (in number of edges) could have effectively shorter length

if the paths are composed of faster nodes - much akin to the total edge weight in a path on a weighted graph. However, exactly how we are defining the length/cost of a path does not impact the formulation of the betweenness centrality - it will effect the actual path on the network/which nodes end up having the highest centrality, but not the actual calculation.

Furthermore, the argument from [38] appears to hold - that is, even if the path length is a function of the scaling parameters, adding nodes to the central node of a star graph is the only way to consistently increase the betweenness measure for the central node. Since the central node is on the only path (and therefore the geodesic) between any two non-central nodes, it has the highest centrality of any of the nodes. Again, this is all just counting geodesics so is not altered by changing the scaling parameters.

So, given a path length function dependent on the scaling parameters, we can use the normal definition for calculating the betweenness centrality of the nodes. However, because the geodesics in the scaled graph are not likely to be the same as the mono-scale network, the most central nodes will change compared to the mono-scale network. It may be that there is little to be stated definitively and generally - it seems that for a length metric that is simply the time scale parameters instead of number of edges, nodes on "fast paths" will see an increase in betweenness measure, but it is likely it will all be very topology/scaling distribution dependent. Furthermore, if an application motivates the converse - a geodesic length metric based on some inverse of the time scale parameters, the nodes on slow paths would be preferred. Thus, the choice of length metric as a function of time scale parameter would need to be the focus of any research for this measure.

6.3.3 Closeness Centrality

Closeness centrality is the measure that bears the most resemblance to the average performance cost results presented in Proposition 16 of Chapter 4.4. It is defined in terms of the measure of a node's decentrality

$$C_c(\nu_k)^{-1} = \sum_{i=1}^{n} d(\nu_i, \nu_k)$$

where $d(\nu_i, \nu_k)$ is the length of the geodesic from ν_i to ν_k. This can be normalized by considering the minimum sum of distances in n-node graph is $n-1$ (for a node adjacent to every other node),

$$C_c'(\nu_k)^{-1} = \frac{\sum_{i=1}^{n} d(\nu_i, \nu_k)}{n-1}.$$

Inverting C_c' then gives the closeness centrality.

Now, let us explore how this measure can be related to results in graph performance under single inputs. In [15] the average performance cost for a tree graph, $\mathcal{T}$, is shown to be given by,

$$J_{\text{avg}}(\mathcal{T}, i) = \frac{1}{n} \sum_{j=1}^{n} d(\nu_i, \nu_j) + 1.$$

Re-arranging this gives,

$$\begin{aligned} J_{\text{avg}}(\mathcal{T}, i) &= \frac{1}{n} \left(\sum_{j=1}^{n} d(\nu_i, \nu_j) + n \right) \\ &= \frac{1}{n} \sum_{j=1}^{n} \left(d(\nu_i, \nu_j) + 1 \right) \end{aligned}$$

The quantity $d(\nu_i, \nu_j) + 1$ can be interpreted as $d(\nu_I, \nu_j)$ where ν_I is the external influencing node, attached to ν_i alone. So, consider the graph on $n+1$ nodes where v_I is a native node, denoted as $\hat{\mathcal{G}}$. Without loss of generality, we can order the nodes in $\hat{\mathcal{G}}$ such that $\nu_I = \nu_{n+1}$. Then,

$$\begin{aligned} J_{\text{avg}}(\mathcal{T}, i) &= \frac{1}{n} \sum_{j=1}^{n+1} d(\nu_I, \nu_j) \\ &= C_c'(\nu_I, \hat{\mathcal{G}})^{-1}. \end{aligned}$$

Thus we can see that the average performance cost for mono-scale graphs is identical to considering the closeness centrality of the graph with the influencing node taken as a native node. The extension of this result in Proposition 16 gives some insight into the definition of closeness centrality for multi-scale networks, which we discuss further below.

Recasting with scaling matrix: The closeness centrality is dependent on the length of the geodesic between pairs of nodes in the graph. Thus, applying it to a scaled graph will be dependent on how that length changes with the scaling. Thus, similar issues as those identified in the betweenness centrality discussion apply here. However, the most straightforward solution to this issue may be found by appealing to the digraph representation of the multi-scale graph. Given the directed edge weightings, there is a clear choice consistent with the literature for defining geodesics - by the paths with minimal total edge weighting. As the digraph representation is a strongly connected digraph, under the assumption that we are considering connected graph topologies, we also do not need to worry about the closeness centrality not being defined for pairs of nodes.

However, unlike in the case of the betweenness centrality, the normalization factor is not time scale independent, as it represents the minimum sum of paths in a graph - which now depends on the intermediate time scales along a path. The minimum case is still a node adjacent to all other nodes - but where the neighbors are weighted by the maximum time scale (so the path to each slow neighbor is $1/(\epsilon_{\max})$, which is the minimum edge weight in the digraph): so the minimum sum of paths for a node adjacent to all other nodes is $((n-1)/\epsilon_{\max})$ for a n-node graph, giving,

$$\tilde{C}_c'(\nu_k)^{-1} = \frac{\epsilon_{\max} \sum_{i=1}^{n} d(\nu_i, \nu_k)}{(n-1)}$$

Weighted sum of closeness measures: Since for the average cost we only care about the trace of the grounded inverse, we can note the following. Let,

$$\alpha = \text{diagonal}(A(G,i)^{-1})$$
$$\omega = \text{diagonal}(E)$$

where diagonal() corresponds to the vector of the diagonal entries in the argument matrix. Then,

$$\mathbf{tr}(A(G,i)^{-1}E) = \alpha^T\omega$$

So we can see that in the mono-scale case we have just a sum of relative centrality measures (the shortest paths), but instead of changing the "centrality" of a node, the scaling is simply causing this sum to be weighted differently. We can also note with interest, that as we will see in the discussion of dynamic centrality, a valid question is how node scales influence some measures of dynamic centrality, but that here we explicitly cast the time scale parameters as weightings in the definition of path lengths. It may be that this interpretation is a bit nicer than trying to find a good "dynamic" centrality measure to reshape with the addition of time scale, though it remains an open avenue to determine whether there are motivating problems that give rise to other interpretations of the parameters for dynamic centrality measures.

6.3.4 Information Centrality

In [9], it's shown that the closeness centrality from a current flow perspective is identical to the information centrality. From the fact that the average performance cost on the (unscaled) tree graph is given by closeness centrality, it is possible that the average cost for non-tree graphs could be given by the information centrality. Before we explore how information centrality may be altered in the case of time scale parameters, let us first introduce effective resistance.

Relating effective resistance to the grounded Laplacian: Let $\mathcal{G}$ be a graph on n nodes in question; we do not make any assumptions as to its topology. Now, influencing node ν_i is equivalent to considering the graph $\hat{\mathcal{G}}$ which has ν_{inf} as a native node, along with all nodes/edges from $\mathcal{G}$. $\hat{\mathcal{G}}$ is obviously a graph on $n+1$ nodes, so without loss of generality, we can let the influencing node be numbered $n+1$. The definition of effective resistance can be found from the fact that a vector of voltages, $\hat{v} \in \mathbb{R}^{n+1}$, will satisfy the following equation,

$$L(\hat{G})\hat{v} = \hat{e}_i - \hat{e}_j$$

where $\hat{e}_k \in \mathbb{R}^{n+1}$ are the Euclidean basis vectors. Taking the pseudo-inverse of the graph Laplacian can then give the unique solution for the voltages:

$$\hat{v} = L(\hat{\mathcal{G}})^{\dagger}(\hat{e}_i - \hat{e}_j)$$

For an applied current of 1 amp, the effective resistance between nodes i and j is simply the difference in voltages:

$$\begin{aligned} R_{ij} &= \hat{v}_i - \hat{v}_j \\ &= (\hat{e}_i - \hat{e}_j)L(\hat{\mathcal{G}})^{\dagger}(\hat{e}_i - \hat{e}_j) \end{aligned}$$

Now, consider that we would like to consider the case where we ground the influencing node. Let $v,\ e_i,\ e_j \in \mathbb{R}^n$ be equal to $\hat{v},\ \hat{e}_i,\ \hat{e}_j$ with the $n+1$th index removed. Obviously, if $L(\hat{\mathcal{G}})\hat{v} = \hat{e}_i - \hat{e}_j$, $A(\mathcal{G},k)v = e_i - e_j$, where $A(\mathcal{G},k)$ is the grounded Laplacian with influencing node attached to node k. Since $A(\mathcal{G},k)$ is invertible (assuming connected $\mathcal{G}$), we don't need to use the pseudo-inverse, and the effective resistance between any nodes in $\mathcal{G}$ is given by:

$$R_{ij} = (e_i - e_j)A(\mathcal{G},k)^{-1}(e_i - e_j).$$

Now, consider the case when $i = n+1$, that is, we want to know the effective resistance between the grounded, influencing node and any node in $\mathcal{G}$. Then, $e_i = \vec{0}$ and,

$$R_{n+1,j} = e_j A(\mathcal{G}, k)^{-1} e_j = [A^{-1}]_{jj}.$$

So, we can see that the effective resistance from the influencing node to any other node in $\mathcal{G}$ is given by the diagonal entries in the inverse of the grounded Laplacian. The total effective resistance for the influencing node (in $\hat{\mathcal{G}}$) attached to node k in $\mathcal{G}$, then, is simply the sum of the effective resistance for all nodes in $\mathcal{G}$, which is equivalent to the trace of A^{-1},

$$R_{tot}(k) = \mathbf{tr}(A(\mathcal{G}, k)^{-1}).$$

We'll see how this plays into the definition of information centrality next.

Information centrality definition: Starting in [112], the information between two nodes in an arbitrary graph has been defined in terms of the pseudo-inverse of the graph Laplacian,

$$I_{ij} = \frac{1}{L^{\dagger}_{ii} + L^{\dagger}_{jj} - 2L^{\dagger}_{ij}}.$$

This definition naturally takes into account the total sum of geodesic and non-geodesic paths that information can take between two nodes. We can see that the denominator is the effective resistance, so the information flow in a graph is equivalent to current flow in a resistive network. The information centrality for a given node is the harmonic average of information between that node and all other nodes in the network,

$$c_I(i) = \left(\frac{1}{n} \sum_{j=1}^{n} \frac{1}{I_{ij}} \right)^{-1},$$

which we can easily write in terms of the effective resistance,

$$c_I(i) = \frac{n}{\mathbf{tr}(A(G, i)^{-1})},$$

or, similar to when we considered the closeness centrality, $c_I(i)^{-1} = (1/n)\mathbf{tr}(A(G, i)^{-1}) = J_{\text{avg}}(\mathcal{G}, i)$, so we can see that nodes with high information centrality correspond to low average state cost for constant input.

Relating Information Centrality to Closeness Centrality: In [9], the argument is made that closeness centrality when defined in terms of current flow is equivalent to information centrality. Furthermore, they note that on trees, the current-flow metrics agree with the shortest-path measures (Theorem 4 - this result isn't necessarily unique, though; dating back to [60] is the notion that resistive distance is equal to the shortest path distance when there is a single path between two nodes.). That is, on trees, shortest-path closeness is equivalent to information centrality. We can see this explicitly from result above for trees from [19]. Here, we've show that we can generalize the tree result to arbitrary graphs using information centrality instead of closeness centrality. Furthermore, we observe that this generalization does not change how the scaling comes in - so the "weighted sum" interpretation follows through nicely too.

6.3.5 Dynamical Influence

In [61], a centrality measure is proposed that seeks to quantify the influence of a node on the overall network dynamics. The measure is as follows. Consider a system with coupled linear dynamics given by,

$$\dot{x} = Mx$$

where M is a $n \times n$ real matrix with maximum eigenvalue equal to zero. Consensus dynamics (scaled or not) fits this, obviously. Let c correspond to the left eigen-vector of M corresponding to $\lambda_{\max} = 0$ and e be the right-eigenvalue. Then, the consensus value is (which matches the result for multi-rate consensus from [85]),

$$x(\infty) = \frac{c^T x(0)}{c^T e} e$$

The claim is then: the projection of $x(0)$ on c is what the system "remembers" about x(0) at large times, so the "coefficient c_i quantifies the extent to which the initial condition at node i affects the final state." Thus, c_i is the dynamical influence of element i for the

coupled linear dynamics.

Recasting for scaled consensus: In this framework, the centrality of the node is given by its associated timescale - ϵ_i (because the vector of scaling parameters is the left eigenvector associated with $\lambda = 0$ for scaled consensus). Note that this means slow nodes are more central than faster nodes, as they have a larger influence on the consensus state than fast nodes.

6.3.6 Constant-Input Centrality Adjustment

We saw in 6.3.3 and 6.3.4 that the average state cost for the single input influenced consensus is equivalent to closeness centrality, which is closely related to the information centrality. It stands to reason, then, that any node can be made most central, and thus raises the question of what value of ϵ_i is necessary to make ν_i the most central node (assuming no other nodes scales are updated). First, we can note that slowing a node down always increases the average state cost. But, given a topology and scaling such that influencing of node l currently has the lowest average state cost, let node k be the desired central node, and $\tilde{\epsilon}_i$ be the new scaling on each node. We are only going to slow down k so $\tilde{\epsilon}_i = \epsilon_i$ for $i \neq k$, thus, we can see that the critical value of $\tilde{\epsilon}_k$ is given by,

$$\begin{aligned} \tilde{J}(G,k) &\leq \tilde{J}(G,l) \\ \sum_{j=1} \tilde{\epsilon}_j (d(k,j)+1) &\leq \sum_{j=1} \epsilon_j (d(k,j)+1) \\ \Rightarrow \tilde{\epsilon}_k &\geq \frac{1}{d(l,k)} \sum_{j \neq k}^{n} \epsilon_j (d(k,j) - d(l,j)). \end{aligned}$$

Now, in practice, finding the critical value in this fashion will give you the scaling of node k such that it is more central than the current central node - however, it does not guarantee that it is the most central node with the new scaling. In order to find the value of $\tilde{\epsilon}_k$ that truly makes node k the most central requires recursively solving the above equation

until k is the most central node. A future refinement of this preliminary result could be to reformulate the problem as a minimization problem with constraints to ensure that k is the most central node. Solving that optimization problem (e.g. via Lagrange multipliers) may yield an analytic solution that avoids potential recursive calculations.

6.3.7 Section Summary

In this section, we have shown that the addition of time scales to networked systems can allow for, or in some cases require, an expansion of the definition of traditional centrality measures. However, the utility of these new formulation, particular through the lens of how they pertain to system-theoretic characteristics, remains unclear and an avenue for future research. Particularly, based on the well-known connection between resistive networks and weighted graphs, what can be stated about the effective resistance (or analogous circuit network) in the presence of time scales. Finally, whenever the time scale parameters can be tuned (as the connection to clock parameters supports), questions of the optimal distribution or assignment of time scale parameters arise, so formalizing those optimization problems (and commenting on any topological-based consequences) would also be a potentially useful area to explore.

Chapter 7

CONCLUDING REMARKS

This work was primarily driven by introducing individual time scale parameters to the agents within the consensus protocol. Along that motivation, we have developed a simple, yet interesting, framework for investigating systems with heterogeneous agent dynamics. We both considered formulations arising from *a priori* assigning agents to adhere to a particular time scale, as well as arrived at similar formulations derived from considering clock-biased measurements and computation. This framework thus covers a wide range of various systems with differing agent dynamics as well as a range of sensor types and configurations.

In the context of *scaled consensus*, we presented results that highlighted the theoretical merit of studying multi-scale consensus by highlighting interesting deviations from mono-scale results (in the case of single-input influenced consensus), as well as the potential for performance-based time scale design (in the case of noise-driven consensus). We observed that despite the heterogeneity introduced by agent time scales, we were able to recover some graph-theoretic notions of performance limits, albeit limits that had to take the relevant time scales into account. We also saw that the performance (in the $\mathcal{H}_2$-norm sense), could be significantly impacted by the presence of time scales, but that this could also be exploited to suggest minimal communication topologies that promoted network resilience and disturbance rejection. Most importantly, we were able to show that these time scale contributions fundamentally differed from those from edge weighting, which add an interesting and important wrinkle into any real-world application of similar results that do not take time scaled into account.

In the context of networked agents operating faulty clocks, we not only observed that many general system/sensor configurations can be categorized into a few different standard

forms, but that these clock errors can inhibit consensus for double integrator systems. This introduces new motivation for clock synchronization algorithms which would reduce these errors. We propose one such algorithm, based on double integrator consensus dynamics, which allows for the consideration of the stability and potential pitfalls of simultaneous clock synchronization along with the consensus-based control laws.

Taken all together, the work presented here improves our ability to understand, analyze, and control networked dynamical systems with heterogeneous agent dynamics. This increases the applicability of the consensus algorithm to wider ranges of systems, as well as allow for approximation of performance for complex networked systems. It also categorizes and gives some formalism to agents with faulty clocks/biased measurements, which allows for understanding and wider usability of agents with less-than-ideal clocks or sensors by characterizing their errors in a way that had not been done before. This work also motivates future work in these areas, especially in the realm of clock-biased agents. In particular, the case where more complex control laws are being considered by networked agents is a necessary and interesting extension of this work.

BIBLIOGRAPHY

[1] M. Arcak and P. Kokotovic. Robust nonlinear control of systems with input unmodeled dynamics. *Systems & Control Letters*, 41(2):115–122, 2000.

[2] Armand Awad, Airlie Chapman, Eric Schoof, Anshu Narang-Siddarth, and Mehran Mesbahi. Time-scale separation on networks: consensus, tracking, and state-dependent interactions. In *Proc. 54th IEEE Conference on Decision and Control*, pages 6172–6177, Osaka, Japan, 2015.

[3] Armand Awad, Airlie Chapman, Eric Schoof, Anshu Narang-Siddarth, and Mehran Mesbahi. Time-Scale Separation in Networks: State-Dependent Graphs and Consensus Tracking. *IEEE Transactions on Control of Network Systems*, pages 1–1, 2018.

[4] Baruch Awerbuch. Optimal Distributed Algorithms for Minimum Weight Spanning Tree, Counting, Leader Election and Related Problems. *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 230–240, 1987.

[5] Bassam Bamieh, Mihailo R. Jovanović, Partha Mitra, and Stacy Patterson. Coherence in Large-Scale Networks: Dimension-Dependent Limitations of Local Feedback. *IEEE Transactions on Automatic Control*, 57(9):2235–2249, 2012.

[6] C. M. Bender and S. A. Orszag. *Advanced Mathematical Methods for Scientists and Engineers I: Asymptotic Methods and Perturbation Theory*. Springer-Verlag, New York, NY, 1999.

[7] Emrah Bıyık and Murat Arcak. Area aggregation and time-scale modeling for sparse nonlinear networks. *Systems & Control Letters*, 57(2):142–149, 2008.

[8] Stephen P Borgatti and Martin G Everett. A graph-theoretic perspective on centrality. *Social networks*, 28(4):466–484, 2006.

[9] Ulrik Brandes and Daniel Fleischer. Centrality measures based on current flow. In *Annual symposium on theoretical aspects of computer science*, pages 533–544. Springer, 2005.

[10] Kenneth R Britting. Inertial navigation systems analysis. 2010.

[11] Stefano Carletta and Paolo Teofilatto. Design and numerical validation of an algorithm for the detumbling and angular rate determination of a cubesat using only three-axis magnetometer data. *International Journal of Aerospace Engineering*, 2018, 2018.

[12] Ruggero Carli and Sandro Zampieri. Networked clock synchronization based on second order linear consensus algorithms. *Proceedings of the IEEE Conference on Decision and Control*, pages 7259–7264, 2010.

[13] Aranya Chakrabortty and Pramod P Khargonekar. Introduction to wide-area control of power systems. In *2013 American Control Conference*, pages 6758–6770. IEEE, 2013.

[14] Airlie Chapman. *Semi-Autonomous Networks: Effective Control of Networked Systems through Protocols, Design, and Modeling*. PhD thesis, University of Washington, 2013.

[15] Airlie Chapman and Mehran Mesbahi. System theoretic aspects of influenced consensus: Single input case. *IEEE Transactions on Automatic Control*, 57(6):1505–1511, 2012.

[16] Airlie Chapman and Mehran Mesbahi. Semi-autonomous consensus: Network measures and adaptive trees. *IEEE Transactions on Automatic Control*, 58(1):19–31, 2013.

[17] Airlie Chapman and Mehran Mesbahi. State controllability, output controllability and stabilizability of networks : A symmetry perspective. In *Proc. 54th IEEE Conference on Decision and Control*, pages 4776–4781, Osaka, Japan, 2015.

[18] Airlie Chapman and Mehran Mesbahi. Multiple time-scales in network-of-networks. In *2016 American Control Conference (ACC)*, pages 5563–5568, July 2016.

[19] Airlie Chapman, Marzieh Nabi-Abdolyousefi, and Mehran Mesbahi. Identification and infiltration in consensus-type networks. *IFAC Proceedings Volumes*, 42(20):84–89, 2009.

[20] Airlie Chapman, Marzieh Nabi-Abdolyousefi, and Mehran Mesbahi. Controllability and observability of network-of-networks via Cartesian products. *IEEE Transaction on Automatic Control*, 59(10):2668–2679, 2014.

[21] Airlie Chapman, Eric Schoof, and Mehran Mesbahi. Online Adaptive Network Design for Disturbance Rejection. In *Principles of Cyber-Physical Systems*. Cambridge University Press, 2015.

[22] Yao Chen, Jinhu Lu, Xinghuo Yu, and David J. Hill. Multi-agent systems with dynamical topologies: Consensus and applications. *IEEE Circuits and Systems Magazine*, 13(3):21–34, 2013.

[23] J. Chow and P. Kokotović. Time scale modeling of sparse dynamic networks. *IEEE Transactions on Automatic Control*, 30(8):714–722, 1985.

[24] Mathias Hudoba de Badyn, Dillon R Foight, Daniel Calderone, Mehran Mesbahi, and Roy S Smith. Graph-theoretic optimization for edge consensus. *arXiv preprint arXiv:2006.16201*, 2020.

[25] Anton H De Ruiter, Christopher Damaren, and James R Forbes. *Spacecraft Dynamics and Control: an Introduction*. John Wiley & Sons, 2012.

[26] Dimos V Dimarogonas, Panagiotis Tsiotras, and Kostas J Kyriakopoulos. Leader–follower cooperative attitude control of multiple rigid bodies. *Systems & Control Letters*, 58(6):429–435, 2009.

[27] Florian Dörfler. Electric networks and algebraic graph theory: Models, properties and applications. *Proceedings of the IEEE*, 106(5):997–1005, 2018.

[28] Ky Fan. Maximum properties and inequalities for the eigenvalues of completely continuous operators. *Proceedings of the National Academy of Sciences*, 37(11):760–766, 1951.

[29] Jian'an Fang, Guoxiang Gu, and Kang-Zhi Liu. Stability analysis for nonlinear feedback control systems with linear actuators. *IEEE transactions on automatic control*, 48(4):649–654, 2003.

[30] Yuguang Fang, Kenneth A Loparo, and Xiangbo Feng. Inequalities for the trace of matrix product. *IEEE Transactions on Automatic Control*, 39(12):2489–2490, 1994.

[31] J Alexander Fax and Richard M Murray. Information flow and cooperative control of vehicle formations. *IEEE transactions on automatic control*, 49(9):1465–1476, 2004.

[32] Katherine Fitch and Naomi Ehrich Leonard. Information centrality and optimal leader selection in noisy networks. In *52nd IEEE Conference on Decision and Control*, pages 7510–7515. IEEE, 2013.

[33] Katherine Fitch and Naomi Ehrich Leonard. Joint centrality distinguishes optimal leaders in noisy networks. *IEEE Transactions on Control of Network Systems*, 3(4):366–378, 2015.

[34] Dillon R Foight, Mathias Hudoba de Badyn, and Mehran Mesbahi. Performance and design of consensus on matrix-weighted and time-scaled graphs. *IEEE Transactions on Control of Network Systems*, 7(4):1812–1822, 2020.

[35] Dillon R. Foight, Mathias Hudoba de Badyn, and Mehran Mesbahi. Time Scale Design for Network Resilience. In *Proc. 58th IEEE Conference on Decision and Control*, pages 1–8, Nice, France, 2019.

[36] Dillon R Foight and Mehran Mesbahi. Influenced consensus for multi-scale networks. In *2019 American Control Conference (ACC)*, pages 2753–2758. IEEE, 2019.

[37] Dillon R Foight, Anshu Narang-Siddarth, and Mehran Mesbahi. Reduced order modeling with slow actuators with application to leader-follower satellite constellations. In *2018 Annual American Control Conference (ACC)*, pages 6433–6438. IEEE, 2018.

[38] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[39] Linton C Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978.

[40] Linton C Freeman, Douglas Roeder, and Robert R Mulholland. Centrality in social networks: Ii. experimental results. *Social networks*, 2(2):119–141, 1979.

[41] Kurt O Friedrichs, Wolfgang R Wasow, et al. Singular perturbations of non-linear oscillations. *Duke Mathematical Journal*, 13(3):367–381, 1946.

[42] R. G. Gallager, P. A. Humblet, and P. M. Spira. A Distributed Algorithm for Minimum-Weight Spanning Trees. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 5(1):66–77, 1983.

[43] Lizhen Gao, Yingying Zhang, Xiaoming Zhang, and Yuyang Xue. A real-time estimation method of roll angle and angular rate based on geomagnetic information. *Mathematical Problems in Engineering*, 2020, 2020.

[44] Yuko Hatano and Mehran Mesbahi. Agreement over random networks. *IEEE Transactions on Automatic Control*, 50(11):1867–1872, 2005.

[45] Jianping He, Xiaoming Duan, Peng Cheng, Ling Shi, and Lin Cai. Accurate clock synchronization in wireless sensor networks with bounded noise. *Automatica*, 81:350–358, 2017.

[46] João P Hespanha. *Linear Systems Theory*. Princeton University Press, 2018.

[47] Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[48] Mathias Hudoba de Badyn. *On the Control of Consensus Networks: Theory and Applications*. PhD thesis, University of Washington, 2017.

[49] Mathias Hudoba de Badyn, Utku Eren, Behçet Açıkmeşe, and Mehran Mesbahi. Optimal mass transport and kernel density estimation for state-dependent networked dynamic systems. In *Proc. 57th IEEE Conference on Decision and Control*, Miami Beach, USA, 2018.

[50] Mathias Hudoba de Badyn and Mehran Mesbahi. Efficient computation of performance on series-parallel networks. In *Proc. American Control Conference*, pages 1–6, Philadelphia, USA, 2019.

[51] Peter C Hughes. *Spacecraft attitude dynamics*. Courier Corporation, 2012.

[52] A. Narang-Siddarth J. Valasek, M. R. Akella and E. Rollins. Adaptive dynamic inversion control of linear plants with control position constraints. 20(4):918–933, July 2014.

[53] Ali Jadbabaie, Jie Lin, and A Stephen Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[54] Z. P. Jiang and M. Arcak. Robust global stabilization with ignored input dynamics: an input-to-state stability (iss) small-gain approach. 46(9):1411–1415, 2001.

[55] Charles R Johnson and Roger A Horn. *Matrix analysis*. Cambridge university press Cambridge, 1985.

[56] R. S. Johnson. *Singular Perturbation Theory: Mathematical and Analytical Techniques with Applications to Engineering*. Springer US, 2005.

[57] Matthew A. Joordens and Mo Jamshidi. Underwater swarm robotics consensus control. In *Proc. IEEE International Conference on Systems, Man and Cybernetics*, number October, pages 3163–3168, San Antonio, USA, 2009.

[58] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, Upper Saddle River, NJ, third edition, 2002.

[59] Shalinee Kishore and Gang Xiong. Analysis of distributed consensus time synchronization with gaussian delay over wireless sensor networks. *Eurasip Journal on Wireless Communications and Networking*, 2009, 2009.

[60] Douglas J Klein and Milan Randić. Resistance distance. *Journal of mathematical chemistry*, 12(1):81–95, 1993.

[61] Konstantin Klemm, M Ángeles Serrano, Víctor M Eguíluz, and Maxi San Miguel. A measure of individual role in collective dynamics. *Scientific reports*, 2(1):1–8, 2012.

[62] Petar Kokotović, Hassan K Khalil, and John O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design*. SIAM, 1999.

[63] Petar V Kokotovic, RE O'Malley, and Peddapullaiah Sannuti. Singular perturbations and order reduction in control theory: An overview. *Automatica*, 12(2):123–132, 1976.

[64] Adolf Krazer. *Negotiations of the third international mathematicians' conference in Heidelberg from August 8th to 13th, 1904*. BG Teubner, 1905.

[65] Shay Kutten and David Peleg. Fast distributed construction of small k-dominating sets and applications. *Journal of Algorithms*, 28(1):40–66, 1998.

[66] Jonathan R Lawton and Randal W Beard. Synchronized multiple spacecraft rotations. *Automatica*, 38(8):1359–1364, 2002.

[67] Norman Levinson. Perturbations of discontinous solutions of non-linear system of differential equations. *Selected Papers of Norman Levinson*, page 288, 1997.

[68] Y Li, J Xiang, and W Wei. Consensus problems for linear time-invariant multi-agent systems with saturation constraints. *IET Control Theory & Applications*, 5(6):823–829, 2011.

[69] Z. Lin. *Low Gain Feedback*. Springer-Verlag, London, UK, 1999.

[70] Jonathan S Litt, T Shane Sowers, A Karl Owen, CE Fulton, AK Chicatelli, et al. Flight simulator evaluation of enhanced propulsion control modes for emergency operation. In *Infotech@ Aerospace 2012*, page 2604. 2012.

[71] Yu Liu, Xidong Tang, Gang Tao, and Suresh M Joshi. Adaptive failure compensation for aircraft tracking control using engine differential based model. In *2006 American Control Conference*, pages 6–pp. IEEE, 2006.

[72] M. K. Maggs, S. G. O'Keefe, and D. V. Thiel. Consensus clock synchronization for wireless sensor networks. *IEEE Sensors Journal*, 12(6):2269–2277, 2012.

[73] AR Meenakshi and C Rajian. On a product of positive semidefinite matrices. *Linear Algebra and its Applications*, 295(1-3):3–6, 1999.

[74] Ziyang Meng, Zhiyun Zhao, and Zongli Lin. On global leader-following consensus of identical linear dynamic systems subject to actuator saturation. *Systems & Control Letters*, 62(2):132–142, 2013.

[75] Mehran Mesbahi and Magnus Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 2010.

[76] Nima Monshizadeh, Harry Trentelman, and M. Kanat Camlibel. Projection based model reduction of multi-agent systems using graph partitions. *IEEE Transactions on Control of Network Systems*, 1(2):145–154, 2014.

[77] Nima Monshizadeh, Harry L. Trentelman, and M. Kanat Camlibel. Stability and synchronization preserving model reduction of multi-agent systems. *Systems and Control Letters*, 62(1):1–10, 2013.

[78] D Subbaram Naidu and Anthony J Calise. Singular perturbations and time scales in guidance and control of aerospace systems: A survey. *Journal of Guidance, Control, and Dynamics*, 24(6):1057–1078, 2001.

[79] A. Narang-Siddarth and J. Valasek. *Nonlinear Time Scale Systems in Standard and Nonstandard Forms: Analysis and Control*. SIAM, Philadelphia, PA, 2014.

[80] A.H. Nayfeh. *Introduction to Perturbation Techniques*. Wiley-VCH, 1993.

[81] Juhani Nieminen. On the centrality in a graph. *Scandinavian journal of psychology*, 15(1):332–336, 1974.

[82] Reza Olfati-Saber. Distributed Kalman filter with embedded consensus filters. In *Proceedings of the 44th IEEE Conference on Decision and Control and the European Control Conference*, pages 8179–8184, Seville, Spain, 2005.

[83] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[84] Reza Olfati-Saber and Richard M. Murray. Consensus protocols for networks of dynamic agents. *Proc. of the American Control Conference*, 2:951–956, 2003.

[85] Reza Olfati-Saber and Richard M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

[86] Reza Olfati-Saber and Richard M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

[87] Fabio Pasqualetti, Florian Dörfler, and Francesco Bullo. Attack detection and identification in cyber-physical systems. *IEEE Transactions on Automatic Control*, 58(11):2715–2729, 2013.

[88] Fabio Pasqualetti, Chiara Favaretto, Shiyu Zhao, and Sandro Zampieri. Fragility and controllability tradeoff in complex networks. In *2018 American Control Conference (ACC)*, pages 216–221, June 2018.

[89] Fabio Pasqualetti, Sandro Zampieri, and Francesco Bullo. Controllability metrics, limitations and algorithms for complex networks. *IEEE Transactions on Control of Network Systems*, 1(1):40–52, 2014.

[90] R Patel and M Toda. On norm bounds for algebraic riccati and lyapunov equations. *IEEE Transactions on Automatic Control*, 23(1):87–88, 1978.

[91] Stacy Patterson and Bassam Bamieh. Leader selection for optimal network coherence. In *49th IEEE Conference on Decision and Control (CDC)*, pages 2692–2697, Dec 2010.

[92] Stacy Patterson and Bassam Bamieh. Consensus and coherence in fractal networks. *IEEE Transactions on Control of Network Systems*, 1(4):338–348, 2014.

[93] Francisco Pedroche Sánchez, Miguel Rebollo Pedruelo, Carlos Carrascosa Casamayor, and Alberto Palomares Chust. Convergence of weighted-average consensus for undirected graphs. In *International Journal of Complex Systems in Science*, volume 4, pages 13–16, 2014.

[94] Kaare Brandt Petersen and Michael Syskind Pedersen. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

[95] Daniel Pickem, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron, and Magnus Egerstedt. The robotarium: A remotely accessible swarm robotics research testbed. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1699–1706. IEEE, 2017.

[96] Mohammad Pirani and Shreyas Sundaram. On the smallest eigenvalue of grounded laplacian matrices. *IEEE Transactions on Automatic Control*, 61(2):509–514, 2016.

[97] Philip D. Powell. Calculating determinants of block matrices, 2011.

[98] Jihene Ben Rejeb, Irinel-Constantin Morărescu, and Jamal Daafouz. Synchronization in networks of linear singularly perturbed systems. In *2016 American Control Conference (ACC)*, pages 4293–4298, July 2016.

[99] Jihene Ben Rejeb, Irinel Constantin Morărescu, and Jamal Daafouz. Control design with guaranteed cost for synchronization in networks of linear singularly perturbed systems. *Automatica*, 91:89–97, 2018.

[100] Wei Ren. Distributed attitude consensus among multiple networked spacecraft. In *American Control Conference, 2006*, pages 6–pp. IEEE, 2006.

[101] Wei Ren. Distributed cooperative attitude synchronization and tracking for multiple rigid bodies. *IEEE Transactions on Control Systems Technology*, 18(2):383–392, 2009.

[102] Wenjing Ren, Bin Jiang, and Hao Yang. A survey on singular perturbation theory in aerospace application. In *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, pages 675–680. IEEE, 2016.

[103] Diego Romeres, Florian Dörfler, and Francesco Bullo. Novel results on slow coherency in consensus and power networks. In *2013 European Control Conference (ECC)*, pages 742–747. IEEE, 2013.

[104] Sandip Roy, Yan Wan, and Ali Saberi. On time-scale designs for networks. *International Journal of Control*, 82(7):1313–1325, 2009.

[105] Peddapullaiah Sannuti. Singular perturbation method in the theory of optimal control. Technical report, ILLINOIS UNIV URBANA COORDINATED SCIENCE LAB, 1968.

[106] Peddapullaiah Sannuti and P v Kokotovic. Near-optimum design of linear systems by a singular perturbation method. *IEEE Transactions on Automatic Control*, 14(1):15–22, 1969.

[107] Luca Schenato and Giovanni Gamba. A distributed consensus protocol for clock synchronization in wireless sensor network. *Proceedings of the IEEE Conference on Decision and Control*, pages 2289–2294, 2007.

[108] Robert Sedgewick and Kevin Daniel Wayne. *Algorithms*. Addison-Wesley, 4th edition, 2011.

[109] Milad Siami and Nader Motee. Fundamental limits and tradeoffs on disturbance propagation in linear dynamical networks. *IEEE Transaction on Automatic Control*, 61(12):4055–4062, 2016.

[110] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control, Analysis and design*. John Wiley & Sons, 2001.

[111] V. Stepanyan and N. Nguyen. Control of systems with slow actuators using time scale separation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 6664–6679, Chicago, IL, August 2009.

[112] Karen Stephenson and Marvin Zelen. Rethinking centrality: Methods and examples. *Social networks*, 11(1):1–37, 1989.

[113] Housheng Su and Michael ZQ Chen. Multi-agent containment control with input saturation on switching topologies. *IET Control Theory & Applications*, 9(3):399–409, 2015.

[114] Housheng Su, Michael ZQ Chen, James Lam, and Zongli Lin. Semi-global leader-following consensus of linear multi-agent systems with input saturation via low gain feedback. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(7):1881–1889, 2013.

[115] Housheng Su, Gui Jia, and Michael ZQ Chen. Semi-global containment control of multi-agent systems with input saturation. *IET Control Theory & Applications*, 8(18):2229–2237, 2014.

[116] Herbert G Tanner, George J Pappas, and Vijay Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, 2004.

[117] GPS Product Team. Global positioning system (gps) standard positioning service (sps) performance analysis report. *GPS Product Team: Washington, DC, USA*, 2014.

[118] Andrei Nikolaevich Tikhonov. Systems of differential equations containing small parameters in the derivatives. *Matematicheskii sbornik*, 73(3):575–586, 1952.

[119] Minh Hoang Trinh and Hyo-Sung Ahn. Theory and applications of matrix-weighted consensus. *arXiv preprint arXiv: 1703:00129v3*, (1):1–21, 2017.

[120] S Emre Tuna. Lqr-based coupling gain for synchronization of linear systems. *arXiv preprint arXiv:0801.3390*, 2008.

[121] J. Urnes. Flight control for multi-engine uav aircraft using propulsion control. In *Infotech Aerospace*, Garden Grove, CA, June 2012.

[122] Adelaida Borisovna Vasil'eva. Asymptotic behaviour of solutions to certain problems involving non-linear differential equations containing a small parameter multiplying the highest derivatives. *Russian Mathematical Surveys*, 18(3):13, 1963.

[123] PKC Wang, FY Hadaegh, and K Lau. Synchronized formation rotation and attitude control of multiple free-flying spacecraft. *Journal of Guidance Control and Dynamics*, 22:28–35, 1999.

[124] JT-Y Wen and Kenneth Kreutz-Delgado. The attitude control problem. *IEEE Transactions on Automatic control*, 36(10):1148–1162, 1991.

[125] Kar-Keung D Young and Petar V Kokotovic. Analysis of feedback-loop interactions with actuator and sensor parasitics. *Automatica*, 18(5):577–582, 1982.

[126] Daniel Zelazo and Mathias Burger. On the definiteness of the weighted Laplacian and its connection to effective resistance. In *Proc. 53rd IEEE Conference on Decision and Control*, pages 2895–2900, 2014.

[127] Daniel Zelazo and Mehran Mesbahi. Edge agreement: Graph-theoretic performance bounds and passivity analysis. *IEEE Transactions on Automatic Control*, 56(3):544–555, 2011.

[128] Daniel Zelazo, Simone Schuler, and Frank Allgöwer. Performance and design of cycles in consensus networks. *Systems and Control Letters*, 62(1):85–96, 2013.