Università
degli Studi
di Padova

Dipartimento
di Ingegneria
dell'Informazione

On Fundamental Trade-Offs and Architecture Design in Networked Control Systems

Is the More Always the Better?

Ph.D. candidate
Luca Ballotta

Advisor
prof. Luca Schenato

Director & Coordinator
prof. Andrea Neviani

Ph.D. School in
Information Engineering

Department of
Information Engineering
University of Padova
2022

Abstract

It can be legitimately said that Networked Control Systems represent one of the biggest breakthroughs in engineering over the latest decades. Stemming from the intertwining among control, computer engineering, and telecommunications, these powerful systems received the legacy of classical communication and computer networks, but leveled it up by virtue of *autonomy* of each involved unit. Nowadays, examples of Networked Control Systems are smart power grids, smart homes and buildings, Industry 4.0 and Industrial Internet of Things, and smart agriculture, to mention a few. Even more futuristic applications, such as networks of autonomous vehicles or search-and-rescue robotic teams, are predicted to be available on the market in a matter of time.

Despite the exponential growth of such systems both in industrial applications and in research, one main reason why the current development is somewhat refrained on several aspects is that designing a Networked Control System is challenging in nature. In fact, not only blending different engineering fields raises novel issues, but also the interdependence of individual subsystems makes it hard to design control and, in general, decision-making procedures at *local level*, whereas design at *global level* is not only undesired but sometimes even unfeasible. To mention just one example of design complexity, while it is well known that the optimal Linear Quadratic controller for a single system can be found by solving a (relatively simple) algebraic matrix equation, it is also known that solving the same problem for a distributed controller is NP hard.

Because engineered systems must work in real life, the lack of strong theoretical results is typically replaced with *ad-hoc* and heuristic methods, that try to take the best of both worlds of human experience and available mathematical tools. While such solutions have already yielded impressive applications, relying on intuition might not always be the best strategy, and theoretical advancement is needed to unleash the full potential of Networked Control Systems. For example, recently introduced Multi-Agent Reinforcement Learning, even though it has proved powerful in some scenarios, still leaves open room for improvement before it can be safely deployed in the real world.

This thesis investigates, and possibly questions, the role that conventional wisdom plays in design of Networked Control Systems. Specifically, the aim is to explore some situations where common design beliefs might not match the real nature of the system to be designed, possibly causing loss in performance. Three conventions will be examined: more sensors improve estimation; more communication links increase control performance; more collaboration enhances cooperative tasks. While such conventions seem indeed reasonable, results exposed in this thesis show that it is not always so because of nontrivial performance trade-offs: in fact, more sensors may hinder estimation under computational delays; more communication links may degrade control performance under communication delays; more collaboration may be dangerous under misbehaving agents.

Even though results mostly focus on analysis, and practical indications for synthesis are still preliminary, the aim, and the hope, of this piece of research is to offer formally solid insights and high-level guidelines that can improve standard design techniques, possibly paving the way to novel research directions towards high-performing Networked Control Systems in the real world.

Contents

1 Introduction

The advent of Networked Control Systems has revolutionized a plethora of existing control application domains, and given rise to others that would have been impossible with classical control tools. In particular, development of wireless communication protocols tailored to industrial applications has yielded novel distributed paradigms, such as Edge and Fog Computing, the Internet of Things, distributed Federated Learning, and Industry 4.0, that can tame unprecedented challenges and provide new kinds of applications at all scales, from single users navigating urban traffic to electrical energy city-wide harvesting.

Examples of such systems are several: from classical Wireless Sensor Networks (WSNs) to the "smart" domains, such as smart houses, smart buildings, smart agriculture, smart grids, to more futuristic applications such as fleets of autonomous cars for efficient traffic management, formations of drones for coordinated delivery of goods, or teams of robots for collaborative exploration or search-and-rescue missions in hazardous environments.

This distributed paradigm has now been around for a while, and new developments keep popping both in research laboratories and in industry. As a matter of fact, while applications do get to work in real life, theory is still limited in several topics, and designers of real systems often need to rely on heuristics-based building blocks to fill those gaps that are not covered by rigorous theoretical tools. This happens because Networked Control Systems may be extremely challenging to deal with from a formal standpoint: from information flow corrupted by limitations of wireless channel, to dynamical couplings affecting architecture of distributed controllers, to local information processing amplified by network propagation, the interconnected nature of such systems introduces unprecedented difficulties in both modeling, analysis, and synthesis.

Hence, in spite of the already huge amount of research involving multifold facets of Networked Control Systems, there is still need for theoretical development to fill those gaps, possibly correcting current design approaches in systems whose actual behavior may not adhere to human intuition.

The more sensors, the better?

The more communication, the better?

The more collaboration, the better?

Figure 1.1: This thesis challenges widespread conventions adopted in design of Networked Control Systems. When configuring sensors, it is assumed that increasing sensing capabilities improves monitoring performance. As for distributed controller architecture, deploying more communication links is regarded as beneficial to closed-loop performance. In cooperative tasks, collaborative protocols are preferred to enhance information exchange across agents. While such heuristic guidelines might hold true, the research developed in this thesis shows that in some cases structural limitations imposed by Networked Control Systems may in fact produce nontrivial performance trade-offs. Taking this into account is important for effective design: if the latter is driven by misleading intuition, performance of the system can drastically degrade.

1.1 Literature Review

In this section, I survey classical and state-of-the-art research that has been pursued to tackle challenges arising with Networked Control Systems. I focus on three broad domains, which are loosely related and can be approached independently of one another.

In Section 1.1.1, I go through results in distributed sensing design, ranging from standard sensor selection and scheduling to the more recent challenges in Edge Computing, which are related to hardware limitations of compute-equipped network nodes.

In Section 1.1.2, I overview research in design of decentralized and distributed controller architecture, where a crucial role is played by communication constraints, in particular those pertaining to wireless channel.

Finally, in Section 1.1.3 I give an overview of collaborative strategies aimed to enhance cooperation among agents within the network, and conversely techniques to tackle the presence of misbehaving agents that can disrupt cooperative tasks.

1.1.1 Sensing design

Design of sensing apparatus has always been at the core of control systems. Indeed, feedback is one of the most basic and important concepts in both theory and applications, whereby physical information about the system to be controlled and, possibly, its surrounding environment, needs to be supplied to apply suitable control actions.

Here, a basic and widespread convention is that *the more sensors are the better*: because these measure a signal of interest, the tendency is to use as many sensors as possible so as to provide the control pipeline with rich information about the signal to be monitored or controlled. More in general, it is usually desired to push on the sensing both in number and capabilities (such as local data processing) of sensors.

In the following, I expose both classical challenges dating back to traditional control theory and new ones introduced the enhanced structure of Networked Control Systems, together with current available solutions. I first give an overview of the sensor selection problem, then present strategies for efficient sensor scheduling, and finally I review more recent trends on distributed computation enabled by "smart sensors".

Sensor Selection

One of the most classical problems is sensor selection: given a set of available sensors, to decide which of those are to be chosen based on constraints. The latter are usually expressed as a budget, that may encode economical restrictions, battery limitations, sensor placement feasibility (*e.g.,* maximal weight allowed on autonomous platforms), or other kinds of constrains. Because of those, selecting all sensors is typically not possible, which leads to optimization problems prone to combinatorial explosion. It is worth noting that this class of problems (and others presented in the following) falls under the broad umbrella of *resource allocation*, which roughly speaking includes those problems where a limited amount of resources needs to be chosen from the full set of available resources, and can be traced back to fields such as economics and logistics.

Despite hardness of selection, control and optimization literature is rich in both methodological and ad-hoc strategies that exploit structural properties of the system at hand to circumvent its combinatorial nature. For example, one notable feature that arises in some cases (for example, when constraints can be expressed as a matroid), is submodularity, which enjoys the diminishing-return property. In words, this ensures that selecting the single best sensor to be added to a given set yields larger improvement if such given set is smaller, and can be exploited to analytically bound performance of greedy (computational efficient) selection algorithms.

A far-from-exhaustive list of methodological approaches is given by [47], [69], [77], [93], [96], [140], [181], [182], [199]. For example, reference [96] uses an LMI approach to compute sensor and actuator requirements from performance specifications; work [69] develops a randomized approach to sensor selection aimed to minimize estimation error variance; paper [47] focuses on strategies to monitor nonlinear models; authors in [182] study submodularity property of sensor selection and propose greedy algorithms with suboptimality guarantees.

On the other hand, works tailored to specific scenarios and applications are [7],

[32], [33], [44], [85], [97], [116], [148], [170]. For example, work [33] tackles selection of vision sensors in resource-constrained robot navigation; paper [85] focuses on underwater monitoring; reference [44] deals with economical constraints involving sets of available cheap and expensive sensors.

It is worth noting that sensor selection in Networked Control Systems comes with old and new problems alike: for example, wireless communication between sensors and controller may be taken into account, as well as the very network structure and how this intertwines with the environment. Tailored approaches to a few of such issues deal with underwater sensor networks [85] or mobility scenarios [33], [170]. A fairly recent trend is co-design of sensing, communication, and control. In particular, not only sensor selection is subject to budget constraints, but the used cost function is task driven and addresses specific control and/or communication requirements. Examples of such trend are [191], [200], where authors study sensor selection within the LQG framework, or [222], [223], that deal with hardware and control co-design of autonomous mobile platforms.

Sensor Scheduling

A topic closely related to sensor selection, which also pertains to the resource allocation umbrella, is sensor scheduling: given a shared communication channel and a set of transmitting sensors, decide which of those get to send their updates overtime. In this case, the limited budget is typically caused by wireless channel constraints, whereby both general-purpose protocols, such as standard Wi-Fi, and industrial protocols can handle only a limited number of simultaneous transmissions to avoid collisions due to interference. Also, even in modern protocols for massive local networks, such as 5G, the need for scheduling can arise if multiple users spatially close-by need to be allotted on the same frequency.

Classical device scheduling, born way before Networked Control Systems, was explored by the telecommunication community and mostly focused on purely communication-motivated quantities to be optimized, such as latency or throughput. However, the introduction of controlled applications has caused a shift in perspective, whereby variables more suitable to control performance are to be taken into account. One major example in this regard is represented by Age of Information (AoI), which is defined as the time elapsed from generation (*e.g.,* sampling) of a received state update. Even though such a quantity is typically a proxy to actual control-theoretic cost functions, it has been proven effective to design compute-efficient scheduling policies with direct relation to control performance. Among the vast body of literature dealing with AoI, relevant works are [80], [189], [190], [197], [215], [228], whereas approaches tailored to estimation- and control-theoretic performance are studied in [26], [36], [89], [90], [142], [184], [196], which also address non-linear functions of AoI to better deal with control frameworks. A notable property that often arises from AoI-based scheduling are threshold-based policies: in words, this means that a sensors should transmit if and only if its latest update is older than a certain threshold. This feature is attractive in that it enables effective scheduling policies at low computational cost.

Besides AoI-driven approaches, strategies more closely related to estimation- and control-theoretic cost functions have been explored. Here, the challenge is that the dynamical program arising from the problem formulation can be hardly addressed from the analytical standpoint, one major obstacle being the partial ordering of error covariance matrices which rules out several mathematical tools for design and often limits achievable results to (partial) analysis of the optimal policy. In this regard, recent works are [57], [75], [95], [115], [146], [210], [227], where both heuristic, randomized, and exact approaches are used to analyze estimation-optimal scheduling in a Networked Control System.

Computation Design

Since the birth of Networked Control Systems, spreading computational tasks across the network has been crucial to scale up capability of globally achievable goals as compared to classical control applications. However, recent developments in hardware and computational power onboard small devices, such as microcontrollers and embedded GPUs, have induced renewed interest in this topic from both research and industrial perspective. In fact, the possibility of performing compute-sophisticated tasks across the whole Networked Control System raises both new targets and novel issues. On the one hand, pushing computation on network nodes can alleviate burden of massive workstations and servers, resulting in increased efficiency and easier scalability. Along this line, the recent trends of Edge and Fog Computing have caused great excitement within the research community, with the promise of further enhancing new paradigms of the Internet of Things, Industry 4.0, and Federated Learning, through "smart" devices at the network edge that can autonomously execute tasks with minimal centralized or distributed coordination. Such trend is attractive also in multi-robot applications, whereby robots are becoming more and more sophisticated by means of both enhanced hardware capabilities and impressive software developments, such as ultra-compression of huge Machine-Learning models (TinyML [209]). More in general, deployment of Machine Learning on lightweight and energy-constrained devices is regarded with renovated attention in several application domains [28], [39], [52], [104], [126], [206].

On the other hand, data processing hardware available on edge devices is still limited, naturally giving rise to a trade-off between resources and performance. As regards dynamical systems, one serious constraint is computation latency, which becomes crucial especially with so-called time-critical systems that need fresh information supply to be properly controlled. For example, an autonomous vehicle needs timely position and velocity updates to be correctly steered towards the desired spatial trajectory: too old information may be misleading and induce useless, if not dangerous, control actions. This entails a nontrivial *latency-accuracy trade-off*, whereby data processing improves accuracy of collected measurements at the cost of non-negligible delay. Further, in a scenario where information exchange occurs among hardware-constrained devices, for example within a network of smart sensors (*e.g.,* UAVs or smart cameras for remote surveillance), such computational limitations may also affect communication. Indeed, data processing often entails data compression, such as visual feature extraction for autonomous navigation. In this case, processed data are delivered more quickly than raw measurements, introducing additional complexity in the design, referred to as *computation-communication trade-off*.

While such delay issue is conceptually related to the Age of Information, the physical data sampling and processing mechanism is different from data packet transportation, possibly requiring alternative models and problem formulations. However, most work puts little attention on the connection between computational delays and control performance, focusing on minimization of latency or energy consumption [81], [106], [113], [163], [188]. For instance, reference [198] studies delays of different devices to find an optimal network processing policy; work [188] characterizes delays occurring in a network with cloud fog offloading, with case study on computation of the Fast Fourier Transform; report [73] investigates multimedia data processing under different architectures. Generally speaking, optimization of local data processing is hardly addressed, whereas the common trend is exploiting computational capabilities of sensors and edge devices while trying to decrease latency by means of available hardware or software (network protocol) resources.

In contrast to the latter, a very recent paradigm emerged in edge applications is computation offloading. In this framework, rather than putting the emphasis on computational capabilities of limited-resource devices, the followed approach is leveraging communication, and especially new protocols enabling low-latency transmissions (*e.g.,* 5G), to move computation from edge devices to cloud servers [40], [111], [134]. A notable application of this paradigm is cloud robotics, which allows robots to offload compute-demanding tasks in order to save time and energy and to also get accurate results (*e.g.,* from huge neural network used for perception inference) [42], [64], [137]. Here, a similar trade-off to the one observed above emerges: indeed, transmission of raw data from edge to cloud may induce channel congestion, especially if large data (such as those provided by vision sensors or lidars) are communicated. To deal with this issue, current approaches typically leverage learning-based methods to learn effective offloading policies [42], even though some works try to get analytical insight through statistical and information-theoretic tools [64].

1.1.2 Controller Architecture Design

The complex and spread out structure of Networked Control Systems has urged since their birth for a paradigm shift in design of controller architecture: from classical centralized control, with a single unit in charge of mastering the whole controlled plant, to decentralized and distributed control, whereby individual network nodes can perform decision-making based on local information. This choice has indeed paid off, by endowing a Networked Control System with nice scalability properties, as well as increasing its robustness to failure of individual agents, making maintenance easier and cheaper, and, crucially, decreasing the overall communication overhead at global level. Indeed, such a paradigm change is not only desired but necessary to many applications, where a single controller or, equivalently, all-to-all communication would be infeasible due to network congestion and limited bandwidth.

When options are given to design the communication and controller architecture, suitably choosing which links are to be deployed may dramatically impact performance. While in some cases standard network topologies are preferred or forced, such as star sensor networks, in other cases more freedom is given, leading to design challenges encoded by complex optimization problems. In the following, I first put the reader into context communication-wise by giving an overview of seminal and recent studies on control with communication constraints. These should clarify why the latter represents a crucial limitations to control performance, and motivate the design of optimal distributed architectures to clear communication occupancy. Next, I overview architecture design strategies, which can be classified under two broad categories: design of structured controllers and optimization of controller architecture.

Control with Communication Constraints

Typical issues pertaining to Networked Control Systems, and especially those using wireless data transmission, are non-idealities of communication channels, such as packet loss, transportation delays, and unreliability.

To tame these issues, research efforts in control theory have mainly focused on design of estimation and control in the presence of constrained communication channel, or, more in general, limited information (so-called "rational inattention"). Pioneering work in this regard is represented by [31], [136], [192], which combine control- and information-theoretic tools to investigate the relation between communication rate, stabilizability, and performance, revealing fundamental limitations on the ability to control a dynamical system with respect to the minimal amount of information that needs to be fed as input to a feedback-based controller. In particular, seminal work [168], [175] deals with Kalman filtering subject to packet loss, finding minimal rate constraints needed to ensure boundedness of estimation error variance. More recent works [43], [147], [171] investigate the relation among sensors, communication channel, and control within the LQG framework, while [98] addresses nonlinear systems where both sensor and actuator links are subject to packet drop.

Structured Controller

To tame constrained channel resources and relax network-level communication requirements, it is only natural to adopt distributed and decentralized controller architectures. In practice, rather than a single central unit that collects all sensory feedback and computes all control commands, each network node decides on its own actions based on locally available information. Specifically, we talk about decentralized control when no communication among nodes is involved, and about distributed control when such local information is constructed from both the node's own measurements and other data exchanged with close-by nodes (*i.e.,* neighbors induced by the communication network topology). Intuitively, if dynamical couplings are either naturally present (such as among LC oscillators in electrical grids) or desired to induce specific network-wide behaviors (such as in consensus problems), removing communication links also reduces available feedback information and hence degrades the control, inducing the classical trade-off between controller complexity and performance [78]. This general consideration motivates the conventional wisdom that *the more communication is the better*: the amount of exchanged information increases with the number of links deployed across the network, so that denser communication is regarded as necessary to improve performance.

The more straightforward way to design distributed controllers is to fix a communication topology beforehand and subsequently choose controller parameters, such as feedback gains. This is typically expressed as an optimization problem with a suitable control-theoretic cost function (*e.g.,* $\mathcal{H}_2$ norm or LQR cost) and constraints on the controller structure, for instance by setting to zero some elements of the feedback gain matrix. This approach is sometimes required, for example when the communication network already exists and/or cannot be easily modified. Early examples of this kind of design are found for linear formations, with classical application to vehicular platoons [37], [38], [108].

Here, the possible architectures are typically limited to nearest-neighbor interaction or leader-follower strategies, which nonetheless induce interesting optimization structures, and relate to important concepts such as string stability [187] and coherence [20].

Examples of applications within this piece of literature are nearly countless, ranging from wireless sensor networks, to multi-robot systems, to satellite networks, to vehicular platoons, to power grids and oscillator networks.

From the methodological standpoint, a large body of works exploits structural properties of the control task or system model at hand to derive some analytical characterization of the optimal controller. For instance, references [8], [9], [54], [130] study properties of spatially invariant systems, whereby optimal controller and estimator (which are dual to each other) can be characterized in a fairly precise way. As for other examples, work [162] proposes a general framework for design of structured controller; paper [10] investigates sparse architectures within the LQR framework; reference [76] proposes a controller design with time-varying links through stochastic failures. An interesting research trend is represented by study of Networked Control Systems where feedback delays are explicitly taken into account and embedded in the dynamics. A few such example are given in [48], [135], [231], [232], where authors explore conditions for stability in consensus networks with different delays and topologies. Finally, papers [55], [56], [107] propose efficient algorithms to design optimal and near-optimal structured controllers by suitably reformulating an $\mathcal{H}_2$-norm minimization problem.

Finally, a very recent trend comes from Machine Learning. Here, design techniques use data-driven approaches structurally tailored to distributed systems, such as recently developed Graph Neural Network, to learn controller parameters that minimize a control-theoretic loss function [61], [62], [120].

Architecture Optimization

Apart from practical issues associated with complex, nonconvex optimization problems [55], carving on stone the controller architecture *a priori* has the disadvantage that several design possibilities are discarded in the first place, possibly including options which would provide higher performance than the chosen one.

As so, a second approach that has pretty soon followed design of control parameters is optimization of the controller architecture, intended as choosing the very communication links to be deployed. This paradigm raises both potential advantages and increased complexity: on the one hand, more architecture options entail more degrees of freedom, and potentially better performance; on the other hand, the resulting control design problem becomes combinatorial in the controller structure, easily making its computational complexity intractable even for small networks.

Specifically, the optimization problem is usually expressed as minimization of a performance metric subject to a maximum cardinality (number of nonzero elements) of the feedback gain matrix, which would require to evaluate all possible controller architectures complying with the constraint. To overcome combinatorial explosion, typical methods in the literature approximate the cardinality constraint with a convex one by resorting to suitable matrix norms (ℓ_1- or ℓ_2-norm), and then possibly tackle the relaxed version of the problem where the constraint is embedded into the cost function as a soft penalty on controller complexity. Examples of this approach, which mostly focus on designing efficient algorithms that exploit the problem structure, are [53], [71], [78], [109], [211], [230]. For example, work [109] exploits separability of the relaxed cost function and proposes an ADMM approach, while [71] designs a proximal gradient-based optimization algorithm for a convex class of architecture design problems.

Besides this kind of approaches, another group of works tackles the design from different perspectives, putting emphasis on communication and latency, even though from a qualitative or heuristic standpoint. In particular, seminal works [121], [122] propose the Regularization for Design, addressing optimization of individual controller elements such as actuators, sensor, and communication links by minimizing appropriate atomic norms, while [6] investigates communication locality and its relation to control design within the novel System Level Synthesis framework.

Finally, another approach leverages tools from game theory, which have recently experienced renewed interest within the control community. This line of work leverages cooperative game theory, and specifically coalitional games, to find efficient "coalitions" (sets) of links with respect to LQR-like performance metrics [114], [119], [132], [133].

1.1.3 Resilient Control

In the previous sections, I have reviewed issues arising in Networked Control Systems because of "feasibility" constraints, related to, *e.g.,* monetary budget, physical limitations of communications channel, or complexity of optimization problems associated with the control design. While all such constraints almost purely depend on physical aspects of the system at hand, and ultimately affect control performance, the interconnected structure of Networked Control Systems generates a novel kind of problem, which not only impacts efficiency but also safety and security aspects.

A key principle of the Networked Control System paradigm is that complex goals can be achieved by leveraging coordination and cooperation among the agents composing the system, while each of those is in charge of a relatively simple local task. The probably most notorious example of cooperative task is the consensus problem, which consists in driving all agents towards a common value. This task is a fundamental tool in a number of scenarios and applications, ranging from distributed estimation to distributed optimization, to rendez-vouz within multi-robot teams, to synchronization of electrical devices, to distributed Federated Learning, to name a few. Hence, one of the warhorses of research on Networked Control Systems is to develop strategies that can make inter-agent collaboration effective. For example, the consensus problem has been completely characterized and can be solved by the consensus protocol [212], which requires only a few mild assumptions on the network topology. More in general, distributed control techniques attempt to exploit inter-agent collaboration in order to enhance achievement of the global task at hand.

However, autonomy of network agents from sensing, decision-making, and actuation standpoints on the one hand, and interdependence among nodes caused by the networked structure on the other hand, raise the critical concern of "failures", namely, arbitrary and uncontrolled behavior which can first emerge at individual agents and subsequently propagate, disrupting tasks at global scale. Notably, this problem can be easily tamed in classical control systems, while the distributed nature of Networked Control Systems makes it harder to both prevent, detect, and contrast local failures, and the risk of noting a problem when it has already spread across the network might be critical.

Interestingly, while the control community found itself in urgency to find suitable countermeasures, computer science and telecommunications communities had already faced issues such as cyber-attacks or cascading software bugs, for instance after the advent of the World Wide Web. Differently from old-fashioned computer networks, Networked Control Systems present different characteristics that require suitable solutions. In particular, the physical component is almost negligible in computer networks, but is extremely relevant in, *e.g.,* power grids or vehicular platoons. Broadly speaking, agent failures may be classified into two categories. The first kind of failures is caused by spontaneous hardware damages or software bugs in network components, which induce cascade effects and impact functioning of neighboring agents. The second type of failure, usually referred to as (cyber-)attacks, is caused by infiltration of adversaries from the outside of the network, that intentionally hack the system. According to the addressed class of failures, different models are used in the literature, that go from fairly simple and dumb misbehavior to sophisticated and unpredictable attacks.

In the rest of this section, I will overview both analytical works that study when a network is robust or resilient and methodological approaches aimed to enhance resiliency of networked control protocols.

Resilience and Robustness Analysis

A large portion of the literature focuses on conditions under which a Networked Control System is robust or resilient to local faults, due to either unintentional or malicious hacking. Given the abundance of related papers in this field, my aim here is just to give a gist of what is available on the table.

Many works study the topology of Networked Control Systems, which clearly represent a fundamental aspect that affects local interactions among network nodes and governs the overall system behavior at global scale. For example, classical work [159] investigates how consensus can be achieved with time-varying interactions; study [176] focuses on modeling road networks impacted by link failures; papers [202], [203] characterize conditions for consensus in undirected and directed graphs under faulty communications; reference [153] is concerned with detectability of node failures; paper [27] focuses on robustness to perturbations in graphs.

A second relevant body of works attempts to characterize effectiveness of different classes of attacks, to reveal fundamental limitations of, *e.g.,* distributed control or estimation methods. In particular, several attack models have been proposed and analytically investigated: seminal work [110] introduces false data injection, which spoofs static estimators; paper [193] investigates stealthy deception attacks against control actions; the line of work [128], [129] studies replay attacks, which repeat temporal sequences of sensor readings overtime; reference [4] is concerned with deception attacks in networks; the book [112, Chapter 6] introduces the concept of Byzantine attack, a widely used model to address powerful adversaries.

Other branches of the literature address resilience under manifold perspectives. To name a few, paper [3] considers performance and robustness of distributed control protocols under multiplicative noise, while the works [179], [185] analyze limitations of distributed optimization strategies in the presence of Byzantine and malicious attacks;

It goes without saying that analytical studies need to be complemented by suitable design and synthesis strategies in order for real systems to work. If the literature on robustness and resilience analysis of Networked Control Systems is huge, not less is the amount of works proposing techniques, methods, and algorithms to tame local failures and achieve network-level resilience. However, an exhaustive exposition of the literature in this field would be massive and is not within the scope of this thesis. Instead, being part of the proposed results focused on the problem of resilient consensus, the next paragraph describes in detail this narrow but relevant branch of the literature, which is then expanded into the more general problem of distributed optimization.

Resilient Consensus

The consensus problem is one of the most fundamental tools in distributed control, whereby the lack of global knowledge is replaced by suitable information processing protocols aimed to enable network-wide coordination from local interactions. In its basic form, the consensus protocol is an update rule as simple as effective, that ensures global alignment among local variables of all participating agents under mild technical conditions. In particular, average consensus, *i.e.,* where the consensus value is the average of initial values of agent variables, is needed in many applications, from distributed estimation to gathering in multi-robot systems. However, the consensus protocol is fragile, in that a single misbehaving agent can steer the rest of the network towards any configuration, arbitrarily far from the desired one. Hence, many techniques have been proposed to robustify the update rule of standard consensus. In particular, resilient consensus is defined as a configuration such that "normal" agents (*i.e.,* that do not misbehave) reach a consensus with each other in the face of misleading interactions with unknown adversaries, while possibly remaining inside a suitable safe region, a standard convention being the convex hull of initial agent conditions.

One of the first approaches, and yet probably the most important and used, is Mean Subsequence Reduced (MSR), presented in foundational work [88]. Essentially, this is a filtering technique that discards some of the received data based on their value: the largest and smallest such data are not used in the local update, ensuring that the agent will not drift towards very large (or very small) values. Seminal work [94], building on this simple rule, proposed two fundamental contributions: the first is algorithmic and consists in an improvement of MSR yielding the now state-of-the-art Weighted Mean Subsequence Reduced (W-MSR), whereby incoming values are not only filtered but also re-weighted at every iteration; the second, and probably most important, contribution is a formal study of theoretical guarantees associated with W-MSR, that give necessary and sufficient conditions for reaching resilient consensus. In words, if the underlying network topology has added structure in the form of a property called r-robustness, where r is a suitably large integer describing the ability of the network topology to internally spread information, then resilient consensus is guaranteed when the number of attackers is smaller than a threshold that depends on r. This fundamental result raises however two important concerns, which are in fact two sides of the same coin. The first is that r-robustness is a property that depends on the network graph elements (nodes and edges) in a combinatorial fashion: in practice, checking if it holds for given r would require and exhaustive search that quickly explodes with the size of the network, making such a verification intractable for large-scale systems. This raises the second issue, that is, evaluating how many (simultaneous) attacks the network is prone to: indeed, this number depends on the maximum parameter r for which r-robustness holds, forcing a designer to guess as the best replacement to ignorance of such r.

Even though theoretical guarantees of W-MSR are bound to such considerations, a huge body of research works and actual implementations certifies the practical effectiveness of this technique. In fact, many approaches to resilient consensus build on W-MSR and apply minor modifications to specialize it to narrower domains. To mention a few such cases, article [49] studies resilient control for double integrators; paper [207] addresses mobile adversaries that can change their attack location; work [201] focuses on a leader-follower framework; reference [174] targets nonlinear systems with state constraints.

While W-MSR is one of the most celebrated resilient consensus technique in the literature, other approaches have been proposed. More or less in the same year, authors of [145] proposed a system-theoretic approach to consensus under misbehaving agents, characterizing the case when some control inputs are not identifiable and providing a detection procedure. From a different viewpoint, paper [125] introduced the idea of "trust" to characterize agents that do not fail, and which can be used as "anchors" at network-level for a distributed update rule. The concept of trust has been recently given new resonance in different flavors: works [22], [218] design adaptive protocol local weights based on trust scores associated with transmissions and enabled by physical channels of information, such as directional signal profiles of wireless messages; authors in [1] study algorithmic robustness enabled by trusted agents; paper [224] proposes dynamically switching update rule for continuous-time double integrators.

Finally, other approaches attempt to reach a slightly different goal, endowed with more intrinsically robust properties compared to average consensus. A typical case is consensus to the median value, which is structurally more robust to generic "outliers" than average consensus. One of the first studies to propose this approach is given in [225], which analyzes its convergence properties under different attack models, whereas this idea is further developed in related works [58], [165], [173], [220].

Resilient Distributed Optimization

As seen in the previous section, literature about resilient consensus is quite vast. The reason is that in most cases some form of consensus either is required by the application or underlies the main control task, hence making the consensus step robust is essential even if it is not the ultimate goal.

Nevertheless, a body of literature has been focusing also on other control tasks, one of the most important ones being distributed optimization. Here, agents in the network need to solve an optimization problem only partially known to each of them, so that both coordination and optimization techniques are required. Early work [186] addresses this problem by analyzing the consensus protocol and assuming additive disturbances in misbehaving agent's updates, showing that normal agents can compute any function of the initial conditions if some connectivity properties hold. Note that this strictly relates to observability of a dynamical system. Other approaches use W-MSR as a building block to obtain resilient distributed optimization protocols. Examples are [180], [185], which address different attack modes and capabilities and show both suboptimality guarantees and fundamental limitations of such approach.

With this research direction rapidly growing over the latest years, several strategies and techniques have been arising for generic distributed optimization and control tasks. To mention a few examples, paper [219] exploits trust scores to dynamically adapt optimization protocol weights; work [41] proposes trust-based algorithms for mobile networks, with focus on traffic systems; article [86] studies to what extent an attacker can learn a controlled system to subtly damaging it, and provide conditions under which the controller can proactively detect and mitigate the attack; the data-driven approach in [127] proposes to use Graph Neural Network combined with suitable Gaussian filtering to identify trustworthy communications; survey [150] recaps strategies, and proposes new problems, to broadly scoped resilience in control multi-robot systems.

Finally, a different perspective is offered by game theory. Here, rather than dealing directly with an optimization problem, researchers investigate optimal strategies for both attacking and attacked agents in a game-like fashion, to evaluate how dangerous an attack could be and possible worst-case countermeasures. Examples of this line of work are [11], which addresses uncertain attacker behavior; paper [74], that studies the effect of random and stubborn agents in stochastic learning games; works [99], [101], that deal with optimal strategies when attacks are formulated as zero-sum games, possibly with more information available on the attacker side.

1.2 Novel Contribution

The literature review in Section 1.1 highlights that, even though long-standing research effort have been devoted to optimal design strategies for Networked Control Systems, some aspects are difficult to analyze and heuristics-based solutions and techniques are often used to support challenging design aspects.

The original contribution in this thesis is threefold and attempts to tackle such

Table 1.1: Outline of established literature and novel contribution proposed in the thesis.

	Chapter 1	**Chapter 2**	**Chapter 3**
State of the art	Sensor selection Sensor scheduling Edge Computing Computation offloading	Structured controller Performance *vs.* complexity Regularization for Design	Mean Subsequence Reduced Trust/trusted nodes Non-cooperative games
Trade-off	Latency *vs.* accuracy	Distributed *vs.* centralized architecture	Competition *vs.* collaboration
Idea and result	Estimation performance is optimized by nontrivial local processing and number of sensors	Closed-loop control performance is optimized by nontrivial distributed architecture	Enforcing some competition among agents can enhance resilience in collaborative tasks

challenges from a more quantitative perspective. The core, unifying idea proposed and investigated in the following is that effective design of Networked Control Systems must take into account the presence of nontrivial trade-offs, which may depend on system dynamics, on available resources, and on the task at hand. Even though the specific trade-off may vary according to the scenario, it is important to note that I do not deal with trade-offs *between performance and resources*, which is typical of both literature and practical design concerns: on the contrary, and this is key for all results, I am interested in trade-offs *of performance over resources*: namely, to find an allocation of available resources that maximizes performance of a control task. While this goal can be trivially solved in some cases, the previous literature review shows that rigorous solutions are not always available, and intuition is easily followed in state-of-the-art design methodologies.

At this point, it is important to correctly set the reader's expectations: because the main goal of the thesis is to investigate the fundamental nature and design trade-offs arising from the structure of a Networked Control System, some realism is sacrificed in favor of modeling assumptions that allow to achieve meaningful analytical results and hence intuition about system behavior and features of an optimal design. Nonetheless, I believe that this approach is necessary in order to start digging into partially uncovered and shady facets of this class of systems, whereby future developments will progressively embed realistic model assumptions and refine the preliminary results presented here. It is also important to bear in mind that the scenarios investigated in the following require careful attention to performance trade-offs generated by involved resources: clearly, other situations may be characterized by far more trivial behaviors and optimal design choices.

Table 1.1 outlines the topics explored in the next chapters, summarizing available literature, key trade-offs which inspired research in those topics, and novel contributions proposed to advance the state of the art with respect to conventional design methods. In the following, I elaborate on and describe in detail the contents of Table 1.1.

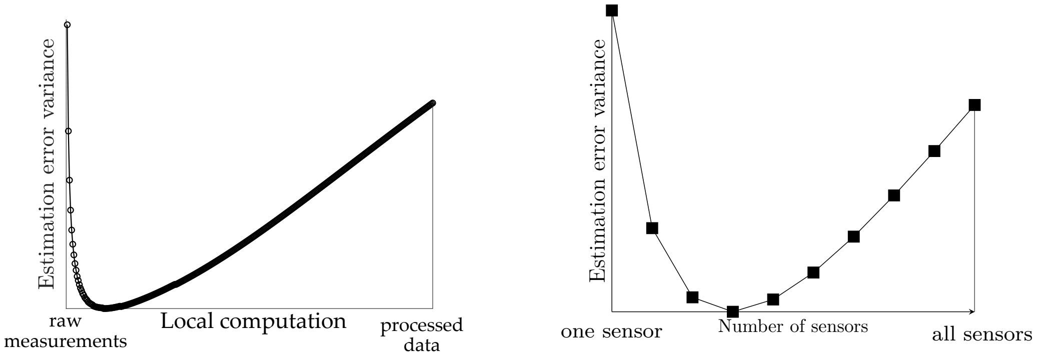


Figure 1.2: Under nontrivial computation latency, monitoring tasks for time-critical application exhibit a performance trade-off: in this thesis, I show that the estimation error variance has a nontrivial point of minimum with respect to both local processing at nodes (left) and number of deployed sensors (right). See Figures 2.6 and 2.11 and Fig. 2.8 for details.

1.2.1 Sensing design

As showed in Section 1.1.1, design of sensors and data processing is grounded in two main conventions: first, as many sensors as possible should be placed to provide a rich description of the signal of interest; second, local processing should be preferred not to overload few network nodes. However, these heuristics do not take into account the trade-off between computation latency and accuracy of both local processing on-board smart sensors and data fusion and aggregation of sensory data at network servers. The latter aspect may be particularly critical if smart sensors send data to a resource-constrained common node (*e.g.,* microcontroller or edge server) that processes all sensory measurements. For example, self-driving cars are forecast to carry many sensors supplying tons of environmental data online, which need to be suitably analyzed to ensure efficient and safe driving.

The first contribution of this thesis targets optimal sensing design for time-critical dynamical systems in the presence of computation and communication constraints: in particular, the latter impact the computation-communication trade-off argued in Section 1.1.1. Towards this goal, I follow a route along three milestones.

The first is mathematical models for latency-accuracy and computation-communication trade-offs that are used to rigorously formulate a sensing design problem that explicitly takes into account computation and communication latency and data accuracy affecting overall system performance. In particular, I focus on optimal estimation of a linear time-invariant system, whose performance is characterized by the covariance matrix of the estimation error. This is essentially the first work that addresses these elements together with system dynamics into an optimal design, whereas classical sensor selection and co-design hardly considers accuracy-dependent processing delays, and works on networks of computational-constrained devices do not usually address effects on dynamics and control performance.

The second milestone consists in rigorous analysis and design for a simple class of systems, namely continuous-time sensor networks composed of identical sensors. While this model is extremely simplified, it allows to achieve relevant analytical insight by virtue of formal results, that somehow reverse common design conventions: that is, (i) there exists an optimal amount of sensors which need not be the maximum, and (ii) there exists an optimal, nontrivial amount of local data processing at sensors. Fig. 1.2 visualizes these findings where performance metric is the steady-state error variance of an optimal estimator.

Finally, more generic systems, such as discrete-time dynamics with heterogeneous sensors (possibly multi-rate), are addressed to both reinforce and investigate consistency of analytical insight achieved previously. In particular, being the problem at hand combinatorial in nature, I propose greedy algorithms to decide both on the subset of available sensors to be used and on the amount of local data processing performed by each chosen sensor. Importantly, these algorithms leverage formal results obtained with the simple class of systems mentioned at the second milestone to tackle the much more challenging problem with heterogeneous sensors. The numerical results obtained in simulation are consistent with the analytical ones, corroborating the intuition that accounting for computation and communication latency in time-critical systems can crucially affect the sensing design, and in particular may disrupt classical heuristic conventions.

1.2.2 Controller Architecture Design

Finding efficient controllers with distributed architecture is a hard problem in general, and only in few special cases admits a convex formulation which ensures that the optimal controller can be efficiently computed [78]. Even in such special cases, communication non-idealities are usually not quantitatively addressed, and design methods proposed in the literature (see Section 1.1.2) use heuristic penalty terms to trade controller complexity for control performance. The intuition is that reducing the former, in particular by selecting only a few communication links, can enhance feasibility in a broad sense, for instance by dropping communication overhead or cost of deploying and maintaining links, at the cost of degrading performance, that benefits from dense information exchange.

However, one key problem is that the performance metric used for design, *e.g.,* $\mathcal{H}_2$-norm of the system, does not include any non-idealities of the dynamics, leaving the system

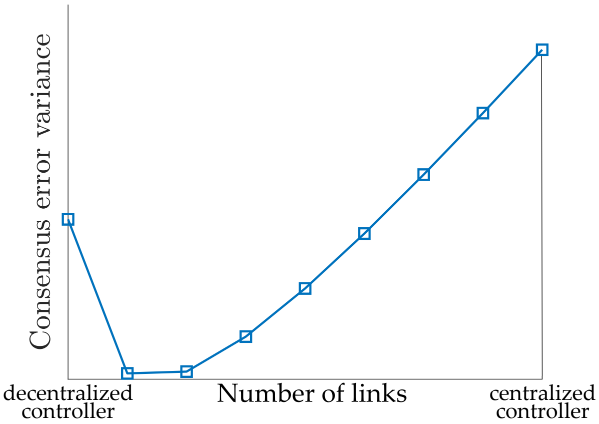


Figure 1.3: When communication latency increases with the total number of interconnections, a fundamental trade-off affecting closed-loop performance arises: in this thesis, I show that the optimal controller architecture is in general distributed. See Fig. 3.1 for details.

designer with the challenging task of suitably tuning regularization hyperparameters or other knobs whose physical interpretation is hard if not impossible. On the other hand, incorporating realistic dynamics in the computation of such cost functions is highly nontrivial, making the resulting trade-off lean towards simplified models that can be used for practical numerical design.

The second contribution of this thesis is an attempt to explore the other side of the wall, namely, to see what happens and to which extent the design can be pushed when computing the exact cost function with non-idealities in the dynamics. Specifically, the focus is on communication delays caused by wireless channel, with the key assumption that more communication links cause longer delays across the whole network. Before summarizing the main results, it is worth mentioning that studies on distributed controllers affected by communication or feedback latency have been extensively done. However, most of these consider structured controllers and focus on computing efficient control parameters without optimizing the architecture. For examples, works [48], [135], [231], [232] address consensus for systems with input delays, while [38], [63], [158], [177], [221] is concerned with stability and performance in more general control problems. On the other hand, the narrow body of work [204], [205] addresses the presence of architecture-dependent communication delays for deterministic consensus dynamics: while the spirit of those works is conceptually related to the aim of this thesis, both modeling assumptions, setup, and results are different.

Under the assumption that communication delays increase with the overall number of links, a fundamental performance trade-off for mean-square consensus in undirected graphs can be shown: there exists an optimal amount of communication links that minimizes the consensus error variance, which is in general smaller than the maximum amount of links (all-to-all communication). Fig. 1.3 illustrates this result, where the cost induced by closed-loop control on the y-axis is plotted against the total number of links within the network. This behavior is in sharp contrast with the widespread convention that increasing the number of links improves control performance. In particular, a near-optimal design allows to analytically prove that the consensus error variance can be decomposed into two additive monotone costs: one is decreasing with the number of network links and reflects the benefit of dense information flow on nominal control performance; the other is increasing and encodes the negative contribution of delays on controlled dynamics, and crucially bends the overall cost into a non-decreasing curve with a nontrivial point of minimum. Also, convexity of the resulting optimal control design problem is exploited to numerically show that the same is true also for the optimal controller, reinforcing the analytical findings.

1.2.3 Resilient Consensus

As shown in Section 1.1.3, standard approaches in resilient consensus and resilient control literature are essentially based on filtering the received messages and trying to guess, and possibly actively detect, which neighbors are to be trusted and which ones are more likely to be misbehaving. This paradigm is indeed intuitive and resembles what a human would most likely do in practice if they had to make rational decisions with information which may be partially wrong. On the other hand, collaborative and cooperative tasks benefit from enhancing collaboration among normally behaving agents. Hence, a classical trade-off arises that tries to strike a balance between the amount of information that is trusted and used in local updates and the amount of information that is discarded because potentially dangerous. However, as explained in Section 1.1.3, optimally solving this trad-off is far from trivial, and design problems may easily arise in that strong theoretical guarantees are usually hard to obtain.

More in general, a point to make about the literature in the field of resilient Networked Control Systems is that designed methods are typically tailored to specific scenarios, or built as ad-hoc strategies for particular applications. This paradigm is surely effective, but methodologies and ideas might need to be re-elaborated depending on the task, which could be avoided if a high-level design framework were available. This thesis proposes a small step towards a possible *unifying resilient approach*, through a novel viewpoint which is inspired by the theory of non-cooperative games. The basic idea is that, in the presence of (unknown) adversaries, normally behaving agents should partially collaborate with

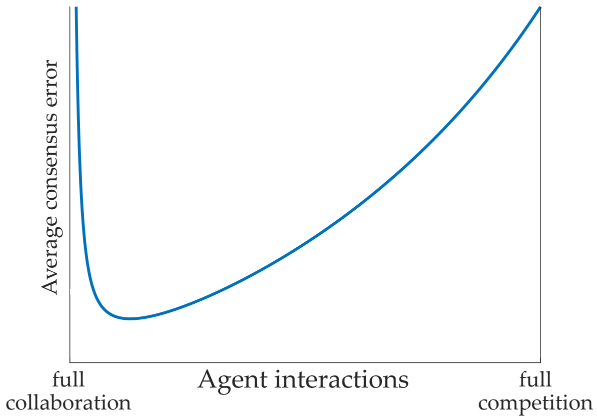


Figure 1.4: In the presence of unknown adversaries, a collaborative task can benefit from competition enforced among agents. In this thesis, I show that the cost associated with resilient average consensus exhibits a nontrivial point of minimum, which corresponds to a specific degree of "competitiveness" in the update rule. See Fig. 4.1 and 4.7 for details.

their neighbors, to work together towards the predefined global goal, but also partially be selfish, in order not to be deviated far off the nominal behavior that would ensure to reach the goal under ideal conditions (*i.e.,* no attacks). In the following, I will refer to such two contrasting attitudes as "collaboration" and "competition", respectively. To draw a pictorial parallel, if an agent following a filtering-like algorithm can be compared to a spy trying to guess which collaborators are traitors to dump, within the proposed framework an agent would resemble a spy keeping in touch with all their potentially untrustworthy collaborators, but only partially trusting each of them. Intuitively, this approach raises a performance trade-off: on the one hand, trusting neighbors too much may yield poor performance, because adversaries can easily induce misleading actions; on the other hand, if an agent does not trust others at all, any task requiring network cooperation will fail by definition. This is pictorially depicted in Fig. 1.4, which shows that the optimal strategy to minimize the cost of a global task (here, average consensus) features a specific level of competition among agents. Hence, an effective local rule should be able to optimally trade collaboration for competition, which may be nontrivial in the presence of misbehaving agents.

While I believe that the proposed framework has the potential to be used in various scenarios, the research work developed in this thesis is preliminary and explores a competition-based approach within the narrow domain of resilient average consensus. In particular, the third contribution that I propose is a suitable model to formally quantify competition and collaboration in agent's updates, which is analyzed it in order to derive performance bounds for resilient consensus. In particular, I use the celebrated Friedkin-Johnsen model as an alternative update protocol, where the presence of a real-valued tunable parameter allows to smoothly transition from pure collaboration, which coincides with standard consensus dynamics, to pure competition, which reduces to agents not moving from their initial condition. The proposed result, confirmed by both analytical and numerical results, is that the optimal parametrization that minimizes average consensus error corresponds to a hybrid strategy, whereby each normally behaving agent trusts its neighbors only partially, as shown in Fig. 1.4. Given the analytical focus, design of an efficient parameter in the update rule under the realistic scenario where attacks are unknown has not been considered yet. However, the performed analysis provides important insight that can be exploited in follow-up work on design aspects. Besides, I perform some comparisons with state-of-the-art W-MSR [94] and recently presented SABA [50] in simulation, showing that the proposed approach can perform better when r-robustness properties do not hold. Moreover, a preliminary heuristic investigation is devoted to the impact of communication network topology on performance of resilient consensus with the proposed protocol, whereby I show that both dense connectivity and degree balance (assuming undirected networks) are beneficial for resilience.

1.3 Organization of the Thesis

Along the lines of this Introduction, the thesis is organized in a modular structure that addresses each explored topic separately, as depicted in Fig. 1.5.

In Chapter 2, I present a novel sensing design in the presence of computation and communication constraints. In particular, I propose a model for a network of smart sensors transmitting data to a base station in Section 2.1, expose analytical results for a simple class of systems in Section 2.2, design greedy selection algorithms for the general

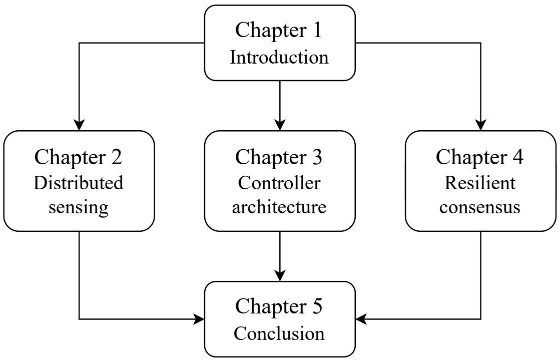


Figure 1.5: Flowchart of chapters in the thesis.

case in Section 2.3, and showcase numerical experiments in Section 2.4.

The material exposed in Chapter 2 is presented in the following papers:

[19] L. Ballotta, L. Schenato, and L. Carlone, "From sensor to processing networks: Optimal estimation with computation and communication latency," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 11 024–11 031, 2020, 21st IFAC World Congress;

[18] L. Ballotta, L. Schenato, and L. Carlone, "Computation-communication trade-offs and sensor selection in real-time estimation for processing networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2952–2965, 2020.

In Chapter 3, I present results on design of distributed controllers in the presence of communication latency. More in detail, I show how to compute stability conditions and cost function for the considered optimal control design problem with continuous- and discrete-time dynamics in Section 3.2 and Section 3.3, respectively, analyze the optimization problem associated with the control design in Section 3.4, and illustrate analytical and numerical results Section 3.5.

The material exposed in Chapter 3 is presented in the following papers:

[15] L. Ballotta, M. R. Jovanović, and L. Schenato, "Optimal network topology of multi-agent systems subject to computation and communication latency," in *Proc. Mediterranean Conf. Control Autom.*, 2021, pp. 249–254;

[14] L. Ballotta, M. R. Jovanović, and L. Schenato, "Can decentralized control outperform centralized? The role of communication latency," *arXiv e-prints*, no. arXiv:2109.00359, Jul. 2022, (submitted to IEEE Control Netw. Syst.).

In Chapter 4, I expose a novel approach to resilient consensus based on the Friedkin-Johnsen model. Specifically, I motivate and explain the model in Section 4.1, present analytical and numerical results respectively in Section 4.2 and Section 4.3, heuristically investigate impact of the communication graph in Section 4.4, and perform comparative simulations with other methods in the literature in Section 4.5.

The material exposed in Chapter 4 is presented in the following papers:

[13] L. Ballotta, G. Como, J. S. Shamma, *et al.*, "Competition-based resilience in distributed quadratic optimization," in *Proc. IEEE CDC*, (to appear), 2022;

[12] L. Ballotta, G. Como, J. S. Shamma, *et al.*, "Can competition outperform collaboration? The role of malicious agents," *arXiv e-prints*, no. arXiv:2207.01346, Jul. 2022, (submitted to IEEE Trans. Autom. Control).

Conclusions and potential avenues for future research are finally drawn in Chapter 5.

2

Sensing Design under Computation Latency

Over the last decade, Networked Control Systems have been steadily integrating with the novel paradigms of Edge and Fog Computing. The latter aim to push computational tasks towards the edge of the network, both to boost fast decision-making on-board network nodes (*e.g.,* smart sensors or autonomous robots) and to alleviate computational burden of central workstations and cloud servers. This paradigm shift is enabled by two key factors. On the one hand, deployment of increasingly powerful communication protocols, such as 5G, carries the promise of further enhancing communication capabilities and scale of network systems, which may host thousands, if not millions, of densely distributed devices featuring low-latency and ultra-reliable communication. On the other hand, advances in electronics, such as embedded GPU-CPU systems and dedicated hardware for embedded systems, have started to enable sophisticated computation and decision-making tasks on-board small devices, which can (partially or totally) share the workload of central severs.

As a matter of fact, most edge and lightweight devices operating in Networked Control Systems are still limited compared to hardware capabilities of powerful cloud or edge servers. In particular, so-called "smart sensing", which refers to both sensing and data processing capabilities on-board network nodes, is typically affected by constrained hardware resources which entail a *latency-accuracy trade-off*: while raw measurements are immediately available for monitoring or control tasks (up to acquisition time needed to physically collect sensory data), locally processing data on smart sensors takes non-negligible time, so that refined measurements can be used after additional *processing delay* from acquisition. More in general, I consider nodes executing *anytime algorithms* [229], *i.e.,* routines whose performance (quality of output values) improves with runtime, such as typical optimization algorithms whose final precision improves with the number of iterations. Classical applications of this class of algorithms can be found, for example, in real-time control, optimization, or combinatorial problems such as the well-known Travel

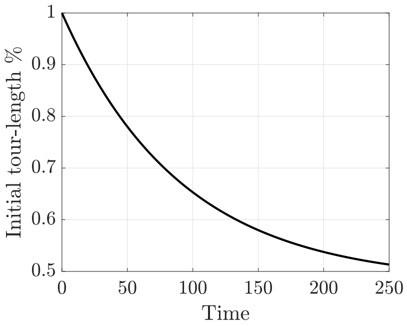


Figure 2.1: Randomized Tour Improvement is a classical greedy algorithm which approximates the optimal tour for the traveling salesman problem, shortening an initial route. Adapted from [229, Fig. 3].

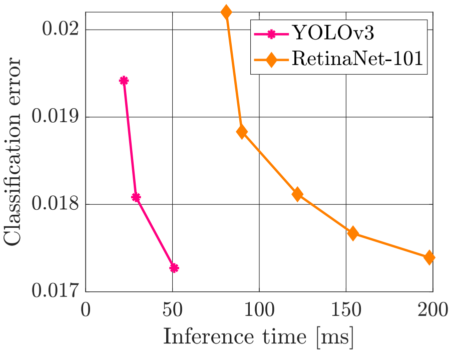


Figure 2.2: YOLO and RetinaNet are Neural Networks that can trade runtime for classification accuracy (errors on y-axis are computed as the inverse of mAP-50 scores). Adapted from [157, Fig. 3].

Salesman Problem Fig. 2.1. Anytime algorithms are popular in real-time control, computer vision, and robotic applications: for instance, papers [5], [157] study resource-aware Neural Networks whose complexity (proportional to accuracy) can be traded for inference time, as depicted in Fig. 2.2; work [91] adapts image-processing filters to real-time tasks by varying their kernels; reference [160] proposes a learning-based adaptive image compression; and paper [83] studies a planning algorithm that asymptotically converges to the optimal solution. Moreover, the concurrent availability of both efficient wireless connections and low-power processing hardware raises a *computation-communication trade-off.* In particular, data processing typically also entails some form of compression, so that it is nontrivial whether a node should send raw data, with negligible local computation but higher communication delay, or process the data locally, obtaining more accurate and compact information to be transmitted in short time. Typical examples are found in vision-based tasks: for instance, geometric perception algorithms used by robots can compress raw camera frames or 3D point clouds to low-dimensional feature vectors [70], which can be quickly transferred over a communication channel as opposed to heavy multimedia data.

Such computation and communication latency affecting sensory data may critically impact performance of a control application. In order to ground the discussion, we consider the case where a set of smart sensors samples a time-critical signal of interest and transmit collected information to a base station over a wireless channel. To stress the computational task deferred to network nodes, in the following we refer to this specific class of systems as *processing networks.* Figure 2.3 provides an example of this scenario in the realm of multi-robot applications: it depicts a network of drones collecting visual

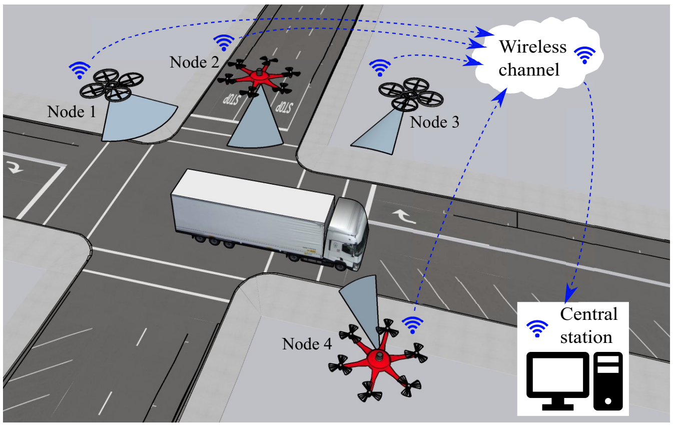


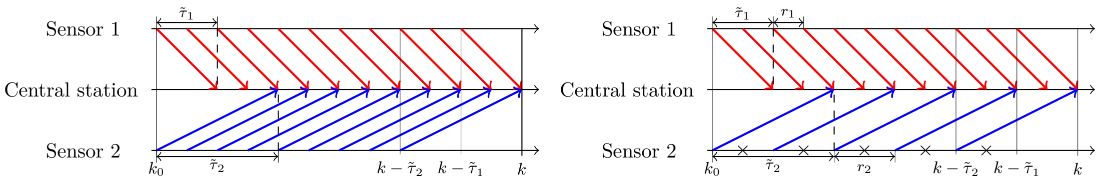
Figure 2.3: Example of *processing network*: drones track a moving vehicle in the presence of computation and communication constraints. Each drone can preprocess the acquired images before transmitting to a central station.

data of a truck on the road (*e.g.*, through a camera image stream) and transmitting them to a fusion station (on the bottom right), which is in charge of tracking the vehicle overtime. Importantly, smart sensors may have heterogeneous resources: for instance, the two hexarotors (nodes 2 and 4) might have powerful onboard GPU-CPU systems, while the quadrotors (1 and 3) may feature limited processing hardware. This presence of heterogeneous sensory and computational resources across the network raises a challenging design problem, as optimally allocating such resources is nontrivial. For example, some sensors might prefer sending raw data and incur larger communication delays, while other might process the collected measurements on-board. These choices will impact the quality of the truck state estimate: larger computational and communication delays will introduce more uncertainty, hindering the tracking task. In particular, overlong delays affecting data from one of the drones might induce misleading control commands by the base station, whose imprecise estimate of the truck state may cause the drone to steer trajectory and lose sight of it.

Figure 2.4 schematically shows the base station receiving outdated information. As discussed in the following, such delayed and heterogeneous data reception induces a challenging sensing design problem, which is the main matter of investigation of this chapter.

I develop the proposed contribution along three main axes.

First, I introduce a mathematical model for a processing network, where nodes (smart sensors) can perform local computation prior to communicate data to the base station (Section 2.1). I consider smart sensors in charge of observing the state of a dynamical system in the presence of communication and computational delays, which give rise to latency-accuracy and computation-communication trade-offs discussed above. In



(a) Homogeneous sensors (single rate). Time $\tilde{\tau}_i$ corresponds to delays caused by computation and communication.

(b) Multi-rate sensors. Crosses on the bottom axis indicate states sampled by sensor 1 but not by sensor 2.

Figure 2.4: Delayed data processing and transmission by two sensors in a processing network.

particular, the constrained nature of local processing is capture by using a computation-dependent measurement noise at each sensor, which is exploited to achieve analytical intuition on the optimal system design, as explained next. In this thesis, I report the model first develop in [18], while an alternative model tailored to sensors with limited storage and time-varying acquisition rate and local data processing is proposed in [16], [17].

Second, I use the proposed model to compute the amount of processing at each node in the simple case of a homogeneous network (all nodes carry the same computation) monitoring a continuous-time scalar linear system (Section 2.2), and prove that the optimal delay (*i.e.,* the allocated computational resources) can be analytically characterized. Also, it is possible to show that sending raw data is in general suboptimal under computation and communication latency. Furthermore, in the presence of computational delays at the base station – contrarily to conventional wisdom –, using more sensors might hinder estimation performance (intuitively: processing data from each sensor induces some computational delay, which adds up and introduces extra uncertainty in the estimation).

Finally, I consider a more realistic heterogeneous network monitoring a discrete-time multivariate linear system. Since using all sensors is not necessarily an optimal choice, I consider an extended problem formulation that both selects an optimal subset of sensors and decides the optimal processing at each selected sensor. Along the way to the proposed solution approach, I first show how to compute an estimation-theoretic cost function to be optimized by the processing network, possibly including multi-rate sensors (see Fig. 2.4b). Then, leveraging the formal analysis on homogeneous networks, I propose greedy algorithms to select sensors and allocate data processing. Numerical results (Section 2.4) show that (i) the proposed algorithms can indeed compute near-optimal policies, (ii) using all sensors is in general suboptimal, and (iii) the proposed policy can largely improve the network performance as opposed to cases that neglect the impact of delay contributions. Conclusions and discussion on future research directions are given in Section 2.5.

2.1 System Model and Problem Formulation

A *processing network* is a set of interconnected *nodes* that collect sensory data and leverage computation embedded on-board to locally process the data before communicating them to a central fusion center, also referred to as base station in the following.[1] For the sake of this work, I consider the case where the base station is tasked with obtaining an accurate estimate of the state of a time-varying phenomenon measured by networked nodes, in the presence of communication and computation latencies.

2.1.1 Anatomy of a Processing Network

Dynamical system. The monitored dynamical phenomenon is modeled through the following discrete-time linear time-invariant (LTI) stochastic system,

$$x_{k+1} = Ax_k + w_k, \tag{2.1}$$

where $x_k \in \mathbb{R}^n$ is the to-be-estimated state of the system at time k, $A \in \mathbb{R}^{n\times n}$ is the state matrix encoding the natural evolution of the state, and $w_k \sim \mathcal{N}(0,Q)$ is i.i.d. zero-mean Gaussian white noise with covariance matrix Q, which captures model uncertainties.

Smart sensors. The processing network includes a set of smart sensors (nodes) to collect measurements of the system state x_k. Each sensor is identified by label $i \in \mathcal{V}$, $\mathcal{V} \doteq \{1,\dots,|\mathcal{V}|\}$. After acquiring raw data, nodes may refine them via local processing. For instance, in the control application depicted in Fig. 2.3, each drone is a smart sensor that can process raw images to get more precise or high-level local measurements of the monitored system (position and velocity of the tracked vehicle). According to the available time and computational resources, a sensor may either run one of different tools deployed on-board (*e.g.,* one of the Neural Networks compared in Fig. 2.2) or adopt a specific instantiation of an available anytime procedure (*e.g.,* choosing the number of visual features to extract [70]) to obtain refined measurements. The data produced by all nodes in the network (possibly after local processing) are modeled as

$$z_k(\mathcal{T}_p) = Cx_k + v_k(\mathcal{T}_p), \qquad z_k(\mathcal{T}_p) = \begin{bmatrix} z_k^{(1)}(\tau_{p,1}) \\ \vdots \\ z_k^{(|\mathcal{V}|)}(\tau_{p,|\mathcal{V}|}) \end{bmatrix}, \tag{2.2}$$

[1] We often refer to the nodes as *smart sensors* to stress their sensing and computational capabilities.

where $z_k^{(i)} \in \mathbb{R}^{m_i}$ is the measurement collected at time k by the i-th node (starting from an initial time $k_0 \leq k$), $\tau_{p,i}$ is the *processing delay* associated with that node, C describes the state-to-output sensor transformation, and $v_k \sim \mathcal{N}(0, R)$ is i.i.d. zero-mean Gaussian noise.[2] The set $\mathcal{T}_p \doteq \{\tau_{p,i}\}_{i\in\mathcal{V}}$ collects all the processing delays, and vector z_k stacks the measurements collected by all nodes. As expressed by (2.2), we model a sensory measurement $z_k^{(i)}$ as dependent on processing delays $\tau_{p,i}$, which affect accuracy of processed data, through measurement noises $v_k^{(i)}$. In order to capture the anytime nature of local node processing discussed above, we model the covariance matrix (intensity) $R(\mathcal{T}_p)$ of the noise v_k as a decreasing function of the delays $\tau_{p,i}$: that is, the more time a node spends on local processing, the more accurate the output measurements are. In general, nodes with more powerful hardware induce a faster decrease of the uncertainty R, since they can quickly process a larger amount of sensory data. The noise model is formalized in Section 2.2.

Communication network. The nodes transmit processed data to the base station for sensor fusion. To account for channel unreliability, we associate with the measurement transmitted from the ith node at time k the binary random variable $\gamma_k^{(i)} \sim \mathcal{B}(\lambda_i)$, which denotes successful reception at the base station. Specifically, $1 - \lambda_i$ is the *packet-loss probability* associated with each transmitted measurement from the ith node, which we assume constant for the sake of simplicity. Further, we assume that $\gamma_k^{(i)}$ and $\gamma_\ell^{(j)}$ are uncorrelated if $k \neq \ell$ or $i \neq j$. Finite capacity is modeled as upper bound on the number of data packets per unit time, which induces a maximum number of nodes transmitting simultaneously.

Given limited bandwidth, data transmission induces a *communication delay* $\tau_{c,i}$ (potentially different for each node i). We consider two possible models for $\tau_{c,i}$, which is expressed as a function of processing delay $\tau_{p,i}$ to quantify the computation-communication trade-off.

Constant $\boldsymbol{\tau_{c,i}}$. The transmitted number of packets is fixed and does not depend on the amount of processing, but may increase with the dimension of the transmitted data.

Decreasing $\boldsymbol{\tau_{c,i}}$. If nodes *compress* the measurements, a longer processing yields fewer packets to transmit. In this case, nodes with more computational resources induce a higher compression rate, leading to a faster decrease of $\tau_{c,i}$.

Finally, the total delay to process and send data from the ith node to the base station is $\tilde{\tau}_i \doteq \tau_{p,i} + \tau_{c,i}$ (see Fig. 2.4).

[2] Raw data generated by the ith node are associated with the minimum value of $\tau_{p,i}$.

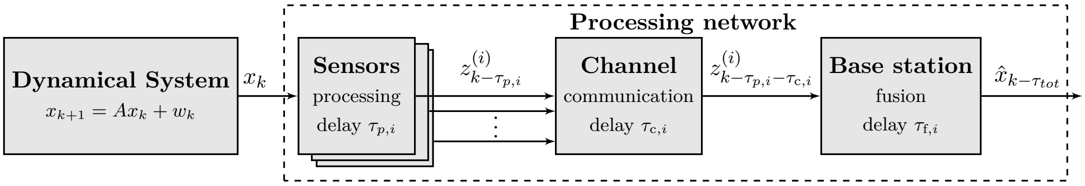


Figure 2.5: Block diagram of the processing network with processing, communication and fusion delays.

Base station. The central base station is in charge of fusing all received sensor data to compute a state estimate. Such centralized processing adds extra latency, referred to as *fusion delay* $\tau_{\mathrm{f,tot}}$, which is the sum of all delays $\tau_{\mathrm{f},i}, i \in \mathcal{V}$, required to process the data stream from each node i.

Akin communication, we model fusion delay $\tau_{\mathrm{f},i}$ as either constant or decreasing with processing delay $\tau_{p,i}$. The second model is related to the computation-communication trade-off: intuitively, the more processing is done at nodes, the less effort is needed for fusion, which receives more compact and lighter information. In particular, in the former case, fusion delays depends only on computational resources at the base station, while in the latter case they might also depend on the amount of data compression performed by nodes.

Figure 2.5 provides an overview of the processing network with the different latency contributions – due to node processing, communication, and centralized fusion. As highlighted in the figure, raw data goes through a number of operations, each inducing some delay. Therefore, the state estimate computed at time k will not include measurements acquired at time k, but only partially outdated measurements collected at times earlier than k through such delays. This is formalized in the following definition, that explicitly describes which data are actually available for real-time estimation at each time.

Definition 2.1.1. The *processed* dataset at time k is

$$\mathcal{Z}_k\left(\mathcal{T}_p\right) \doteq \left\{z^{(i)}_{\ell_i}(\tau_{p,i}) : \ell_i \in [k_0, k - \tilde{\tau}_i - \tau_{\mathrm{f,tot}}], \gamma^{(i)}_{\ell_i} = 1\right\}_{i\in\mathcal{V}}. \tag{2.3}$$

In words, the processed dataset includes all correctly received measurements *except* the most recent ones collected during communication and data processing, *i.e.*, during the latest $\tau_{p,i} + \tau_{\mathrm{c},i} + \tau_{\mathrm{f,tot}}$ timestamps.

2.1.2 Optimal Estimation in Processing Networks

In this section I first motivate the interest in optimizing the amount of processing at each node and the need to select a subset of nodes. Then, I provide a suitable metric to measure estimation performance. Finally, these elements are put together to formulate the problem of optimal estimation in processing networks (Prob. 2.1.2).

Processing selection. While sensory data might be received and fused with some (computation and communication) delay, a time-critical system needs a real-time, accurate state estimate at current time k. This entails fusing sensor information $\mathcal{Z}_k(\mathcal{T}_p)$ (partially outdated, due to the computation and communication delays) with the open-loop system prediction in (2.1). These delays create a nontrivial trade-off: is it best to transmit raw sensor data and incur larger communication and fusion delays, or to perform more processing at the edge and transmit more refined (less noisy and more compressed) information? For instance, consider again Fig. 2.3 where robots compute local estimates from images. Consider the case in which local estimates of the truck state are computed using local features extracted from camera images, as common in geometric computer vision [70]. Each extracted feature both enhances node-side accuracy and possibly reduces transmission and fusion latency. However, feature extraction entails some processing latency at the edge. A trade-off emerges: on the one hand, many features cause a delayed prediction; on the other hand, few provide poor accuracy. An *optimal estimation* policy has to decide the processing at each node in a way to maximize the final estimation accuracy.

Sensor selection. In addition to delays caused by local processing at nodes and communication channel, the fusion latency at the base station increases with the number of sensors transmitting data, by definition. As a consequence, the length of open-loop prediction required to compensate for the fusion delay increases with the number of nodes, hence adding more sensors does not necessarily improve performance. Therefore, in order to maximize the estimation accuracy, the network can also decide to use only a subset of available sensors $\mathcal{S} \subseteq \mathcal{V}$ (below we refer to those as *active* nodes), such that the state estimate is computed using only data from those sensors, $\mathcal{Z}_k\left(\mathcal{S}, \mathcal{T}_p^{\mathcal{S}}\right) \subseteq \mathcal{Z}_k\left(\mathcal{T}_p\right)$, where $\mathcal{T}_p^{\mathcal{S}}$ denotes the set of processing delays associated with active nodes.

Performance metric. In a state estimation problem, the performance can be measured as the Mean Squared Error (MSE) of a estimates, *i.e.*, $\mathrm{Var}\left(x_k - \hat{x}_k(\mathcal{T}_p^{\mathcal{S}})\right)$, where $\hat{x}_k\left(\mathcal{T}_p^{\mathcal{S}}\right) \doteq g\left(\mathcal{Z}_k\left(\mathcal{S}, \mathcal{T}_p^{\mathcal{S}}\right)\right)$ is the state estimate from an optimal estimator that uses the reduced processed dataset $\mathcal{Z}_k\left(\mathcal{S}, \mathcal{T}_p^{\mathcal{S}}\right)$. For linear systems with Gaussian noise, the Kalman filter is typically used, being the optimal MSE estimator. However, the optimal filter comes with the nuisance of time variance and dependence on the specific packet arrivals, and convergence analysis is not feasible (cf. [167], [175]). To overcome this problem, in the following I resort to the (suboptimal) filter with constant gains (*i.e.*, not depending on the arrival-sequence instance), and address the steady-state expected performance.

Problem formulation. Given the previous definitions, the problem of Optimal Estimation in Processing Networkcan be formalized as follows.

Problem 2.1.2 (Optimal Estimation in Processing Network)**.** Given system (2.1) with available sensor set $\mathcal{V}$ and measurement model (2.2), find the optimal sensor subset $\mathcal{S}$ (the *active* sensors) and processing delays $\mathcal{T}_p^{\mathcal{S}}$ that minimize the steady-state expected estimation error variance:

$$\underset{\substack{\mathcal{S}\subseteq\mathcal{V}\\ \mathcal{T}_p^{\mathcal{S}}=\{\tau_{p,i}\}_{i\in\mathcal{S}}\in\mathbb{N}^{|\mathcal{S}|}}}{\arg\min} \quad \mathrm{Tr}\left(P_{\infty|\infty-\tau_{\mathrm{tot}}}\left(\mathcal{T}_p^{\mathcal{S}}\right)\right), \tag{2.4}$$

where the total delay τ_{tot} is defined as

$$\tau_{\mathrm{tot}} \doteq \underbrace{\min_{i\in\mathcal{S}}\tilde{\tau}_i}_{\doteq\tilde{\tau}_{min}} + \underbrace{\sum_{i\in\mathcal{S}}\tau_{\mathrm{f},i}}_{\doteq\tau_{\mathrm{f,tot}}} \tag{2.5}$$

and the steady-state expected error covariance is given by

$$P_{\infty|\infty-\tau_{\mathrm{tot}}}\left(\mathcal{T}_p^{\mathcal{S}}\right) \doteq \lim_{k\to+\infty}\mathbb{E}\left[\mathrm{Var}\left(x_k-\hat{x}_k\left(\mathcal{T}_p^{\mathcal{S}}\right)\right)\right], \tag{2.6}$$

where the expectation is taken with respect to the sequence $\{\gamma_k^{(i)}\,\forall k\geq k_0,\forall i\in\mathcal{S}\}$.[3] The delay τ_{tot} accounts for the fact that, because of delays, the steady-state estimate relies on partially outdated measurements: $\tilde{\tau}_{min}$ is the time it takes to receive all processed data from the sensors (including the freshest data collected in $\mathcal{Z}_k(\mathcal{S},\mathcal{T}_p^{\mathcal{S}})$), while $\tau_{\mathrm{f,tot}}$ is the time it takes to fuse them at the base station.

Remark 2.1.3 (Parallel data collection *vs.* sequential fusion)*.* The delay $\tilde{\tau}_{min}$ is computed as the minimum over the active sensors, as these work in "parallel", while the fusion delay $\tau_{\mathrm{f,tot}}$ is additive, because in general the fusion center processes all data sequentially. Therefore, the latter is more sensitive to variations of computational delays. Besides, the fusion delay increases with the number of sensors, possibly limiting the network scalability.

[3] Packet-loss probabilities are assumed to be small enough so as that the steady-state estimator with constant gains exists.

Remark 2.1.4 (Comparison with sensor selection)*.* The problem formulation (2.4) differs from standard sensor selection, where each sensor comes with a cost and one aims at maximizing performance under cost constraints [77], [96], [200]. In particular, the focus here is purely on performance which is subject to latency-accuracy and communication-computation trade-off, binding the sensor selection to that of suitable processing delays. Therefore, in our setup, rather than associating a cost to sensors as resources, the penalty in using a sensor is captured by the amount of computation and delay it induces at the fusion station.

Remark 2.1.5 (LQG control and sensing co-design)*.* The optimal-estimation problem (2.4) readily extends to the co-design of sensing and LQG control. The latter is defined as the following dynamical program,

$$\underset{\substack{\{u_k\}_{k\geq k_0}\\ \mathcal{S},\mathcal{T}_p^{\mathcal{S}}}}{\text{minimize}} \quad \mathbb{E}\left[\sum_{k=k_0}^{\infty}\left(x_k^TQx_k+u_k^TRu_k\right)\right] \tag{2.7a}$$

$$\text{subject to} \quad x_{k+1}=Ax_k+Bu_k+w_k, \tag{2.7b}$$

where (2.7b) is the controlled system dynamics with control input u_k at time k. Indeed, it can be shown (see *e.g.,* [200]) that the above problem is equivalent to the following cascade,

$$\mathcal{S}^*,\mathcal{T}_p^{\mathcal{S},*}=\underset{\mathcal{S},\mathcal{T}_p^{\mathcal{S}}}{\arg\min} \quad \operatorname{Tr}\left(M_\infty P_{\infty|\infty-\tau_{\text{tot}}}\left(\mathcal{T}_p^{\mathcal{S}}\right)\right), \tag{2.8a}$$

$$u_k^*=-\left(B^\top S_tB+R_t\right)^{-1}B^\top S_tA\hat{x}_t\left(\mathcal{T}_p^{\mathcal{S},*}\right), \tag{2.8b}$$

where

$$M_\infty=A^\top S_\infty B\left(B^\top S_\infty B+R\right)^{-1}B^\top S_\infty A, \tag{2.9a}$$

$$S_\infty=Q+A^\top\left(BR^{-1}B^\top+S_\infty^{-1}\right)^{-1}A. \tag{2.9b}$$

In particular, the cost function in (2.8a) is equivalent to the one in (2.4) up to the constant matrix coefficient M_∞, which simply weighs the components of the estimation error covariance according to the control task. Hence, even though I focus on optimal estimation for the sake of simplicity, analytical results and selection algorithms presented in this chapter can be readily extended to the LQG optimal control scenario.

From now on, I will write $\tau_i=\tau_{p,i}$ and $\mathcal{T}=\mathcal{T}_p^{\mathcal{S}}$ for the sake of readability. Before designing algorithms to solve Problem 2.1.2, I perform an exact analysis on its continuous-time counterpart, which can be analytically solved when the set of sensors is fixed and homogeneous. Such simplified approach provides useful insights on the cost function in (2.4), which are used to tackle the more general case in Section 2.3.

2.2 Continuous-Time Analysis

In this section I consider a continuous-time scalar system monitored by a homogeneous network, composed of V independent sensors featuring identical local processing and delays. To achieve analytical insight, in the following I solve the reduced version of Problem 2.1.2 where only processing delays τ are optimized and the sensor set is fixed. The need for selection is motivated in Section 2.2.2 with a numerical example. Also, dealing with a continuous stream of sensory data, infinite channel capacity and reliable communication is first assumed. Such assumptions are relaxed in Section 2.3.2 where more realistic discrete-time dynamics and multivariate system state are addressed. The material presented in this section is developed in detail in [19].

Consider the following continuous-time scalar system,

$$dx_t = ax_t dt + dw_t \qquad dw_t \sim \mathcal{N}(0, \sigma_w^2 dt), \tag{2.10}$$

and the homogeneous-network model

$$z_t(\tau) = \mathbb{1}_V c\, x_t + v_t(\tau) \quad v_t(\tau) \sim \mathcal{N}\left(0, I_V \sigma_v^2(\tau)\right), \tag{2.11}$$

where a describes the state dynamics, w_t is the process noise, and σ_w^2 is its variance. Vector $\mathbb{1}_V \in \mathbb{R}^V$ stack all ones, and c and $\sigma_v^2(\tau)$ are scalars modeling the noisy state-output transformation of each sensor. The vector $z_t(\tau) \in \mathbb{R}^V$ collects all the measurements from the V sensors and $v_t(\tau)$ is the overall measurement noise, with covariance matrix $I_V \sigma_v^2(\tau)$.

The anytime nature of local processing at each node, qualitatively discussed in the previous section, is formally captured by modeling the measurement noise covariance $\sigma_v^2(\tau)$ as a decreasing function of the processing delay τ. Motivated by the estimation error variance of Least Squares, which is inversely proportional to the number of collected samples at each node, the following model is used,

$$\sigma_v^2(\tau) = \frac{b}{\tau} \qquad b > 0. \tag{2.12}$$

The coefficient b depends on the node parameters: on the one hand, nodes with large computational resources improve quickly their output accuracy, yielding a small b; on the other hand, if the collected raw data are heavy (e.g., images), refining them takes more time, inducing a larger b. Communication and fusion delays $\tau_{\mathrm{c}}(\tau), \tau_{\mathrm{f}}(\tau)$ are defined as

$$\text{constant}: \begin{cases} \tau_{\mathrm{c}}(\tau) \equiv \tau_c \\ \tau_{\mathrm{f}}(\tau) \equiv \tau_f \end{cases} \quad (2.13\text{a}) \qquad \tau\text{-varying}: \begin{cases} \tau_{\mathrm{c}}(\tau) = \frac{c}{\tau} \\ \tau_{\mathrm{f}}(\tau) = \frac{f}{\tau} \end{cases}, \quad (2.13\text{b})$$

where the delays are either fixed constants τ_c, τ_f as in (2.13a), or they are inversely proportional to the processing delay (with given coefficients c and f), as in (2.13b). Parameters τ_c, τ_f, c and f are assumed positive and known. Roughly speaking, both communication and fusion compression coefficients c and f increase with the dimension of the raw measurements. Conversely, sensors with more computational resources can compress faster and induce smaller coefficients.

Remark 2.2.1. While the models in (2.13b) are mainly used for mathematical convenience, in a real setup the compression functions might be learned or estimated from data, *e.g.*, [160].

In a homogeneous network, the total delay simplifies to (cf. (2.5) in the case that all nodes are active and have the same delays)

$$\tau_{\mathrm{tot}} = \tau + \tau_{\mathrm{c}}(\tau) + \tau_{\mathrm{f}}(\tau)V. \tag{2.14}$$

Importantly, the total fusion delay depends linearly on the sensor amount V. In such setup, Problem 2.1.2 simplifies to the following formulation, which focuses on the computation of the optimal processing delay (equal for all sensors).

Problem 2.2.2 (Optimal Estimation in Continuous-time Processing Network)**.** Given system (2.10) with V identical sensors and measurement model (2.11), find the optimal processing delay τ that minimizes the steady-state expected estimation error variance,

$$\operatorname*{arg\,min}_{\tau\in\mathbb{R}_+} \; p_{\infty|\infty-\tau_{\mathrm{tot}}}(\tau). \tag{2.15}$$

It turns out that 2.2.2 has a unique analytical solution, as formalized next.

Theorem 2.2.3 (Optimal processing for continuous-time homogeneous network)**.** *Consider the LTI system (2.10)–(2.11) with measurement noise variance $\sigma_v^2(\tau)$ as per (2.12), communication and fusion delays $\tau_c(\tau)$, $\tau_f(\tau)$ as per (2.13a) or (2.13b) and initial condition $x_{t_0} \sim \mathcal{N}(\mu_0, p_0)$. Assume $\hat{x}_t(\tau)$ is the Kalman-filter estimate at time t given measurements collected until time $t - \tau_{tot}$. Then, the steady-state error variance $p_{\infty|\infty-\tau_{tot}}(\tau)$*

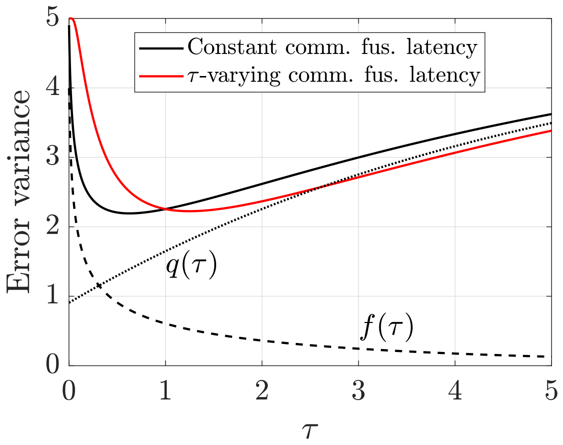


Figure 2.6: Representation of variance $p_{\infty|\infty-\tau_{\text{tot}}}(\tau)$.

is

$$p_{\infty|\infty-\tau_{tot}}(\tau) = \underbrace{e^{2a\tau_{tot}}p_{\infty}(\tau)}_{\doteq f(\tau)} + \underbrace{\frac{\sigma_w^2}{2a}\left(e^{2a\tau_{tot}}-1\right)}_{\doteq q(\tau)} \tag{2.16}$$

where

$$p_{\infty}(\tau) = \frac{\tilde{b}}{\tau}\left(a+\sqrt{a^2+\frac{\sigma_w^2}{\tilde{b}}\tau}\right) \qquad \tilde{b} \doteq \frac{b}{Vc^2} \tag{2.17}$$

with limits

$$\lim_{\tau\to 0^+} p_{\infty|\infty-\tau_{tot}}(\tau) = \lim_{\tau\to+\infty} p_{\infty|\infty-\tau_{tot}}(\tau) = \begin{cases} +\infty, & a \geq 0 \\ \dfrac{\sigma_w^2}{2|a|}, & a < 0 \end{cases} \tag{2.18}$$

and has a unique global minimum at $\tau_{opt} > 0$. Finally, when the delays $\tau_c(\tau)$ and $\tau_f(\tau)$ are constant, as per (2.13a), τ_{opt} satisfies

$$\frac{\sigma_w^2}{\tilde{b}}\tau_{opt}^3 = -a^2\tau_{opt}^2 + \frac{1}{4}. \tag{2.19}$$

Proof. See Appendix A.1. □

The proof exploits quasi-convexity of the expected variance $p_{\infty|\infty-\tau_{\text{tot}}}(\tau)$. Figure 2.6 illustrates the cost function with the two models for communication and fusion delays (black for constant and red for τ-varying) for an asymptotically stable system; for the former, the contributions due to estimation $f(\tau)$ and to process noise $q(\tau)$ as given in (2.16) are shown as dashed and dotted lines, respectively. The solid curves cross, the red one being lower for $\tau > 1$, suggesting that compressing data at fixed rate is convenient if the processing delay is kept below a certain threshold.

Equation 2.19 allows for a closed-form computation of τ_{opt} if model (2.13a) holds. In general, being the variance $p_{\infty|\infty-\tau_{\text{tot}}}(\tau)$ quasi-convex, a numerical solution can be computed efficiently. Optimal processing with alternative models for $\sigma_v^2(\tau)$ is discussed in Appendix A.2.

Example 2.2.4 (Brownian systems)*.* One interesting case arises when the system (2.10) describes a Brownian motion,

$$dx_t = dw_t. \tag{2.20}$$

In this situation, the optimal delay has a simple closed-form expression.

Corollary 2.2.5 (Brownian motion)*. Given system* (2.20) *and* (2.11) *and hypotheses as per Theorem 2.2.3, the steady-state expected error variance has the following expression,*

$$p_{\infty|\infty-\tau_{tot}}(\tau) = \underbrace{\sqrt{\frac{\tilde{b}\sigma_w^2}{\tau}}}_{f(\tau)} + \underbrace{\sigma_w^2\tau}_{q(\tau)}, \tag{2.21}$$

admitting the unique global minimum

$$\tau_{opt}^{B} = \sqrt[3]{\frac{\tilde{b}}{4\sigma_w^2}}. \tag{2.22}$$

The cubic root in (2.22) strongly reduces the parametric sensitivity of $\tau_{\text{opt}}^{\text{B}}$, which may intuitively help under model uncertainty.

2.2.1 Sensitivity of Optimal Processing

Based on (2.19), with constant delays (2.13a) the behavior of the optimal delay τ_{opt} can be analyzed as a function of the system parameters. In particular, σ_w^2 and $\tilde{b}$ do not act independently, so it is more interesting to focus on their ratio $\rho \doteq \sigma_w^2/\tilde{b}$.

Proposition 2.2.6. *Let τ_{opt} be the solution of* (2.19) *with $\tau_c(\tau)$, $\tau_f(\tau)$ as per* (2.13a)*; then, τ_{opt} is strictly decreasing with ρ and a^2.*

Proof. See Appendix A.3. □

On the one hand, Proposition 2.2.6 states that it is more convenient to reduce the processing for "unpredictable systems", characterized by fast dynamics or large process noise. On the other hand, if the sensor noise is large, it is better to further refine measurements, which explains why τ_{opt} grows with b. Also, since the parameter $\tilde{b}$ is

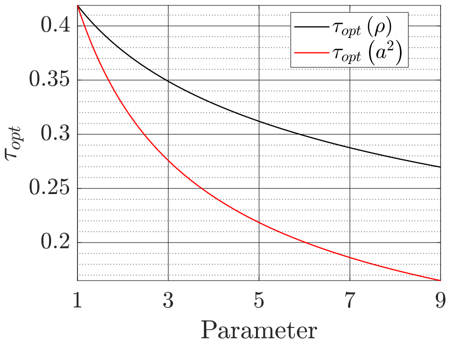


Figure 2.7: Optimal delay τ_{opt} as a function of ρ ($a^2 = 1$) and a^2 ($\rho = 1$).

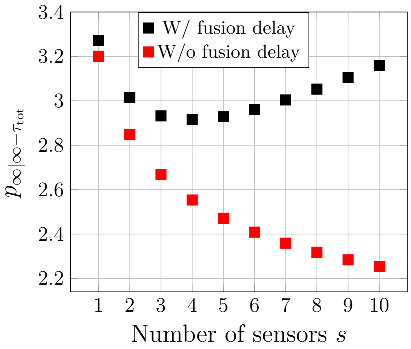


Figure 2.8: Variance $p_{\infty|\infty-\tau_{\mathrm{tot}}}(s)$ with fixed delays and varying number of sensors.

inversely proportional to the number of sensors V, then τ_{opt} also decreases with V: the more data are provided, the less processing is needed to extract accurate information. Figure 2.7 shows the typical behavior of τ_{opt} with respect to the system parameters.

Remark 2.2.7. (Insights from continuous-time scalar case) The analysis on continuous-time homogeneous networks yields two important insights. Firstly, the cost function is quasi-convex. This is exploited in Section 2.3.2 to design a descent strategy optimizing the processing delays of a given sensor subset. Secondly, using all sensors is not necessarily an optimal strategy, and – in the presence of fusion delays – using a proper subset of the available sensors leads to optimal estimation performance. This justifies the selection in Problem 2.1.2 and motivates design of the proposed greedy sensor selection.

2.2.2 Homogeneous Network: Performance *vs.* Number of Sensors

According to (2.14), the total delay τ_{tot} depends linearly on the sensor amount V, being the fusion delay additive with respect to sensors (cf. Remark 2.1.3). Therefore, if $p_{\infty|\infty-\tau_{\mathrm{tot}}}$ is seen as a function of the number of sensors $s \in \{1, \dots, V\}$ (with delays τ, $\tau_{\mathrm{c}}(\tau)$ and $\tau_{\mathrm{f}}(\tau)$ fixed), then $p_{\infty|\infty-\tau_{\mathrm{tot}}}(s)$ has the same structure of $p_{\infty|\infty-\tau_{\mathrm{tot}}}(\tau)$ when communication and fusion delays are constant, and can be minimized analogously (on discrete domain).

Figure 2.8 shows the expected estimation error variance as a function of the sensor amount. The red marks shows that the error decreases monotonically with the number of sensors in the absence of fusion delays. However, in the realistic case with non-negligible fusion delays (black marks), using more sensors might increase variance and hinder performance.

2.3 Latency-Aware Sensing Design

This section addresses the general discrete-time, multidimensional formulation in Problem 2.1.2. In discrete time, delays are expressed in time steps with respect to a sampling period Δ associated with discretization of the continuous-time system which produces dynamics (2.1).

Problem 2.1.2 cannot be solved analytically, due to its combinatorial nature. Also, the cost $\text{Tr}\left(P_{\infty|\infty-\tau_{\text{tot}}}(\mathcal{T})\right)$ cannot be computed in closed form in general, since it derives from the solution of a Riccati equation. To make things even more complicated, given a sensor subset, the structure of the cost function depends on how the delays are sorted.

To circumvent these issues, I propose greedy selection algorithms. This is done in two steps. In Section 2.3.1, I describe a procedure (based on [166]) to compute the cost function in (2.4) (and in particular the steady-state expected covariance $P_{\infty|\infty-\tau_{\text{tot}}}(\mathcal{T})$) for a given set of sensors and given processing delays. Then, algorithms to select sensors and to compute the optimal processing are presented in Section 2.3.2.

2.3.1 Computation of Expected Steady-State Error Covariance

This section shows how to compute the steady-state expected covariance for a given choice of the active sensors $\mathcal{S} \subseteq \mathcal{V}$ and given processing delays. For notational convenience and without loss of generality, active sensors are labeled as $\mathcal{S} = \{1, 2, \dots, s\}$, with corresponding processing delays $\tau_1, \tau_2, \dots, \tau_s$. Finally, active sensors are sorted as discussed below.

Assumption 2.3.1 (Sensor sorting)**.** Sensors in $\mathcal{S}$ are labeled according to $\tilde{\tau}_{i-1} \leq \tilde{\tau}_i$, for $i = 2, ..., s$ (cf. Fig. 2.4).

Assumption 2.3.1 states that, if $i < j$, the fusion station receives data from the ith sensor before than data from the jth sensor, and therefore sensor i transmits fresher information.

We now provide a procedure which, given sensor delays and parameters, computes the steady-state expected covariance $P_{\infty|\infty-\tau_{\text{tot}}}(\mathcal{T})$ (and hence the cost $\text{Tr}\left(P_{\infty|\infty-\tau_{\text{tot}}}(\mathcal{T})\right)$ in (2.4)), and is exploited by the first algorithm in Section 2.3.2 to assess the performance of sensor subsets. The following is *not* a closed-form – but rather an iterative – computation. I first outline the procedure and then provide an illustration of the procedure with $s = 3$ active sensors using Fig. 2.9.

We start by expanding the matrices that describe the measurement model (2.2) as

$$C = \left[C_1^T \dots C_s^T\right]^T \qquad , R(\mathcal{T}) = \text{diag}(R_1(\tau_1), ..., R_s(\tau_s)), \tag{2.23}$$

where we assume independent sensors with state-output matrix C_i and covariance $R_i(\tau_i) = {}^{b_i}/_{\tau_i} I_{m_i}$ (cf. (2.12)).

Before introducing the key result in Theorem 2.3.3 below, it is helpful to introduce some definitions associated with the Kalman filter with constant gains (cf. [167], [175]), which are used to state the theorem.

Definition 2.3.2. We define the following operations associated with the Kalman filter with constant gains and acting on the extended state estimate covariance matrix P.

Multi-step prediction with $\tau > 0$ steps:

$$\mathcal{P}^{\tau}(P) \doteq \underbrace{\mathcal{P} \circ \ldots \circ \mathcal{P}}_{\tau \text{ times}}(P), \quad \mathcal{P}(P) \doteq APA^{T} + Q. \tag{2.24}$$

Measurement update with data acquired at time $k - \tau_{\mathrm{f,tot}} - \delta$:

$$\mathcal{U}\left(P, \mathcal{T}_{\delta}\right) \doteq \left(P^{-1} + \tilde{\Gamma}(\mathcal{T}_{\delta})\right)^{-1}. \tag{2.25}$$

for any delay δ, where the information matrix of the processed data when the Kalman gains are constant is[4]

$$\tilde{\Gamma}(\mathcal{T}_{\delta}) = \sum_{i \in \mathcal{S}(\delta)} \lambda_i \left[\Gamma_i - \Gamma_i \left(\frac{P^{-1}}{1 - \lambda_i} + \Gamma_i \right)^{-1} \Gamma_i \right]. \tag{2.26}$$

In the previous expression, $\Gamma_i = C_i^T (R_i(\tau_i))^{-1} C_i$ is the information matrix of the ith sensor and $1 - \lambda_i$ is its packet-loss probability (for details about the derivation of $\tilde{\Gamma}$ with packet loss, see Appendix A.4). Recall that the update is restricted to the sensors from which measurements have been received by time $k - \tau_{\mathrm{f,tot}}$. Formally, this set of processed sensors is

$$\mathcal{S}(\delta) \doteq \{ i \in \mathcal{S} : \tilde{\tau}_i \leq \delta \}, \tag{2.27}$$

and their processing delays are in $\mathcal{T}_{\delta} \doteq \{\tau_i\}_{i \in \mathcal{S}(\delta)}$.

One-step KF iteration with data acquired at time $k - \tau_{\mathrm{f,tot}} - \delta$:

$$\mathcal{I}\left(P, \mathcal{T}_{\delta}\right) \doteq \mathcal{P} \circ \mathcal{U}\left(P, \mathcal{T}_{\delta}\right). \tag{2.28}$$

[4] All updates use the Kalman filter in information form to handle the fusion more easily. This is also useful if sensor measurements have infinite variance at some locations. Having independent sensors yields a nice expression for $\tilde{\Gamma}$, where each contribution is visible and disjoint from the others.

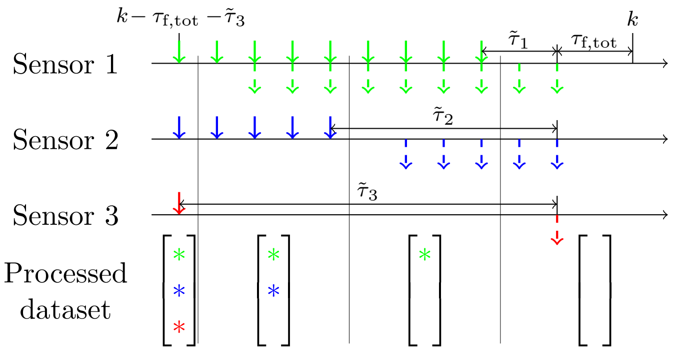


Figure 2.9: Estimation at time k. Solid and dashed arrows show acquired and received (by central station) data, respectively. Colored stars represent sensor data which are available in the processed dataset.

Multi-step KF iteration with data acquired in the time interval $[k-\tau_{\text{f,tot}}-\delta_i+1,\ k-\tau_{\text{f,tot}}-\delta_j]$ for any delays $\delta_i > \delta_j$:

$$\mathcal{I}^{\delta_i-\delta_j}\left(P,\mathcal{T}_{\delta_i-1}\right) \doteq \mathcal{I}\left(\dots\mathcal{I}\left(P,\mathcal{T}_{\delta_i-1}\right),\dots,\mathcal{T}_{\delta_j}\right), \tag{2.29}$$

where the one-step KF iterations may involve different subsets of active sensors, according to their delays.

Then the following result can be obtained.

Theorem 2.3.3. *Using the terminology and notation in 2.3.2, the cost function in* (2.4) *is given by the trace of*

$$P_{\infty|\infty-\tau_{tot}}(\mathcal{T}) = \mathcal{P}^{\tau_{pred}}\left(\mathcal{I}^{\tilde{\tau}_s-\tilde{\tau}_1}\left(P_\infty\left(\mathcal{T}\right),\mathcal{T}_{\tilde{\tau}_s-1}\right)\right), \tag{2.30}$$

where:

- $\tau_{pred} \doteq \tilde{\tau}_1 - 1 + \tau_{f,tot}$ *is the length of the multi-step prediction;*
- $\tilde{\tau}_s - \tilde{\tau}_1$ *is the time between oldest and newest processed data;*
- $P_\infty(\mathcal{T})$ *solves the ARE where all active sensors are considered,* i.e.,

$$P_\infty(\mathcal{T}) = \mathcal{I}\left(P_\infty(\mathcal{T}),\mathcal{T}_{\tilde{\tau}_s}\right). \tag{2.31}$$

Proof. See Appendix A.5. □

Algorithm 1 implements (2.30): line 1 solves the ARE (2.31); loop 3–5 computes the multi-step KF iterations, with jth iteration involving the subset $\{1,\dots,s-j\}$; line 6 computes the multi-step prediction. This procedure yields the desired steady-state expected covariance $P_{\infty|\infty-\tau_{\mathrm{tot}}}(\mathcal{T})$.

Algorithm 1 `computeCovariance` subroutine

Input: System (A, Q), state-output matrix C_i, noise covariance $R_i(\cdot)$, communication and fusion delays $\tau_{\mathrm{c},i}$, $\tau_{\mathrm{f},i}$ for each active sensor $i \in \mathcal{S}$, processing delays $\mathcal{T}$.

Output: Expected error covariance $P_{\infty|\infty-\tau_{\mathrm{tot}}}(\mathcal{T})$.

```
Compute solution $P_\infty(\mathcal{T})$ of ARE with all sensors;
$P \leftarrow P_\infty(\mathcal{T})$;
for sensor amount $i \leftarrow s-1$ down to 1 do
    multi-step KF iteration: $P \leftarrow \mathcal{I}^{\tilde{\tau}_{i+1}-\tilde{\tau}_i}\left(P, \mathcal{T}_{\tilde{\tau}_i}\right)$;
end for
multi-step prediction: $P_{\infty|\infty-\tau_{\mathrm{tot}}}(\mathcal{T}) \leftarrow \mathcal{P}^{\tau_{\mathrm{pred}}}(P)$;
return $P_{\infty|\infty-\tau_{\mathrm{tot}}}(\mathcal{T})$.
```

Figure 2.9 illustrates the procedure with $s = 3$ active sensors. At time $k - \tau_{\mathrm{f,tot}}$, the fusion station initiates the computation to produce an estimate at time k. When performing the fusion, the station has access to the data from all sensors collected until time $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_3$. However, due to the computation and communication delays, it will only have access to a subset of the sensor data after $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_3$. In particular, sensor 3 has the largest processing-and-communication delay $\tilde{\tau}_3$, and the data it collects after time $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_3$ will only be received at the fusion station after time $k - \tau_{\mathrm{f,tot}}$. The insight of Algorithm 1 is simple: this procedure computes the expected covariance until time $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_3$ (when all sensors are available), and then it collects the sporadic measurements collected after $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_3$ which arrived at the fusion station before time $k - \tau_{\mathrm{f,tot}}$ (*i.e.*, the ones from sensor 1 and 2 in the figure). In particular, accounting for all measurements collected until time $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_3$ leads to the steady-state expected error covariance $P_\infty(\mathcal{T})$ satisfying (2.31), while the subsequent measurements are captured by (2.30). In Fig. 2.9, sensor 3 only provides one measurement, and the following four estimates use sensors 1 and 2. Afterwards, also sensor 2 becomes outdated and the last measurement updates only involve sensor 1. After the processed dataset has been used, the current-state estimate is retrieved with an open-loop prediction compensating for the remaining delay (in this case induced by sensor 1 and fusion). In the figure, the sensor contribution to the state estimates overtime is highlighted with the matrix in the bottom row: at first, all sensor data are available (full bottom-left matrix), then some sensors become outdated, until no more sensor measurement is received and the state estimate must be propagated in open loop (empty bottom-right matrix).

Remark 2.3.4 (Cost computation with state augmentation). Equation 2.30 can be equiv-

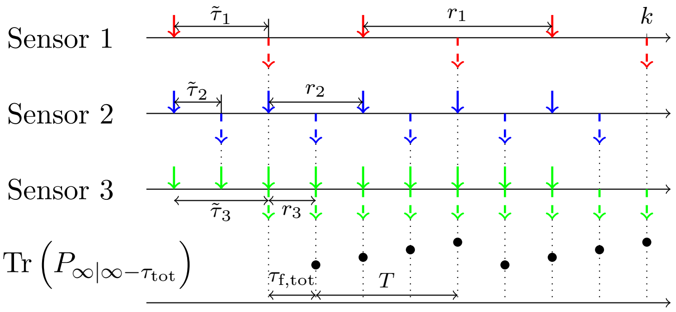


Figure 2.10: T-periodic cost with three multi-rate sensors. Solid arrows: sensor acquisitions; dashed arrows: data reception at central station.

alently written in a more compact way, by considering the augmented system with $\tilde{\tau}_s + \tau_{\mathrm{f,tot}}$ consecutive states and C and R having nonzero blocks according to processed data. The cost $\mathrm{Tr}\left(P_{\infty|\infty-\tau_{\mathrm{tot}}}(\mathcal{T})\right)$ would be retrieved by computing the steady-state expected covariance of the augmented-state-estimate error, and cropping the bottom-right block. However, this is numerically inefficient and does not exploit the specific structure of the problem.

Remark 2.3.5 (Adaptive selection). Algorithm 1 can be modified for an adaptive design, *e.g.,* to deal with the case when the system parameters change overtime, or if it is desired to schedule different sensors over time. The time-varying counterpart of Problem 2.1.2 (where the estimation error covariance is evaluated over a finite time horizon instead of steady state, as done in [200]) might be solved iteratively over suitable horizons, in an MPC-like fashion. In particular, line 1 can be substituted with online KF iterations (2.28) from time k_0 to $k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_s$, including all sensors in the updates with measurements. An alternative adaptive design based on Reinforcement Learning is proposed in [16], [17].

Remark 2.3.6 (Multi-rate networks). Algorithm 1 can also deal with systems where nodes have heterogeneous acquisition times r_i (see Fig. 2.4b). This case is quite natural in processing networks: for example, the drones in Fig. 2.3 may carry cameras with different frame rates. The corresponding information matrix for the ith sensor can be easily modeled in the setup adopted above as a time-varying matrix,

$$\Gamma_i(\tau_i, k) \doteq \begin{cases} C_i^\top (R_i(\tau_i))^{-1} C_i, & \text{if } k_0 = k \bmod r_i \\ 0_{n\times n}, & \text{otherwise,} \end{cases} \tag{2.32}$$

which can be readily used in Algorithm 1. Figure 2.10 shows a qualitative behavior of the cost with three multi-rate sensors.

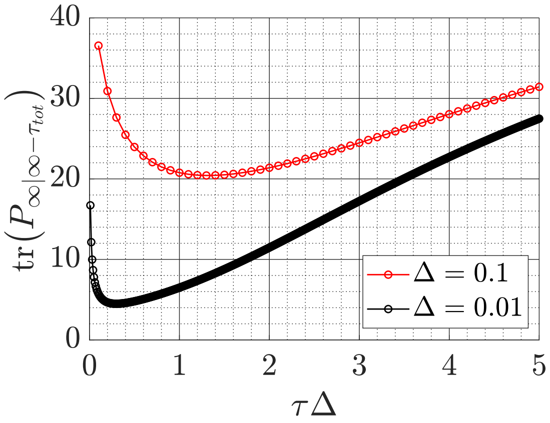


Figure 2.11: Cost function for a homogeneous network. The original continuous-time system has poles $\sigma(A) = \{-1, -0.1\}$ and $Q = 10I_2$.

Example 2.3.7 (Homogeneous network). After presenting a procedure to compute the cost function in the discrete-time case, this can be tested in the simple case of a homogeneous network with a single processing delay for all sensors, without addressing sensor selection. This allows establishing a connection with the formal setup in Section 2.2. The numerical simulations in Fig. 2.11 exhibit a quasi-convex behavior, similarly to the continuous-time counterparts in Fig. 2.6. This motivates the exploration of greedy "descend" methods that attempt to iteratively minimize the cost function.[5]

2.3.2 Algorithms for Sensing Design

Since Problem 2.1.2 cannot be solved analytically, it is tackled via a two-step greedy approach relying on the insights highlighted in Remark 2.2.7. On the one hand, motivated by the need of choosing an optimal sensor subset, the algorithm iteratively selects one sensor at a time, until the expected performance cannot be further improved. On the other hand, leveraging the intuition of a quasi-convex cost, the delays of each tentative subset are optimized via a dedicated subroutine. The next paragraph shows how the latter optimizes the delays for a given sensor set, and then I present the full iterative procedure.

[5] In the homogeneous case, a single processing delay needs to be computed, and the optimal one can be easily found by a brute-force search. However, the ultimate goal in Problem 2.1.2 is to tackle the general case of heterogeneous networks, where the optimization variable has high dimension.

Sensor-wise Descent for Selection of Processing Delays

A *sensor-wise descent* subroutine is used to compute near-optimal computational delays $\mathcal{T}^*$ for a given active set $\mathcal{S}$. The algorithm optimizes one delay τ_i at a time, by minimizing the one-dimensional problem associated with sensor i, with all other delays fixed. Algorithm 2 shows the subroutine steps. The to-be-returned delays and cost are initialized with the input delays $\mathcal{T}_I$ and the trace of the expected error covariance computed with $\mathcal{T}_I$, respectively (lines 1–2).[6] The outer loop between lines 3–16 optimizes the delay $\mathcal{T}^*[i] = \tau_i$ with a one-dimensional descent. For each delay, an exploratory iteration is first used to set the sign of the unitary stepsize α, according to the descent direction (line 4–9). Once α is chosen, the inner loop (lines 11–13) refines the processing delay: the descent direction is explored until a local minimum is found (condition 14). The best achieved delay is restored and saved in line 15. Figure 2.12 shows the cost function with three sensors: it looks quasi-convex, consistently with homogeneous networks. Figure 2.13 illustrates an execution of the Algorithm 2, which highlights its sensor-wise nature.

Algorithm 2 `sensorWiseDescent` subroutine

Input: System (A, Q), state-output matrix C_i, noise covariance $R_i(\cdot)$, communication and fusion delays $\tau_{\mathrm{c},i}$, $\tau_{\mathrm{f},i}$ for each active sensor $i \in \mathcal{S}$, initial delays $\mathcal{T}_I$.

Output: Near-optimal delay set $\mathcal{T}^*$, cost $\mathrm{Tr}\left(P_{\infty|\infty-\tau_{\mathrm{tot}}}(\mathcal{T}^*)\right)$.

```
 1: $p_{\min} \leftarrow \mathrm{Tr}\,(\texttt{computeCovariance}(A, Q, \mathcal{S}, \mathcal{T}_I))$;
 2: $\mathcal{T}^* \leftarrow \mathcal{T}_I$;
 3: for each $i \in \mathcal{S}$ do
 4:     Stepsize $\alpha \leftarrow -1$ ;                    // default: delay $\tau_i$ is decreased
 5:     $\mathcal{T}^*[i] \leftarrow \mathcal{T}^*[i] + \alpha$;
 6:     $p_{\mathrm{curr}} \leftarrow \mathrm{Tr}\,(\texttt{computeCovariance}(A, Q, \mathcal{S}, \mathcal{T}^*))$;
 7:     if $p_{\min} \leq p_{\mathrm{curr}}$ then
 8:         $\alpha \leftarrow +1$;                                  // delay $\tau_i$ is increased
 9:     end if
10:     repeat
11:         $p_{\min} \leftarrow p_{\mathrm{curr}}$;
12:         $\mathcal{T}^*[i] \leftarrow \mathcal{T}^*[i] + \alpha$;
13:         $p_{\mathrm{curr}} \leftarrow \mathrm{Tr}\,(\texttt{computeCovariance}(A, Q, \mathcal{S}, \mathcal{T}^*))$;
14:     until $p_{\min} \leq p_{\mathrm{curr}}$
15:     $\mathcal{T}^*[i] \leftarrow \mathcal{T}^*[i] - \alpha$;
16: end for
17: return $\mathcal{T}^*$, $p_{\min}$.
```

[6] The initial delays $\mathcal{T}_I$ are provided by the sensor selection algorithm exposed next.

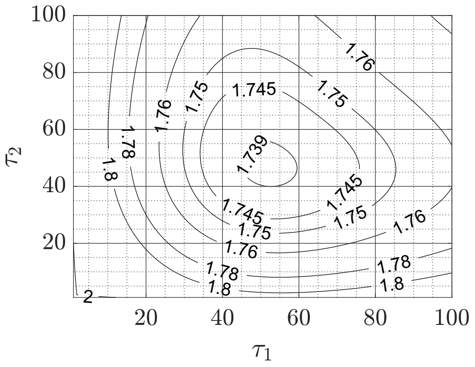


Figure 2.12: Cost-function levels with constant $\tau_{\text{c},i}$, $\tau_{\text{f},i}$ (τ_3 is fixed).

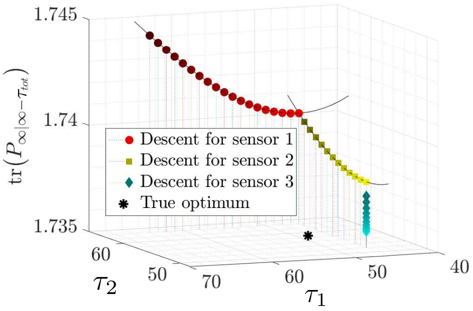


Figure 2.13: Visualization of `sensorWiseDescent` on cost function in Fig. 2.12 (τ_3 goes from 64 to 52): iterations go from darker to lighter marks.

Sensor Selection

This section presents the main procedure to solve Problem 2.1.2. A greedy algorithm iteratively selects one sensor at a time, as long as the cost can be decreased. For each tentative subset, processing delays are optimized as described in the previous paragraph.

Algorithm 3 shows the pseudo-code of the overall procedure. First, each available sensor i is considered in isolation and its optimal processing delay $\tau_{\text{opt},i}$ and corresponding variance are found via brute-force search (line 1). The to-be-returned sensor and delay subsets $\mathcal{S}^*$, $\mathcal{T}^*$ and the minimum cost $p_{\min}$ are initialized with the sensor achieving the minimum variance (lines 2–4). The outer loop 5–21 adds one sensor at a time to the selection $\mathcal{S}^*$, stopping when the cost hits a local minimum (no other sensor can be added to further reduce the cost) or all available sensors have been selected. The inner loop 7–16, given the current selection $\mathcal{S}^*$, builds the tentative subsets $\mathcal{S}_{\text{curr}}$ (line 8) by adding excluded sensors (`toTry`) one at a time. The near-optimal delays $\mathcal{T}_{\text{curr}}$ for the tentative set are initialized with the best delays obtained so far for the sensors in $\mathcal{S}_{\text{curr}}$ (line 9), *i.e.,* with the current near-optimal delays $\mathcal{T}^*$ for the already-selected sensors, and with the single-sensor optimal delay for the tentative sensor: intuitively, a "small" difference between subsets yields "small" differences between optimal delays. The sensor-wise descent is in charge of computing the near-optimal delays and cost for each subset (line 10). When a tentative subset hits a new minimum (line 11), the sensor `toTry` becomes `toSelect` (line 12), *i.e.,* it is the best candidate to be added to the selected subset. The temporary variable $\mathcal{T}^*_{\text{curr}}$ allows not to overwrite the delays used to initialize `sensorWiseDescent`. When all available sensors have been tried, the one `toSelect` (if any) and the new near-optimal delays are stored in the to-be-returned variables (lines 17–20).

Algorithm 3 Sensor selection

```
Input: System $(A, Q)$, state-output matrix $C_i$, noise covariance $R_i(\cdot)$, communication
    and fusion delays $\tau_{\mathrm{c},i}$, $\tau_{\mathrm{f},i}$ for each available sensor $i \in \mathcal{V}$.
Output: Near-optimal sensor set $\mathcal{S}^*$ and delay set $\mathcal{T}^*$.
Compute optimal delays $\tau_{\mathrm{opt},i}$ for one-sensor subsets $\{i\}$;
$\mathcal{S}^* \leftarrow$ one-sensor subset achieving minimum cost;
$\mathcal{T}^* \leftarrow \{$optimal delay $\tau_{\mathrm{opt},\mathcal{S}^*}$ for $\mathcal{S}^*\}$;
$p_{\mathrm{min}} \leftarrow$ minimum cost achieved by $\mathcal{S}^*$;
repeat
    toSelect $\leftarrow \emptyset$;
    for each sensor toTry $\in \mathcal{V} \backslash \mathcal{S}^*$ do
        $\mathcal{S}_{\mathrm{curr}} \leftarrow \mathcal{S}^* \cup \{\mathtt{toTry}\}$;
        $\mathcal{T}_{\mathrm{curr}} \leftarrow \mathcal{T}^* \cup \{\tau_{\mathrm{opt},\mathtt{toTry}}\}$;
        $[\mathcal{T}_{\mathrm{curr}}, p_{\mathrm{curr}}] \leftarrow$ sensorWiseDescent $(A, Q, \mathcal{S}_{\mathrm{curr}}, \mathcal{T}_{\mathrm{curr}})$;
        if $p_{\mathrm{min}} > p_{\mathrm{curr}}$ then
            toSelect $\leftarrow$ toTry;
            $p_{\mathrm{min}} \leftarrow p_{\mathrm{curr}}$;
            $\mathcal{T}^*_{\mathrm{curr}} \leftarrow \mathcal{T}_{\mathrm{curr}}$;
        end if
    end for
    if $\exists$ toSelect then
        $\mathcal{S}^* \leftarrow \mathcal{S}^* \cup \{\mathtt{toSelect}\}$;
        $\mathcal{T}^* \leftarrow \mathcal{T}^*_{\mathrm{curr}}$;
    end if
until $p_{\mathrm{min}} \leq p_{\mathrm{curr}}$ or $s = |\mathcal{V}|$
return $\mathcal{S}^*$, $\mathcal{T}^*$.
```

Remark 2.3.8 (Non-detectable subsystems)*.* Some costs for single-sensor subsets in line 1 may be infinite if pairs (A, C_i) are not detectable. If this holds for some sensors, these can be discarded from initialization. Otherwise, the latter may be replaced by the greedy selection of the minimum-cardinality, minimum-cost sensor subset ensuring detectability.

Remark 2.3.9 (Finite channel capacity)*.* The channel capacity can be handled by adding a termination condition in line 21 to stop the algorithm when the selected sensors "fill" the channel. A threshold $\bar{s} < |\mathcal{V}|$ may be fixed *a priori* to select at most $s \leq \bar{s}$ sensors (in this case, the structure of the algorithm ensures that the greedy-best $\bar{s}$-sensor subset is selected), or the bandwidth utilization of each selected sensor may be traced online during execution. As mentioned in Section 1.1.1, typical works in the telecommunication literature tackle limited channel capacity from a scheduling perspective. A unifying framework which merges these two approaches is possible, even though it leads to an intractable problem in general. Paper [195] extends the present work towards that direction, studying co-design of sensor local processing and communication scheduling when sensors observing decoupled dynamical system. Also, Algorithm 3 might be iteratively run over a finite time horizon if sensors need to be scheduled (cf. Remark 2.3.5).

Remark 2.3.10 (User-driven selection)*.* If the task needs specific sensory data (such as images or infra-red), Algorithm 3 can be customized accordingly, *e.g.,* forcing the initial subset (lines 2–4) to include the corresponding sensors.

2.4 Numerical Simulations

Inspired by Fig. 2.3, the proposed numerical experiments simulate a drone network in charge of tracking position and velocity of a ground vehicle. The system state $\boldsymbol{x} = [x\,\dot{x}\,y\,\dot{y}]^T$, x and y being spatial coordinates, has dynamics given by (2.1) with

$$A = I_2 \otimes \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \qquad Q = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \otimes \begin{bmatrix} \Delta^2/4 & \Delta^3/2 \\ \Delta^3/2 & \Delta^2 \end{bmatrix}, \tag{2.33}$$

where $\sigma_x^2 = \sigma_y^2 = 0.1$ convey the inaccuracy given by approximating the actual vehicle motion with constant speed, and the sampling time is set as $\Delta = 1\text{ms}$.

The available set $\mathcal{V}$ is composed of six heterogeneous smart sensors:

- sensor 1 models a drone equipped with a powerful GPU-CPU processing hardware and a high-resolution camera working at 60fps, with a sparse matrix $C_i \in \mathbb{R}^{4\times100}$ with density coefficient 0.3 (command `sprand` in Matlab);

Table 2.1: Parameters used in simulation for each sensor (sensor ID in parenthesis).

Parameter	b_i	c_i	f_i	r_i
Powerful drone (1)	5×10^{-3}	2500	1250	15
Lightweight drone (2, 3)	7.5×10^{-2}	110	55	30
Event camera (4, 5, 6)	3.4	2	1	10

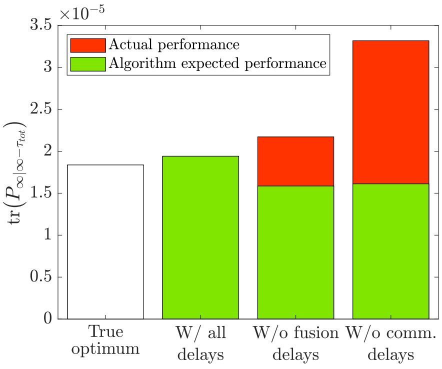


Figure 2.14: Optimal and greedy costs with different models. Numerical values (actual): $\{1.84, 1.94, 2.17, 3.32\} \times 10^{-5}$.

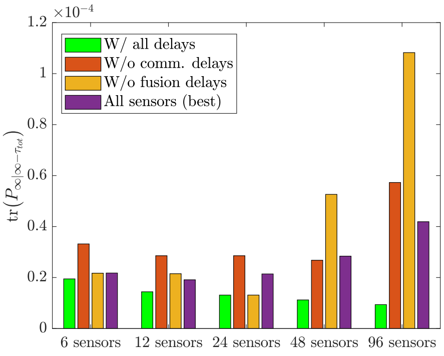


Figure 2.15: Greedy-selection and all-sensor costs with increasing set size. Fusion delays become crucial with large sets.

- sensors 2 and 3 model drones with low-resolution cameras working at 30fps, with sparse matrices $C_i \in \mathbb{R}^{4\times 20}$;
- sensors 4, 5 and 6 model event cameras collecting events at 100Hz, which output noisy estimates of the position.

Table 2.1 collects the parameters used in the simulation, assuming wireless connection working at 25 Mbps, which is needed to ensure real-time performance of the high-resolution camera. Also, packet-loss probability $1 - \lambda = 0.25$ is assumed for all sensors. The choice of parameters is based on the real-world experiments described in [123], and assume varying communication and fusion delays according to model (2.13b). For instance, drone 2 is assumed equipped with modest computational capabilities: with a 30ms-long processing delay, this sensor can estimate the vehicle position with an error standard deviation of 5cm. On the other hand, its low-quality camera provides images which take little time to be compressed/delivered via wireless. For instance, transmitting raw images (*i.e.*, with minimum processing $\tau_2 = 1$ms) takes 110ms. Also, modest computational resources are assumed at the fusion center, which takes half of the communication time to process data from each sensor. The processing delays range from 10ms to 290ms, only considering multiples of 20 to make the comparative brute-force search feasible.

Figure 2.14 shows the optimal cost (left) and the one achieved by the algorithm

Table 2.2: Sensors and delays: optimal and greedy selection.

Sensor		**1**	**2**	**3**	**4**	**5**	**6**
True optimum		/	30ms	30ms	/	/	/
Greedy	All delays	/	50ms	50ms	/	/	/
	W/o fusion	50ms	30ms	30ms	/	/	/
	W/o comm.	30ms	/	/	/	/	/

Table 2.3: Number of available sensors $|\mathcal{V}|$ *vs.* sensors selected by Algorithm 3 $|\mathcal{S}|$.

	Number of sensors				
Available	6	12	24	48	96
Greedy w/ all delays	2	2	3	5	5
Greedy w/o fusion delays	3	6	3	48	96

considering either the full model or only some delays. In particular, red bars show the actual cost (computed with all delays), while green bars represent the performance estimated by `computeCovariance`, which is an underestimate when the used model lacks either communication or fusion delays. The optimal cost is computed via brute-force search and is only used for benchmarking, since this strategy does not scale in the size of the network. The greedy selection yields a suboptimality gap of about 5.4% when all delays are considered, 18% when neglecting fusion delays, and 80% when neglecting communication delays. This translates into a larger tracking error: for instance, using only sensors 1 raises to 4m/s the optimal error on velocity, which is 3m/s. Selecting more sensors than necessary impacts also other aspects, *e.g.,* energy consumption.

Considering all delays, the proposed algorithm selects the optimal sensor subset and near-optimal processing delays (Table 2.2). According to intuition, the optimal choice features the same delay for both the selected sensor (sensors 2 and 3), being these (almost) identical. The powerful drone (sensor 1) is discarded because of its burdening impact on communication and fusion latency: this also explains the large variance when communication delays are neglected, as in this case sensor 1 is erroneously considered the best choice. The event cameras are excluded because of their large processing noise, not balanced by their fast acquisition rate and small communication and fusion delays.

If we consider a larger number of available sensors, accounting for the fusion delays becomes even more important. Figure 2.15 shows the performance obtained by greedy selection with increasing set size, together with the near-optimal cost when all sensors are selected (averaging 20 iterations of `sensorWiseDescent` with random initial delays). For each increment in set size, the added sensors have different delay parameters either for communication (coefficient c_i) or for computation (coefficients b_i and f_i), which range from 0.9 to 0.1 times of the original parameters in Table 2.1. One may think about these variations as different choices for the sensor hardware or better channel state/device position. We see the impact of better-performing sensors in the sets with 12 and 24 sensors, where the costs obtained when neglecting some delays are in the range of the ones in Fig. 2.14. However, from 6 to 96 sensors, the gaps between the proposed approach (green bars) and alternatives that do not account for communication and fusion delays steadily increases, which is particularly evident with 48 and 96 sensors. Also, given the availability of better-performing sensors when increasing the sensor set, the near-optimal costs decrease. Table 2.3 reports the number of sensors selected by Algorithm 3: it is evident that neglecting fusion delays leads to an unnecessarily large set of sensors in general, while the proposed approach enhances performance (as highlighted in the previous figures) while also using a smaller selection.

2.5 Conclusion

In this chapter, the problem of optimal estimation in a processing network in the presence of communication and computational delays has been investigated as motivation to explore sensing design under resource constraints. In particular, anytime local processing at sensor nodes has been modeled with a noise covariance decreasing with the amount of computation, which induces a nontrivial trade-off between accuracy and latency. For homogeneous networks monitoring a continuous-time scalar system, it is possible to prove that the processing delay can be analytically optimized, and that transmitting raw data is typically a suboptimal policy. For the general case of heterogeneous networks with discrete-time dynamics, the problem of jointly finding the optimal amount of processing at sensors and selecting the most informative sensors is tackled with greedy algorithms, which leverage insight provided by analytical results about the continuous-time scenario. Numerical simulations show the effectiveness of selection algorithms, corroborating the idea that the proposed model and latency-aware optimization problem leads to more accurate estimates. This is achieved by selection of an efficient subset of nodes providing high performance, whereas including all sensors is shown to perform poorly, as opposed to conventional wisdom that all sensors should be selected under no budget constraints.

This preliminary work opens several avenues of research. First, it is desirable to obtain suboptimality bounds on the proposed algorithms, or to design more effective selection strategies. Second, the system model can be made more realistic by introducing non-ideal communication (unreliability, random delays), nonlinear dynamics, or parameter uncertainty. Finally, it would be interesting to consider a fully distributed setup, where estimation is solved by local exchange of information, without a central fusion station.

3
Controller Architecture Design under Communication Latency

The previous chapter deals with a crucial source of latency affecting dynamics of a Networked Control System, namely computation latency caused by constrained hardware resources at network nodes. This is indeed a key factor to be accounted for if estimation or control tasks heavily rely on distributing computation across the network. In that case, the graph topology underlying information exchange was fixed, and I put most emphasis on designing local information processing for performance maximization within a resource-allocation framework, where resource constraints were implicitly defined by latency-accuracy and computation-communication trade-offs.

Another challenging issue, arising especially in large-scale wireless Networked Control Systems, is the latency due to constraints of communication channels, such as limited bandwidth, message retransmissions caused by interference, or packet loss due to channel erasure. While this aspect is partially addressed in Chapter 2, this chapter tackles it from a different standpoint, which is design of the network topology underlying inter-node communications. In fact, in a control framework, such communications actually define which feedback information is made available to each node from its neighbors, inducing an equivalence between communication network and architecture of a distributed controller. Crucially, this also means that transmitted information can be used by a local controller only after some communication delay, which draws a parallel with the estimation with delayed updates addressed in the previous chapter. Indeed, if the information transmitted from a neighbor directly enters a feedback loop, the controller can compute a control action only with outdated measurements, which can critically impact performance, among other aspects. In particular, if feedforward information such as a reference trajectory is not available, the lack of timely feedback measurements will prevent a controller to optimally track a desired signal, or more in general to follow a target behavior.

When communication constraints are given, control-related research has been moving towards three main assets.

One of these, at the intersection between control and telecommunications, studies stability and performance of controlled systems by specifically addressing features of communication channel such as delays and unreliability (see details in Section 1.1.2). This line of research, grounded in early works dating back to the first generation of network systems, puts emphasis on design of controller parameters and communication protocol, but typically does not address the network topology and just focuses on conditions of peer-to-peer communication links.

While this first research direction addresses wireless channel from a more precise and comprehensive viewpoint, other approaches within the control community just consider communication as a nuisance in a mostly qualitative fashion, focusing on aspects more strictly related to control and optimization. The second main asset of research is concerned with structured distributed architectures where communication constraints enter local feedback loops in form of input delays. To fix ideas, this means that the standard state update equation for a controlled system with state x_k and feedback control input $u_k = -Kx_k$,

$$x_{k+1} = (A - BK)x_k + w_k, \tag{3.1}$$

becomes the following delayed (or retarded) system with delay τ,

$$x_{k+1} = Ax_k - BKx_{k-\tau} + w_k, \tag{3.2}$$

which is more difficult to design because the presence of a time lag in (3.2) excites new modes as compared to (3.1). Indeed, it can be shown that a naive design of the feedback gain matrix K can make dynamics (3.2) unstable even if the nominal (*i.e.*, without considering delays) closed-loop matrix $A - BK$ is Schur, considerably restricting admissible design choices (see more details in Section 3.3.1).

In fact, a large body of work on delay systems focuses on stability: works [158], [183] are concerned with finite-time delay-dependent stability of discrete-time systems; paper [25] finds sufficient conditions for uniform stability of linear delay systems; article [135] characterizes stability and consensus conditions with homogeneous and heterogeneous feedback delays; references [38], [141] analyze consensus and error compensation for vehicular platoons. Another line of work deals with maximizing performance for structured controllers: for instance, the body of works [48], [65], [124] study $\mathcal{H}_2$-norm minimization for time-delay network systems; paper [177] proposes a cyber-physical architecture with LQR for wide-area power systems; study [131] develops a procedure for time-varying dead-time compensation by adapting the Filtered Smith Predictor.

Both these research approaches assume the network topology (controller architecture) be given, and purely focus on designing efficient control law or parameters in order to enforce either stability or effective performance. Conversely, a more recent trend is optimizing the controller architecture. Broadly speaking, the idea is that a careful design of the information flow across the network and among nodes can crucially impact global performance, given that it is required to keep low complexity of the controller (equivalently, small number of communication links). The latter requirement is useful in order not to overload shared communication channels, which happens for example with time-slotted protocols working with few carrier frequencies, and also allows a designer to drop maintenance costs and enable easier scalability. From a control-theoretic perspective, this typically turns into optimization problems where the to-be-optimized variable is the feedback gain matrix K (note that feedforward control inputs are local and do not need information sharing among nodes) and the constraint is given by an upper bound on the zero norm of K, which in words enforces a maximum number of communication links that can be deployed and used for distributed feedback. This is clearly a combinatorial problem that scales poorly with the size of the network, and hence is usually relaxed by considering convex relaxations or heuristics with low computational complexity. For example, this can be achieved by turning the constraint into a penalty term that is embedded into the cost function to trade performance for controller complexity, as done in works [6], [10], [51], [105], [109], [122].

However, one main drawback of this strategy is that latency is usually neglected in order to keep the optimization problem at an acceptable computational complexity, with penalty terms on the controller feedback gains that heuristically approximate exact impact of latency. In fact, while the fully connected architecture is avoided because of practical limitations, it is usually regarded as a theoretical upper bound for performance [78]. Hence, if delays affecting system dynamics are comparable with the time scale of state evolution, control performance might be seriously affected. In particular, it is still unclear how network connectivity affects the closed-loop performance in the presence of architecture-dependent communication latency. When the total available bandwidth does not increase with the size of the network [63], or when multi-hop communication is used for information exchange between low-power devices [204], the number of active communication links may affect such latency in non-negligible way. In this case, it is important that the control design takes into account increase in delays when new communication links are introduced. To the best of our knowledge, the only works where architecture-dependent delays are used to compute the performance metric are [204],

[205], where however the authors limit the scope of their study to deterministic evolutions of single integrators on regular lattices with delays due to asynchronous transmissions with a time-slotted protocol.

Given what is available in the literature today, the aim in this chapter is to bridge the two domains of delay-aware control and architecture design by quantifying how the latter affects performance under architecture-dependent communication delays. In order to do that, two key challenges are addressed. The first is about *optimal performance*, whereby *stability* is a prerequisite to control design needed to provide a bounded cost function. As discussed above, stability conditions may get much more challenging when delays enter the dynamics with respect to the nominal, delay-free case. Hence, stability conditions are derived that are instrumental to an optimal control design problem. The second point is identification of the *optimal controller architecture* under delays, which allows to quantify fundamental performance trade-offs. Towards this goal, to circumvent the discrete nature of graphs, I work along through two stages: first, each architecture is parametrized with a coefficient n which characterizes both number of links and delay associated with that architecture, and the optimal controller is computed for given n. Then, the optimal performance obtained for architectures with different values of n is compared, which allows to fairly establish which architectures provide the best closed-loop performance. In contrast to [204], [205], we examine mean-square performance of stochastically forced networks, study generic delay functions, and address optimal design of feedback gains for different controller architectures.

To make this study effective, I utilize undirected graphs with single- and double-integrator agent dynamics, and examine fundamental performance limitations in networked systems with architecture-dependent communication delays. In particular, symmetric feedback matrix gains induce convexity of a minimum-variance control design problem with respect to the feedback gains, which allows to solve optimization problems efficiently up to numerical precision. I exploit this fact to demonstrate that the choice of controller architecture has profound impact on network performance in the presence of delays: in particular, when the delays increase fast enough with the number of links, sparse topologies can outperform highly connected ones.

A crucial fact which used to corroborate the investigated performance trade-off on the number of communication links is the following: it can be show that the steady-state variance of a stochastically forced network, denoted by $J_{\text{tot}}(n)$, can be written as the sum of two monotone functions of the number of neighbors n (Fig. 3.1),

$$J_{\text{tot}}(n) = J_{\text{network}}(n) + J_{\text{latency}}(n). \tag{3.3}$$

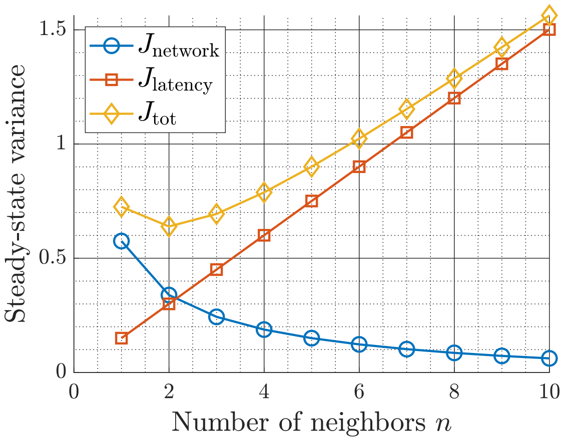


Figure 3.1: Steady-state variance $J_{\text{tot}}(n)$ versus number of neighbors. The variance is the sum of two costs: $J_{\text{network}}(n)$ represents impact of control architecture, while $J_{\text{latency}}(n)$ is due to the delays affecting the dynamics.

Here, $J_{\text{network}}(n)$ quantifies impact of control architecture and $J_{\text{latency}}(n)$ determines influence of communication latency on network performance. While $J_{\text{network}}(n)$ decreases with n and is minimized by a fully-connected centralized architecture, $J_{\text{latency}}(n)$ increases with n. This demonstrates the presence of a fundamental trade-off: on the one hand, feedback control takes advantage of dense topologies that enhance information sharing but, on the other hand, many communication links induce long delays which have negative effect on performance. While (3.3) can be derived analytically for ring topology with continuous-time, single-integrator dynamics, computational experiments show that a similar *centralized-distributed trade-off* can be observed for general undirected topologies and with double-integrator and discrete-time agent dynamics. Furthermore, in some cases, decentralized architecture with nearest neighbor information exchange provides optimal performance, in stark contrast with the conventional wisdom that increasing the information flow across the network is always beneficial for performance.

This chapter is organized as follows. Section 3.1 presents the models used for communication and controller architecture and the formulation of the minimum-variance control design problem. While ring topology are first used to achieve analytical insight, it can be shown that the proposed framework is also suitable for general undirected topologies (Section 3.4.1). In Section 3.2 and Section 3.3, I derive conditions for mean-square stability and compute the steady-state variance of continuous-time stochastically forced systems with continuous- and discrete-time systems, respectively, which are used to set up the optimal control design problem. In Section 3.4, I prove that the control design problem is convex, which is used to carry out a fair comparison among different controller architectures and to reinforce validity of numerical experiments.

In Section 3.5, I present the main result of this chapter: by numerically computing the optimal controller gains, I show that the closed-loop performance can be optimized by sparse architectures, disrupting the common assumption that more communication links ensure higher performance. Furthermore, I derive analytical expression (3.3) for continuous-time single-integrator dynamics, which demonstrates that the minimizer is in general nontrivial. Concluding remarks are provided in Section 3.6.

3.1 Problem Setup

Consider an undirected network with N agents in which the state of the ith agent at time t is given by $\bar{x}_i(t) \in \mathbb{R}$ with the control input $u_i(t) \in \mathbb{R}$. For notational convenience, in the following I use the aggregate state of the system $\bar{x}(t)$ and the aggregate control input $u(t)$, which respectively stack states and control inputs of each subsystem $\bar{x}_i(t)$ and $u_i(t)$.

Problem Statement. The agents aim to reach consensus towards a common state trajectory. The ith component of the vector $x(t) \doteq \Omega\bar{x}(t)$ represents the mismatch between the state of agent i and the average network state at time t [20], where

$$\Omega \doteq I_N - \frac{\mathbb{1}_N \mathbb{1}_N^\top}{N} \tag{3.4}$$

and $\mathbb{1}_N \in \mathbb{R}^N$ is the vector of all ones, such that $\Omega\mathbb{1}_N = 0$.

Ring Topology. To start with, I focusing on ring topology to obtain analytical insights about optimal control design and fundamental performance trade-offs in the presence of communication delays. This owes to the fact that eigenvalues of a circulant matrix can be obtained in closed form a functions of the matrix element, a fact that will be exploited later in the analysis performed in Section 3.5.1. While some notation is tailored to such topology (*e.g.,* see equations (3.5) and (3.7)), in Section 3.4.1 I discuss extension of the optimal control design to generic undirected networks and complement these developments with computational experiments in Section 3.5, which indeed proves the proposed approach pretty general as for scope of network topology.

Assumption 3.1.1 (Communication model)**.** Data are exchanged through a shared wireless channel in a symmetric fashion. Agent i receives state measurements from n pairs of agents, where both agents in each such pair are at equal distance ℓ from i. In what follows, without loss of generality, I assume that such n agent pairs coincide with the $2n$ closest agents in ring topology, and that each pair is at distance $\ell = 1, \dots, n < {}^N\!/_2$. For example, $n = 1$ corresponds to nearest-neighbor interaction and $n = \lfloor {}^{(N-1)}\!/_2 \rfloor$ to all-to-all communication. All measurements are received with delay $\tau_n \doteq f(n)$ where $f(\cdot)$ is a positive increasing sequence.

Remark 3.1.2 (Architecture parametrization)*.* Parameter n will play a crucial role throughout our discussion. In particular, it is used to (i) evaluate the optimal performance that can be attained for a given budget of communication links, and to (ii) compare optimal performance of different control architectures. In the first part of the paper, I consider circular formations and n represents how many neighbor pairs communicate with each agent. For general undirected networks, n determines the number of communication hops for each agent. In general, the parameter n characterizes sparsity of a controller architecture: sparse controllers correspond to small n whereas highly connected ones correspond to large n.

Feedback Control. Agent i uses the received information to compute the state mismatches $y_{i,\ell^\pm}(t)$ relative to its neighbors,

$$y_{i,\ell^\pm}(t) = \begin{cases} \bar{x}_i(t) - \bar{x}_{i\pm\ell}(t), & 0 < i \pm \ell \leq N \\ \bar{x}_i(t) - \bar{x}_{i\pm\ell\mp N}(t), & \text{otherwise}, \end{cases} \tag{3.5}$$

and the proportional control input is given by

$$u_{P,i}(t) = -\sum_{\ell=1}^{n} k_\ell \left(y_{i,\ell^+}(t-\tau_n) + y_{i,\ell^-}(t-\tau_n) \right), \tag{3.6}$$

where measurements are delayed according to Assumption 3.1.1. The proportional input can be compactly written as $u_P(t) = -K\bar{x}(t-\tau_n) = -Kx(t-\tau_n)$. With ring topology, the feedback gain matrix is determined by

$$K = \text{circ}\left(\sum_{\ell=1}^{n} k_\ell, -k_1, \dots, -k_n, 0, \dots, 0, -k_n, \dots, -k_1 \right), \tag{3.7}$$

where $\text{circ}\,(a_1, \dots, a_n)$ denotes the circulant matrix in $\mathbb{R}^{n \times n}$ with elements $a_1, \dots, a_n$ in the first row.

For networks with double integrator agents, the control input $u_i(t)$ may also include a derivative term,

$$u_i(t) = \eta u_{P,i}(t) - \eta \frac{d\bar{x}_i(t)}{dt} = \eta u_{P,i}(t) - \eta \frac{dx_i(t)}{dt}. \tag{3.8}$$

The derivative term in (3.8) is delay free because it only requires measurements coming from the agent itself, which I assume to be available instantaneously. On the contrary, communication delays are explicitly written in (3.6) because they are comparable with the time evolution of the state $\bar{x}(t)$.

Problem 3.1.3. Design the feedback gains in order to minimize the steady-state variance of the consensus error,

$$\text{P control:} \qquad \arg\min_{K} \ \sigma^2(K), \tag{3.9a}$$

$$\text{PD control:} \qquad \arg\min_{\eta,K} \ \sigma^2(\eta, K), \tag{3.9b}$$

where

$$\sigma^2 \doteq \lim_{t\to+\infty} \mathbb{E}\left[\|x(t)\|^2\right] \tag{3.10}$$

and without loss of generality I assume $\mathbb{E}\left[x(\cdot)\right] \equiv \mathbb{E}\left[x(0)\right] = 0$.

3.2 Continuous-Time Agent Dynamics

In this section, I examine continuous-time networks with single- (Section 3.2.1) and double-integrator (Section 3.2.2) agent dynamics, derive conditions for mean-square stability, and compute the steady-state variance of a stochastically forced system. These developments are instrumental for the formulation of the control design problem which is used to compare different control architectures. In the optimal control problem, the steady-state variance determines the objective function and stability conditions represent feasibility constraints. Note that all results in this section hold for generic undirected topologies.

3.2.1 Single Integrator Model

The dynamics of the ith agent are described by the first-order differential equation driven by standard Brownian noise $\bar{w}_i(\cdot)$,

$$d\bar{x}_i(t) = u_{P,i}(t)dt + d\bar{w}_i(t). \tag{3.11}$$

The network error dynamics are

$$dx(t) = -Kx(t-\tau_n)dt + dw(t), \tag{3.12}$$

where the process noise is given by $dw(t) \sim \mathcal{N}\left(0, \Omega\Omega^\top dt\right)$. Exploiting symmetry of the matrix K, the change of variables $x(t) = T\tilde{x}(t)$, with $K = T\Lambda T^\top$, is employed to obtain

N decoupled scalar subsystems with state $\tilde{x}_j(t)$, $j = 1, \dots, N$,

$$d\tilde{x}_j(t) = -\lambda_j \tilde{x}_j(t - \tau_n)dt + d\tilde{w}_j(t), \tag{3.13}$$

where λ_j is the jth eigenvalue of K. The subsystem corresponding to $\lambda_1 = 0$ has trivial dynamics, *i.e.*, $d\tilde{x}_1(t) \equiv 0$, where the initial condition is $\tilde{x}_1(0) = 0$ by construction. For $j \neq 1$, subsystem (3.13) is a single integrator driven by standard Brownian noise.

Stability Analysis. Mean-square stability of scalar stochastic differential equations of the form (3.13) has been addressed in the literature. The classical result in [92] allows to characterize consensus stability for the multi-agent formation.

Proposition 3.2.1 (Stability of CT single integrators)**.** *The network error $x(t)$ is mean-square stable if and only if*

$$\lambda_j \in \left(0, \frac{\pi}{2\tau_n}\right), \quad j = 2, \dots, N. \tag{3.14}$$

In this case, $x(t)$ is a Gaussian process and its steady-state variance is determined by

$$\sigma^2(K) = \sum_{j=2}^{N} \sigma_I^2(\lambda_j), \quad \sigma_I^2(\lambda_j) = \frac{1 + \sin(\lambda_j \tau_n)}{2\lambda_j \cos(\lambda_j \tau_n)}, \tag{3.15}$$

where $\sigma_I^2(\lambda_j)$ is the variance of the trivial solution of (3.13).

Sketch of Proof. In view of the decoupling, stability of (3.12) amounts to stability of all subsystems (3.13), $j = 1, \dots, N$, with the variances of $x(t)$ and $\tilde{x}(t)$ being equal. Condition (3.14) and expression (3.15) were derived in [92]. □

While the variance of delay-free systems is bounded for any positive eigenvalues $\lambda_2, \dots, \lambda_N$, the presence of delay constrains a stabilizing control according to (3.14). In fact, longer delays τ_n induce smaller upper bounds on the eigenvalues.

The following corollary will turn useful in the control design in Section 3.4.

Corollary 3.2.2. *Let λ satisfy (3.14). Then the function $\sigma_I^2(\lambda)$ is strictly convex and the minimizer λ^* is determined by*

$$\lambda^* = \frac{\beta^*}{\tau_n}, \qquad \beta^* = \cos \beta^*. \tag{3.16}$$

Proof. See Appendix B.1. □

3.2.2 Double Integrator Model

I now examine networks in which each agent obeys a second-order dynamics with the PD control input (3.8),

$$\frac{d^2\bar{x}_i(t)}{dt^2} = u_i(t) + \frac{d\bar{w}_i(t)}{dt}. \tag{3.17}$$

For simplicity, we normalize the delay by rescaling (3.17),

$$\bar{x}_i(\cdot) \leftarrow \bar{x}_i(\tau_n\,\cdot), \quad \eta \leftarrow \tau_n\eta, \quad k_\ell \leftarrow \tau_n k_\ell, \quad \bar{w}_i(\cdot) \leftarrow \tau_n\bar{w}_i(\cdot). \tag{3.18}$$

Stacking the agent errors and their derivatives in the formation vector, the error dynamics can be decoupled as before, yielding

$$\frac{d^2\tilde{x}_j(t)}{dt^2} = -\eta\frac{d\tilde{x}_j(t)}{dt} - \eta\lambda_j\tilde{x}_j(t-1) + \frac{d\tilde{w}_j(t)}{dt}. \tag{3.19}$$

Stability Analysis. The following result characterizes which controller parameters yield mean-square stability.

Proposition 3.2.3 (Stability of CT double integrators)**.** *The network error $x(t)$ is mean-square stable if*

$$\lambda_j \in \left(0, \frac{\beta}{\sin\beta}\right), \ \eta = \beta\tan\beta, \ \beta \in \left(0, \frac{\pi}{2}\right), \ j = 2, \dots, N. \tag{3.20}$$

Condition (3.20) can be equivalently written as

$$(\eta, \lambda_j) \in \mathcal{S} \doteq \left\{(\eta, \lambda_j) \in \mathbb{R}^2_+ : \lambda_j < \phi(\eta)\right\}, \ j = 2, \dots, N, \tag{3.21}$$

where the implicit function $\phi(\cdot)$ is concave increasing and

$$\phi(0) = 1, \quad \lim_{\eta\to+\infty} \phi(\eta) = \frac{\pi}{2}. \tag{3.22}$$

If $\exists j \neq 1 : (\eta, \lambda_j) \notin \overline{\mathcal{S}}$, the system is mean-square unstable.

Proof. See Appendix B.2. □

Similar to the single-integrator case, 3.2.3 states that the presence of delay requires more restrictive conditions than positive gains. In words, the system is stable if the instantaneous component of the control input in (3.8) is sufficiently “strong” compared to the delayed one. The steady-state variance of $\tilde{x}_j(t)$ for $j \neq 1$ can be computed using [208,

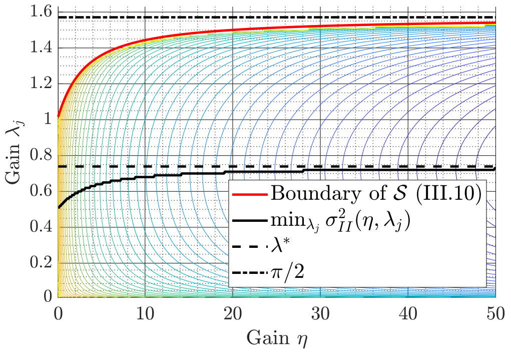


Figure 3.2: Level curves of the steady-state variance for the continuous-time double integrator (3.19) and points of minimum with fixed derivative gain.

Section 4],

$$\sigma^2_{II}(\eta, \lambda_j) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \frac{d\omega}{|-\omega^2 + j\eta\omega + \eta\lambda_j e^{-jw}|^2}, \tag{3.23}$$

and $\sigma^2 = \sigma^2(\eta, K) = \sum_{j=2}^{N} \sigma^2_{II}(\eta, \lambda_j)$. A graphical illustration of the level curves of $\sigma^2_{II}(\eta, \lambda_j)$ is provided in Fig. 3.2.

Model Approximation. When the feedback gain η is sufficiently high, separation of time scales allows us to approximate (3.19) with the first-order dynamics,

$$d\tilde{x}_j(t) = -\lambda_j \tilde{x}_j(t-1)dt + dn(t), \tag{3.24}$$

where the variance of Brownian motion $n(t)$ is inversely proportional to η. In words, when the damping is high enough, the derivative of $\tilde{x}_j(t)$ converges to zero much faster than $\tilde{x}_j(t)$, which represents the dominant component of the dynamics. The detailed derivation of (3.24) is provided in B.3. Utility of this approximation is illustrated in Fig. 3.2: with fixed $\bar{\eta}$, the point of minimum of the corresponding 1D variance curve, *i.e.*, $\arg\min_{\lambda_j} \sigma^2_{II}(\bar{\eta}, \lambda_j)$ (solid black line), approaches the minimizer λ^* of the single integrator model (dashed black, see 3.2.2) with increase of $\bar{\eta}$. It can also be noted that the variance decreases with η.

3.3 Discrete-Time Agent Dynamics

I now turn to discrete-time agent dynamics, which is more realistic in the context of wireless communication. With discrete-time dynamics, time instants are denote by $\{k\}_{k\in\mathbb{N}} \doteq \{kT\}_{k\in\mathbb{N}}$, T being the sampling time. Similarly, delays are re-defined as the number of delay steps $\tau_n \doteq \lceil {}^{\tau_n}/{}_{T_s} \rceil$.

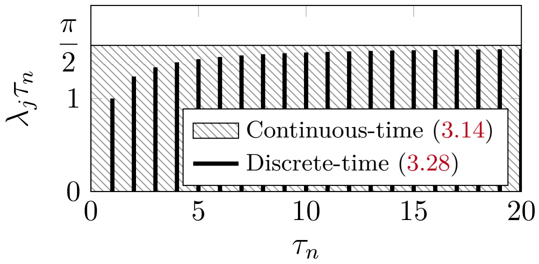


Figure 3.3: Stability regions of decoupled single integrators in continuous (dashed rectangle) and discrete time (solid vertical lines).

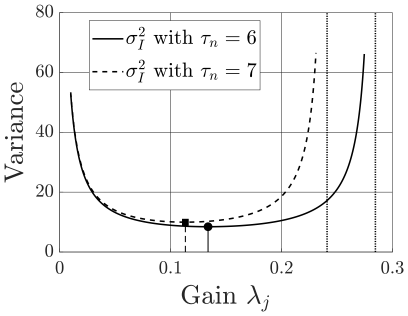


Figure 3.4: Steady-state variance for decoupled discrete-time single integrators.

3.3.1 Single Integrator Model

The discrete-time versions of the agent dynamics considered in Section 3.2 are given by

$$\bar{x}_i(k+1) = \bar{x}_i(k) + u_{P,i}(k) + \bar{w}_i(k), \tag{3.25}$$

with $u_{P,i}(k)$ defined in (3.6). The global error dynamics can be written as

$$x(k+1) = x(k) - Kx(k-\tau_n) + w(k), \tag{3.26}$$

and the corresponding error dynamics can be mapped into decoupled subsystems similarly to what showed in Section 3.2.1, by exploiting symmetry of K and using the change of basis induced by its eigenvectors. The resulting subsystems obey the following equation,

$$\tilde{x}(k+1) = \tilde{x}(k) - \lambda\tilde{x}(k-\tau_n) + \tilde{w}(k). \tag{3.27}$$

Stability Analysis. The decoupled subsystems (3.27) are asymptotically stable if and only if all the roots of their associated characteristic polynomials lie inside the unit circle in the complex plane.

In general, given a delay τ_n, stability conditions with respect to the control gains can be derived in the form of polynomial inequalities through the Jury criterion. However, in this case, stability can be explicitly expressed by the following simple condition.

Proposition 3.3.1 (Stability of DC single integrators)**.** *The network error $x(t)$ is mean-square stable if and only if*

$$\lambda_j \in \left(0, 2\sin\left(\frac{\pi}{2}\frac{1}{2\tau_n+1}\right)\right), \quad j = 2, \dots, N. \tag{3.28}$$

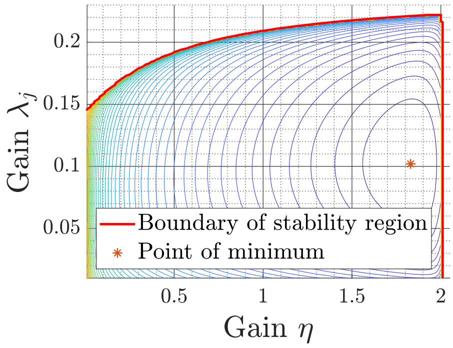


Figure 3.5: Steady-state variance of decoupled discrete-time double integrators.

Proof. See Appendix B.4. □

The upper bound in (3.28) approaches its continuous-time counterpart (3.14) from below as the delay steps tend to infinity (see Fig. 3.3). Indeed, given the same absolute delay, a finer sampling yields more delay steps. and thus at the limit the discretized dynamics converges to the continuous-time one. and retrieves the same constraint. In general, condition (3.28) is tighter than (3.14). On the other hand, the asymptotic behavior of the threshold gain suggests that the gap between continuous-time and discretized systems only matters when the delay is comparable with the sampling time, while, when the former gets too long, the loss of feedback information neglects the dynamics discretization.

Performance Evaluation. With fixed parameters, the steady-state variance of each decoupled subsystem can be computed numerically via the Wiener–Khintchine formula. Also, for any given value of τ_n, a closed-form expression of the variance can be obtained via moment matching through a recursive formula, reported in B.5 to make this exposition compact. Figure 3.5 shows the typical profiles of the variance function for decoupled subsystems with single-integrator dynamics. In this case, convexity of the variance function $\sigma^2(\lambda_i)$ can be checked by studying the second derivative of the decoupled subsystems, which can be done by suitable software tools applied to the analytical expression in B.5.

3.3.2 Double Integrator Model

The discrete-time versions of the agent dynamics considered in Section 3.2 are given by

$$\begin{aligned}\bar{x}_i(k+1) &= \bar{x}_i(k) + \bar{z}_i(k)\\ \bar{z}_i(k+1) &= (1-\eta)\bar{z}_i(k) + \eta u_{P,i}(k) + \bar{w}_i(k).\end{aligned} \tag{3.29}$$

In this case, the error dynamics can be decoupled into the following scalar subsystems,

$$\begin{aligned}\tilde{x}(k+1) &= \tilde{x}(k) + \tilde{z}(k)\\ \tilde{z}(k+1) &= (1-\eta)\tilde{z}(k) - \eta\lambda\tilde{x}(k-\tau_n) + \tilde{w}(k).\end{aligned} \tag{3.30}$$

Stability Analysis and Performance Evaluation. Under dynamics (3.30), analytical stability conditions analogous to the ones in Proposition 3.3.1 are hard to obtain. Hence, I resort to standard application of the Jury stability criterion to obtain feasibility constraints for the optimal control design problem. Given their polynomial structure, those unfortunately prevent the optimization problem to be convex in this case, however, numerical results shown in Section 3.5 are consistent with both the centralized-distributed trade-off hypothesis and with other simulations with convex formulations.

The steady-state variance can be instead explicitly computed through a recursive formula as for single integrators. Details are provided in Appendix B.5.

3.4 Control Design

Single Integrator Model. For systems (3.12)–(3.25), Problem 3.1.3 amounts to

$$k_1^*, \dots, k_n^* = \underset{\{k_\ell\}_{\ell=1}^n}{\arg\min}\ \sigma^2(K), \tag{3.31}$$

and parameterization (3.13) allows to rewrite it as

$$k_1^*, \dots, k_n^* = \underset{\{k_\ell\}_{\ell=1}^n}{\arg\min} \sum_{j=2}^{N} \sigma_I^2\left(\lambda_j\right), \tag{3.32}$$

with stability condition given by (3.14). Linear dependence of the eigenvalues of K on the feedback gains [66] and 3.2.2 guarantee convexity of optimization problem (3.32). Thus, the optimal feedback gains can be computed efficiently.

To make analytical progress and gain intuition, I also consider the following approximation of (3.32),

$$\tilde{k}_1^*, \dots, \tilde{k}_n^* = \underset{\{k_\ell\}_{\ell=1}^n}{\arg\min} \sum_{j=2}^{N} \left(\lambda_j - \lambda^*\right)^2, \tag{3.33}$$

which squeezes the spectrum of K about the "optimal" eigenvalue λ^*. The variance $\sigma_I^2\left(\cdot\right)$ can be approximated with a quadratic function around its minimum because it is strictly convex, differentiable in the stability region, and it blows up at the boundaries $\{0, {}^{\pi}\!/_{2}\}$,

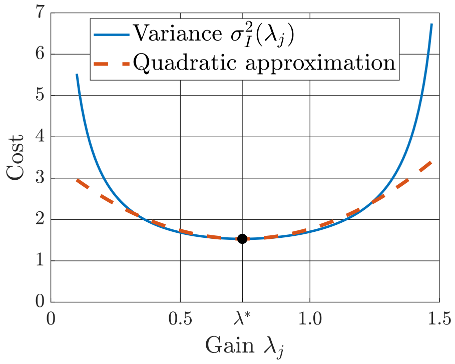


Figure 3.6: Exact variance function (3.15) and its quadratic approximation.

see Fig. 3.6.

Proposition 3.4.1 (Near-optimal proportional control)**.** *The solution of problem* (3.33) *is determined by*

$$\tilde{k}_\ell^* \equiv \tilde{k}^* \doteq \frac{\lambda^*}{2n+1}.$$

Proof. See Appendix B.6. □

Proposition 3.4.1 shows that spatially-constant feedback gains provide good performance even when spatially-varying feedback gains are allowed. According to Corollary 3.2.2, the suboptimal gain $\tilde{k}^*$ decreases with the delay τ_n and with the number of agents involved in the feedback loops, thereby reflecting benefits of communication.

Double Integrator Model. For continuous-time models, approximation (3.24) and Fig. 3.2 show that, for sufficiently large η, the variance of the double-integrator subsystem (3.19) has structure similar to the single integrator, *i.e.*, $\sigma_{II}^2(\eta, \lambda_j) \approx c\sigma_I^2(\lambda_j)$ for some "small" $c > 0$. This suggests that the control design (3.9b) can be approximated as follows,

$$\tilde{\eta}^*, \underset{\{k_\ell\}_{\ell=1}^n}{\arg\min} \sum_{j=2}^{N} \sigma_I^2(\lambda_j), \tag{3.34}$$

where $\tilde{\eta}^*$ is chosen beforehand so that the time-scale separation argument provides a reasonable approximation (3.24). In particular, the optimization problem for proportional feedback gains in (3.34) coincides with the control design for single integrators (3.32), with the exception that the stability condition is now given by $\lambda_j < \phi(\tilde{\eta}^*)$, $j = 2, \dots, N$ (see (3.21)).

Remark 3.4.2 (Convexity enables comparison)*.* Convexity of the optimal control design problems (3.32)–(3.34) enables both efficient numerical computations of the optimal feedback gains for *given* n and fair comparison of the best achievable performance for *different* values of n.

Remark 3.4.3 (Gain scaling). The optimal feedback gains $\{k_\ell^*\}_{\ell=1}^n$ and $\tilde{\eta}^*$ are to be scaled by ${}^1/_{\tau_n}$ according to (3.18).

Remark 3.4.4 (Optimal design for double integrators). Local minimizer of the original problem approximated by (3.34) can be solved using the gradient-based method proposed in [65]. However, this approach has no guarantees of global optimality, and its computational complexity is impractical for large-scale systems. In contrast, convex approximation (3.34) draws a parallel to the optimal design for the single-integrator model and provides insight into a centralized-distributed trade-off.

For discrete-time models, both cost function and feasibility region can be computed analytically as discussed in Equation 3.3.1. Hence, the optimization problem can be numerically solved. Given that the lack convexity prevents to use theoretical results for optimality guarantees, this last case remains an heuristic validation of the trade-off in hypothesis, which is however strongly supported by numerical experiments showed in Section 3.5.

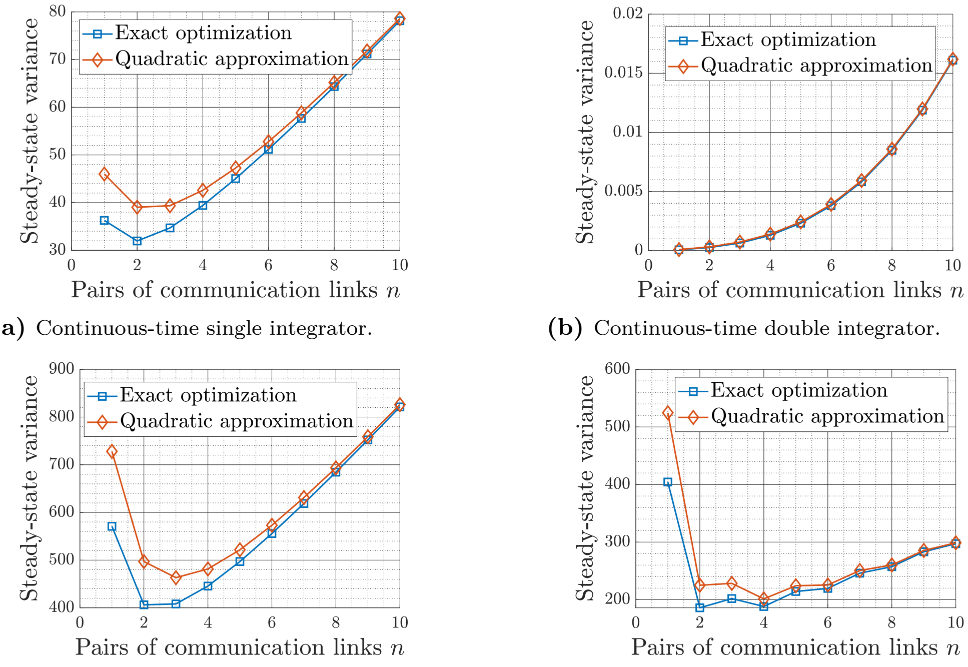
3.4.1 General Symmetric Network Topology

Even though we utilized ring topology to derive analytical results (see Section 3.5.1), the control design can be extended to general undirected networks with symmetric feedback gain matrices K. For the single integrator model, this reads

$$K^* = \arg\min_{K} \ \sigma^2(K). \tag{3.35}$$

The steady-state network error variance $\sigma^2(K)$ is a convex function if and only if $\sigma_I^2(\lambda_j)$ is convex [46], which holds at least for single integrators and for the first-order approximation of continuous-time double integrators. The optimal gains can then be found numerically via gradient-based methods, where gradients of the eigenvalues can be computed using analytical [138], [149] or numerical [60] methods. On the other hand, the derivative feedback gain in $\sigma_{II}^2(\eta, \lambda_j)$ prevents us from establishing convexity for second-order systems in general. However, if $\sigma_{II}^2(\eta, \lambda_j)$ is convex in each coordinate[1], the design problem can be solved by alternatively optimizing proportional and derivative gains and the centralized-distributed trade-off can be studied irrespective of the particular topology.

[1] This can be checked for discrete-time double integrators through their analytical expression.



(a) Continuous-time single integrator.

(b) Continuous-time double integrator.

(c) Discrete-time single integrator.

(d) Discrete-time double integrator.

Figure 3.7: Optimal and suboptimal steady-state scalar variances with linear delay increase for different agent dynamics.

3.5 The Centralized-Distributed Trade-Off

In the previous sections I formulated the optimal control problem for a given controller architecture (*i.e.,* the number of links) parametrized by n and showed how to compute minimum-variance objective function and the corresponding constraints. In this section, I present the main result: the optimal control problem is solved for each n, and the best achievable closed-loop performance are compared with different control architectures.[2] In particular, for delays that increase linearly with n, *i.e.,* $f(n) \propto n$, it is possible to see that distributed controllers with few communication links outperform controllers with larger number of communication links.

Figure 3.7a shows the steady-state variances obtained with single-integrator dynamics (3.31) and the quadratic approximation (3.33) for ring topology with $N = 50$ nodes. The best performance is achieved for a sparse architecture with $n = 2$ in which each agent communicates with the two closest pairs of neighboring nodes. This should be compared and contrasted to nearest-neighbor and all-to-all communication topologies which induce higher closed-loop variances. Thus, the advantage of introducing additional

[2] Recall that small (large) values of n mean sparse (dense) architectures.

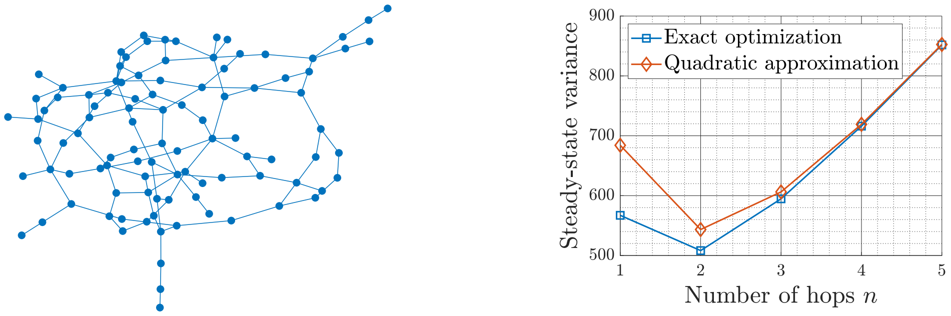


Figure 3.8: Network topology and its optimal closed-loop variance.

communication links diminishes beyond a certain threshold because of communication delays.

Figure 3.7b shows that the use of approximation (3.34) with $\tilde{\eta}^* = 70$ identifies nearest-neighbor information exchange as the near-optimal architecture for a double-integrator model with ring topology. This can be explained by noting that the variance of the process noise $n(t)$ in the reduced model (3.24) is proportional to ${}^1\!/\!_\eta$ and thereby to τ_n, according to (3.18), making the variance scale with the delay.

Figures 3.7c–3.7d show the results obtained by solving the optimal control problem for discrete-time dynamics. The oscillations about the minimum in Fig. 3.7d are compatible with the investigated centralized-distributed trade-off (3.3): in general, the sum of two monotone functions does not have a unique local minimum. Interestingly, double integrators with continuous- (Fig. 3.7b) ad discrete-time (Fig. 3.7d) dynamics exhibits very different trade-off curves, whereby performance monotonically deteriorates for the former and oscillates for the latter. While a clear interpretation is difficult because there is no explicit expression of the variance as a function of n, one possible explanation might be found in the first-order approximation used to compute gains in the continuous-time case.

Finally, Fig. 3.8 shows the optimization results for a random graph topology with discrete-time single integrator agents. Here, n denotes the number of communication hops in the "original" network, shown in Fig. 3.8: as n increases, each agent can first communicate with its nearest neighbors, then with its neighbors' neighbors, and so on. For a control architecture that utilizes different feedback gains for each communication link (*i.e.*, we only require $K = K^\top$) it can be seen that, in this case, two communication hops provide optimal closed-loop performance.

Additional computational experiments performed with different rates $f(\cdot)$ show that the optimal number of links increases for slower rates: for example, the optimal number of links is larger for $f(n) = \sqrt{n}$ than for $f(n) = n$.

3.5.1 Ring Topology: Analytical Insight Into the Trade-Off

For a ring topology with continuous-time single-integrator agent dynamics, a centralized-distributed trade-off can be explicitly quantified. By utilizing Proposition 3.4.1 to compute the feedback gains, the objective function can be factorized as

$$\sigma^2 = \underbrace{f(n)}_{\tilde{J}_{\text{latency}}(n)} \cdot \underbrace{\sum_{j=2}^{N} \tilde{C}_j^*(n)}_{\tilde{J}_{\text{network}}(n)}, \tag{3.36}$$

where $\sigma_I^2(\tilde{\lambda}_j^*) = \tilde{C}_j^*(n)\tau_n$ and $\tilde{C}_j^*(n)$ only depends on n and can be computed exactly. Indeed, when the feedback gains are given by (3.33), the eigenvalues of K have expression (cf. [66])

$$\tilde{\lambda}_j^* = 2\tilde{k}^* \left(n - \sum_{\ell=1}^{n} \cos\left(\frac{2\pi(j-1)\ell}{N} \right) \right), \tag{3.37}$$

which can be compactly written as $\tilde{\lambda}_j^* = g_j(n)\tilde{k}^*$. Writing the feedback gains as $\tilde{k}^* = \tilde{\alpha}^*(n)\lambda^*$, it holds $\tilde{\lambda}_j^* = \tilde{c}_j^*(n)\lambda^*$ with $\tilde{c}_j^*(n) \doteq g_j(n)\tilde{\alpha}^*(n)$. Interestingly, the decomposition $\tilde{\lambda}_j^* = \tilde{c}_j^*(n)\lambda^*$ can be interpreted as a decoupling of the impact of network (expressed by $\tilde{c}_j^*(n)$) and latency (given by λ^*) effects on the control design. Then, each subsystem (3.13) has variance

$$\sigma_I^2\left(\tilde{\lambda}_j^*\right) = \frac{1+\sin(\tilde{\lambda}_j^*\tau_n)}{2\tilde{\lambda}_j^*\cos(\tilde{\lambda}_j^*\tau_n)} \overset{(i)}{=} \frac{1+\sin(\tilde{c}_j^*(n)\beta^*)}{2\tilde{c}_j^*(n)\beta^*\cos(\tilde{c}_j^*(n)\beta^*)}\tau_n = \tilde{C}_j^*(n)\tau_n, \tag{3.38}$$

where (3.16) is used in *(i)*.

By inspection, it can be seen that $\tilde{J}_{\text{network}}(n)$ is a decreasing function of n and that $\tilde{J}_{\text{latency}}(n)$ is determined by $f(n)$. Furthermore, when $f(\cdot)$ is sublinear, expression (3.36) can be equivalently written in form (3.3),

$$\sigma^2 = \underbrace{f(n) \cdot \sum_{j=2}^{N} \left(\tilde{C}_j^*(n) - C^* \right)}_{J_{\text{network}}(n)} + \underbrace{(N-1)C^* f(n)}_{J_{\text{latency}}(n)}, \tag{3.39}$$

where $\sigma_I^2(\lambda^*) = C^*\tau_n$ is the optimal variance according to (3.15) and Corollary 3.2.2. Indeed, the summation decreases with superlinear rate, so that $J_{\text{network}}(n)$ is a decreasing sequence. The terms in $J_{\text{network}}(n)$, each associated with a decoupled subsystem (3.13), illustrate benefits of communication: as n increases, the eigenvalues of K have more degrees of freedom and can squeeze more tightly about λ^*, reducing performance gaps between subsystems and theoretical optimum. Also, it can be noted that $J_{\text{network}}(n)$ vanishes when the controller architecture turns fully connected.

Even though analogous expressions could not be obtained for other dynamics, the curves in Fig. 3.7 exhibit trade-offs which are consistent with the above analysis.

3.6 Conclusion

In this chapter, I have studied a minimum-variance optimal control design problem on undirected networks with both continuous- and discrete-time agent dynamics in the presence of communication delays. When feedback delays increase with the number of communication links, convexity of the optimization problem proves the existence of a fundamental performance trade-off: distributed control architectures can offer superior performance to centralized ones that utilize all-to-all information exchange. This is due to two contrasting components of the cost function (steady-state variance) that behave differently as communication links are deployed. On the one hand, the information used to feed local control input loops becomes richer with more links, which pushes performance to initially improve with the number of links. On the other hand, delays in feedback dynamics induced by communication also increase with the number of links, inducing performance degradation if those are added beyond a certain threshold. This latter behavior in is sharp contrast with conventional results in literature and also with common design choices, which build on the assumption that the more communication is better. Overall, such twofold nature of the system dynamics yields a nontrivial performance trade-off on the number of links, which is in general optimized by a distributed controller architecture.

Given the preliminary results in this chapter, the hope is to pave the way to a new body of research which will enable control design with a deeper understanding of the fundamental behavior and limitations of large-scale wireless network systems. Future work will focus on extending the results found here to other classes of control problems which include more complex system dynamics and communication models, more realistic information about structure of delays in a distributed scenario, as well as different cost functions.

4
Resilient Consensus

The design trade-offs faced in the previous chapters have most to do with feasibility aspects, that arise from limitations of individual devices (such as resource-constrained sensors) or of peer-to-peer communications (such as feedback delays) that involve the networked nature of a Networked Control System as a secondary element.

In this chapter, I steer to a different kind of problems arising in Networked Control Systems, that are due precisely to their interconnected structure. Indeed, while the latter brings numerous benefits, interdependence of networked subsystems at dynamics level is a concern if nodes cannot be guaranteed to always evolve as expected or designed. In particular, a major drawback is represented by misbehaving nodes, whose uncontrolled behavior can seriously affect their neighbors and possibly the rest of the system. For example, malicious agents can locally intrude from any point in the system and cause damage at global scale. Such aspect of cybersecurity in Networked Control Systems has come under the spotlight for a few years now. For example, Department of Energy secretary has recently stated that enemies of the United States can shut down the U.S. power grid, and it is known that hacking groups around the world have high technological sophistication [156]. Another emblematic case is represented by cyberattacks that hit the Italian health care infrastructures during the COVID-19, disrupting services for weeks [30]. Other concerns are due to accidental failures spreading from single source nodes. Cascading failure damages have notable examples, from city-wide electricity blackouts to denial of service of web applications. Furthermore, as new frontiers through massively connected devices in Networked Control Systems are breached, thanks to powerful communication protocols such as 5G, this problem will only gain in importance.

The above problems have been extensively addressed in the literature, with a large body of work investigating techniques to overcome fragility in application domains, such as smart grids [72], [155], [217]; cascading failures [68], [152], [154]; denial of service [35], [214], multi-robot systems [2], [150], [216]; attack detection within Cloud Computing and Internet of Things [35], [214]; distributed estimation [169]. From a methodological perspective, related literature in control and optimization mostly focuses on robustness of distributed algorithms and control protocols to a fraction of misbehaving agents. Specifically, the framework of *resilient consensus* aims to enforce consensus of normally behaving agents in the face of unknown adversaries. The classical consensus problem is a cornerstone in control and optimization. Indeed, it has been deeply studied in the past two decades [212] and underlies a plethora of application domains. In particular, *average consensus* is a fundamental tool in distributed estimation [169] and optimization [172], [213], management of power grids [87], distributed Federated Learning [29], [164], to mention a few. Unfortunately, standard consensus dynamics is extremely sensitive to the behavior of individual nodes, so that misbehaving agents can arbitrarily deviate the system trajectory, for example forcing all other agents to reach an inefficient or dangerous consensus. To tame this issue, the most common approaches rely on the filtering strategy "Weighted Mean Subsequence Reduced" (W-MSR), whereby agents opportunistically discard suspicious messages from local updates [94]. A review of resilient consensus, and more generic resilient distributed optimization and control, techniques investigated in the literature is provided in Section 1.1.3.

Despite the success of MSR-based strategies, a critical point is dependence of theoretical guarantees on r-robustness of the underlying graph, a sufficient property ensuring that agents reach resilient consensus provided that r is large enough. In words, this parameter r characterizes the ability of the network topology to spread information: a large r means that each node is well connected with all other nodes in the network, so that it can both transmit information to, and receive information from, nodes far away with just a few hops. A critical limitation of W-MSR and related protocols is that characterizing the steady-state behavior of agents is difficult if some minimal r-robustness is not verified. Indeed, even though algorithms might work in practice, comprehensive theoretical guarantees are missing. What is more, some applications may require more conservative approaches, pushing less on performance but with solid safety guarantees, as commonly required by industrial safety protocols. Also, while in some cases agents may just agree on a common value, other tasks critically require *average consensus* to succeed.

In view of the discussion above, I depart from classical filtering strategies and seek a technique that can offer theoretical guarantees in a broader sense – while ensuring some resilience. Towards this goal, I set the stage with two key moves. First, I replace the hard constraint of achieving consensus with the cost of a distributed optimization problem. This allows to relax requirements on the system, in that enforcing consensus is in general

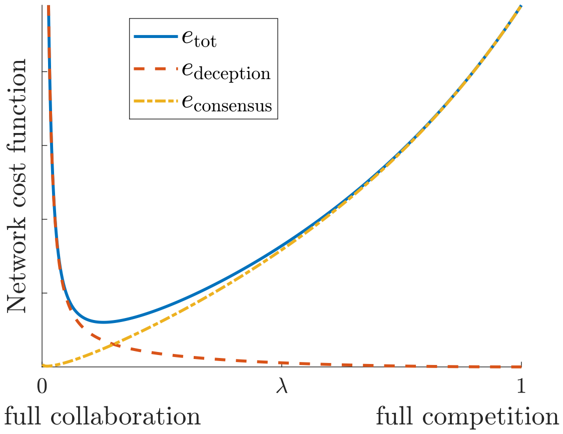


Figure 4.1: Competition *vs.* collaboration in distributed quadratic optimization. The global cost e_{tot} is the sum of two contributions, that reflect two contrasting attitudes of normally behaving agents: $e_{\mathrm{deception}}$ is caused by trusting (erroneously) adversarial agents, which yields a drift from nominal average consensus, while $e_{\mathrm{consensus}}$ is due to inter-agent competition, in an attempt to mitigate potential attacks from neighbors, but doing so prevents agents to reach consensus. The tunable parameter $\lambda \in [0,1]$ allows normal agents to shift smoothly from full collaboration, where they trust equally all agents in the network, to full competition, where they trust only themselves, inducing a richer range of behaviors at local and global scale.

much more restrictive than finding "small" values of a cost function. Second, I draw inspiration from game theory to design an alternative update protocol at nodes. Even though distributed control protocols and games may seem contrasting approaches, they share manifold features which have been explored in the literature [23], [45], [79], [103], [118], [132], [151]. In particular, the starting point in this chapter is paper [117], where the authors discuss the relationship between potential games and consensus dynamics. Stepping forward, I propose an update rule based on the celebrated Friedkin-Johnsen (FJ) dynamics [59] to enhance resilience in the presence of misbehaving agents. One key feature of the FJ model is a tunable parameter $\lambda \in [0,1]$ which allows to smoothly transition from *full collaboration*, where each normally behaving agent puts equal trust in all agents in the network (including itself), leading to standard average consensus, to *full competition*, whereby agents do not trust each other, namely they regard all other agents as adversaries. Such an approach allows to study resilience and performance trade-offs arising from different choices of agents that may trust their neighbors or not, a choice that turns out to be crucial if adversaries are present. In fact, a fundamental *competition-collaboration trade-off* can be observed: in general, *the optimal choice to achieve resilience is a hybrid strategy that makes agents trust neighbors only partially*, as illustrated in Fig. 4.1. In particular, the global cost function (solid blue) is the sum of two conflicting contributions, representing respectively deception due to collaboration with malicious agents (dashed red) and inefficiency caused by competition against agent's neighbors (dashed-dotted yellow). To achieve analytical intuition, I discuss such competition-collaboration trade-off using tools drawn from *opinion dynamics*, in particular *social power* and FJ model, that shed light on the role of malicious agents and parametrization of agent dynamics on optimization performance.

After characterizing the proposed competition-based protocol, I shift attention to the network topology, in order to assess how the latter impacts performance of regular agents with respect the considered task. In particular, I use regular and quasi-regular graphs to numerically show that network connectivity can mitigate malicious attacks and how performance varies when the topology gets sparser. In fact, it can be heuristically observed that not only high connectivity, but also degree balance among agents is beneficial to tame unknown adversaries, which could intuitively exploit highly connected hubs to disrupt the optimization at network level.

This chapter is organized as follows. Average consensus for distributed optimization is first motivated in Section 4.1, and a specific class of adversaries (*malicious agents*) is described in Section 4.1.1. In Section 4.2, I propose the competition-based protocol to strengthen resilience of regular agents. In particular, I describe the link with game theory in Section 4.2.1 and analytically characterize the cost function and its minimizer in Sections 4.2.2–4.2.4. To reinforce formal arguments, numerical tests on the theoretical performance of the proposed protocol are performed in Section 4.3. Further, some insights to interpret the competition-collaboration trade-off from a formal standpoint are given in Section 4.3.1, where the result pictorially shown in Fig. 4.1 is deeply analyzed. Then, impact of communication topology on performance is explored in Section 4.4. To assess effectiveness of the proposed approach, I perform simulations on large-scale systems in Section 4.5, showing that the FJ dynamics with suitably chosen parameter λ can provide superior performance to classical MSR-based methods if r-robustness sufficient conditions are not verified. I conclude by addressing open questions and compelling avenues for future research in Section 4.6.

4.1 Setup and Problem Formulation

Consider a Networked Control System composed of N agents in the set $\mathcal{V} = \{1, \dots, N\}$. The state of agent $i \in \mathcal{V}$ is denoted by $x_i \in \mathbb{R}$, and all states are stacked in the column vector $x \in \mathbb{R}^N$. Initially, each agent i carries local information encoded by *prior* $\theta_i \in \mathbb{R}$.

Assumption 4.1.1. *Priors are distributed as random variables with zero mean and covariance matrix* $\Sigma = \Sigma^\top \in \mathbb{R}^{N\times N}$, $\Sigma \succ 0$, *where* $\Sigma_{ii} = \sigma_i^2 > 0 \, \forall i \in \mathcal{V}$ *and* $\Sigma_{ij} = \sigma_{ij} \, \forall i \neq j$.

Within the network, some agents behave according to the control task at hand, while others cannot be controlled and behave arbitrarily. The former are called *regular* and the latter *malicious agents*. Clearly, the identity of malicious agents is assumed to be unknown to regular agents, so that a straightforward selection of trustworthy messages is not possible. Malicious agents cannot be involved in any cooperative task though their uncontrolled nature, hence the global cost function is based on the interaction among regular agents only. In particular, each regular agent i needs to minimize the mismatch f_{local} involving priors of regular agents,

$$f_{\text{local}}(x_i) \doteq \sum_{j\in\mathcal{R}} (x_i - \theta_j)^2 , \quad i \in \mathcal{R}, \tag{4.1}$$

where $\mathcal{R} \subseteq \mathcal{V}$ is the subset collecting all regular agents. By straightforward calculations, (4.1) can be rewritten as

$$\begin{aligned} f_{\text{local}}(x_i) &= R\left(x_i - \bar{\theta}_{\mathcal{R}}\right)^2 - R^2\bar{\theta}^2 + R\sum_{j\in\mathcal{R}} \theta_j^2 \\ &= R\left(x_i - \bar{\theta}_{\mathcal{R}}\right)^2 + \text{const}, \end{aligned} \tag{4.2}$$

where $R \doteq |\mathcal{R}|$ and $\bar{\theta}_{\mathcal{R}}$ is the average of priors $\{\theta_i\}_{i\in\mathcal{R}}$.

The distributed optimization task is then given by

$$\begin{aligned} \min_x f(x), \quad f(x) &\doteq \frac{1}{R}\sum_{i\in\mathcal{R}} f_{\text{local}}(x_i) \\ &= \sum_{i\in\mathcal{R}} \left(x_i - \bar{\theta}_{\mathcal{R}}\right)^2 + \text{const}, \end{aligned} \tag{4.3}$$

which is clearly solved if and only if all regular agents reach *average consensus* among them, *i.e.*, $x_i = \bar{\theta}_{\mathcal{R}}$ for all $i \in \mathcal{R}$.

However, because of network connectivity, such desired alignment may be hindered by malicious agents behaving arbitrarily. Hence, the aim is to design a local update rule for regular agents so as to minimize the *expected average consensus error* (or simply, consensus error), defined as

$$e_{\mathcal{R}} \doteq \mathbb{E}\left[\sum_{i\in\mathcal{R}} \left(x_i - \bar{\theta}_{\mathcal{R}}\right)^2\right], \tag{4.4}$$

where the expectation is taken with respect to (w.r.t.) the distribution of priors $\{\theta_i\}_{i\in\mathcal{V}}$.

4.1.1 Malicious Agents

Malicious agents follow state trajectories with potentially no relation to the optimization task (4.3), and broadcast potentially misleading information to their neighbors. The subsets of malicious agents is denoted by $\mathcal{M}$ with cardinality $M \doteq |\mathcal{M}|$. Clearly, $\mathcal{V} = \mathcal{M} \cup \mathcal{R}$ and $\mathcal{M} \cap \mathcal{R} = \emptyset$. To address a disruptive behavior induced by attacks, the priors of malicious agents are modeled as outliers with respect to the distribution of priors in Assumption 4.1.1, and they are assumed constant overtime.

Assumption 4.1.2 (Priors of malicious agents)**.** The *actual prior* of malicious agent $m \in \mathcal{M}$ is $\tilde{\theta}_m = \theta_m + v_m$, where $v_m \sim (0, d_m)$ is exogenous noise that quantifies attack aggressiveness. For regular agent $i \in \mathcal{R}$, it holds $v_i \equiv 0$ and $\tilde{\theta}_i = \theta_i$. Exogenous noises are uncorrelated among themselves and with nominal priors, *i.e.,* $\mathbb{E}\left[v_m v_n\right] = 0 \, \forall m \in \mathcal{M}$, $n \in \mathcal{M}$, $m \neq n$, and $\mathbb{E}\left[v_m \theta_i\right] = 0 \, \forall m \in \mathcal{M}$, $i \in \mathcal{V}$. Actual priors and noises are stacked in the vectors $\tilde{\theta} \in \mathbb{R}^N$ and $v \in \mathbb{R}^N$, respectively.

Assumption 4.1.3 (Behavior of malicious agents)**.** Malicious agents keep their state constant overtime and equal to their corrupted prior, *i.e.,* $x_m(k) \equiv \tilde{\theta}_m \ \forall m \in \mathcal{M}$. Accordingly, the rows in the matrix W corresponding to malicious agents have all off-diagonal elements equal to 0, with 1 on the diagonal.

The covariance matrices of noises and actual priors are denoted by $V \doteq \mathbb{E}\left[vv^\top\right]$ and $\tilde{\Sigma} \doteq \mathbb{E}\left[\tilde{\theta}\tilde{\theta}^\top\right] = \Sigma + V$, respectively. Without loss of generality, agents are re-labeled as $\mathcal{R} = \{1, \dots, R\}$ and $\mathcal{M} = \{R+1, \dots, N\}$, so that matrix V can be conveniently partitioned as

$$V = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & V_{\mathcal{M}} \end{array}\right], \qquad V_{\mathcal{M}} \doteq \operatorname{diag}\left(d_{R+1}, \dots, d_N\right), \tag{4.5}$$

and, accordingly,

$$\Sigma = \left[\begin{array}{c|c} \Sigma_{11} & \Sigma_{12} \\ \hline \Sigma_{12}^\top & \Sigma_{22} \end{array}\right], \qquad \tilde{\Sigma} = \left[\begin{array}{c|c} \tilde{\Sigma}_{11} & \tilde{\Sigma}_{12} \\ \hline \tilde{\Sigma}_{12}^\top & \tilde{\Sigma}_{22} \end{array}\right], \tag{4.6}$$

with $\Sigma_{11}, \tilde{\Sigma}_{11} \in \mathbb{R}^{R \times R}$ and $\Sigma_{22}, \tilde{\Sigma}_{22} \in \mathbb{R}^{M \times M}$.

Remark 4.1.4 (Behavior of malicious agents)*.* Assumption 4.1.3 is consistent with a portion of the resilient consensus literature, where algorithms are tested against constant or drifting malicious agents that keep pulling their neighbors far off nominal average consensus [22], [50], [207], [218]. On the other hand, attackers may behave intelligently to avoid being detected, which needs to be contrasted by suitable strategies [99], [101]. This scenario goes beyond the scope of this thesis and its study is left to future work.

4.2 Resilient Average Consensus

4.2.1 The Consensus Problem and Game Theoretic Models

Classical consensus is fragile to misbehaving agents [94], hence alternative strategies are needed to minimize (4.4).

To this aim, I consider a game-theoretic interpretation of (4.1)–(4.3). In particular, assume that each regular agent needs to maximize the following utility (*cognitive dissonance*),

$$u_i(x_i) = -\lambda_i \left(x_i - \theta_i\right)^2 - (1-\lambda_i) \sum_{j\in\mathcal{V}} P_{ij} \left(x_i - x_j\right)^2, \tag{4.7}$$

where $\lambda_i \in [0,1]$ and $P_{ij} > 0$ weighs information exchange between i and j, with $P_{ij} = 0$ if i and j do not communicate. In words, utility (4.7) makes the ith agent anchor to its prior proportionally to parameter λ_i. Interestingly, for $\lambda_i = 0$, we retrieve the utility function used in [117], where the authors analyze the classical consensus protocol from a game-theoretic perspective. In this case, agents have no incentive in retaining prior information, while the opposite is true as soon as $\lambda_i > 0$ in (4.7). Greedily maximizing utility (4.7) at step $k+1$ yields the celebrated Friedrick-Johnsen (FJ) dynamics [59],

$$x_i(k+1) = \lambda_i\theta_i + (1-\lambda_i) \sum_{j\in\mathcal{V}} W_{ij} x_j(k), \tag{4.8}$$

where W_{ij} is obtained by normalizing P_{ij} w.r.t. agent i. In the following, I set $\lambda_i \equiv \lambda$, which yields a scalar variable for the addressed optimization problem.

The rest of this chapter is mostly devoted to exploring performance of update rule (4.8) w.r.t. the optimization task (4.3) with initial condition $x_i(0)$ of agent i given by its prior $\tilde{\theta}_i$. The intuition behind using the above rule is that priors of regular agents are likely to be closer to the nominal average consensus $\bar{\theta}_{\mathcal{R}}$ than priors of attackers, and thus regular agents may prefer to act a bit selfishly rather than be misled by malicious neighbors. In the following, the FJ dynamics with $\lambda = 0$ (standard consensus) is referred to as *full collaboration*, and the case $\lambda = 1$ as *full competition*. Tuning the parameter λ within the interval $[0,1]$ originates a nontrivial *communication-computation trade-off*: should an agent fully collaborate, fully compete, or choose a hybrid strategy and trust neighbors only partially? Such performance trade-off in the presence of malicious agents is the main matter under investigation of this chapter.

Assumption 4.2.1 (Network topology and weights)**.** Weights W_{ij} in update (4.8) define an irreducible doubly-stochastic communication matrix W. In view of the local optimization tasks (4.1), there are no self-loops, *i.e.*, $W_{ii} = 0\,\forall\, i \in \mathcal{V}$.

Remark 4.2.2 (Heterogeneous λ_i). While I focus on the case with all parameters λ_i equal in the interest of a simpler analysis, it can be easily observed that the qualitative behavior of the system is the same even with heterogeneous λ_i. Design of different λ_i's to improve performance even further is nonetheless an important topic, whose investigation is left to future work.

4.2.2 Full Competition *vs.* Full Collaboration

A first remarkable result is that, in this scenario, letting $\lambda = 1$ in (4.8) – which is equivalent to a totally unbalanced dynamics enforcing full competition among agents – may outperform the standard consensus protocol if noises are sufficiently intense.

Proposition 4.2.3 (Full competition *vs.* full collaboration)**.** *In the presence of malicious agents, FJ dynamics* (4.8) *with* $\lambda = 1$ *yields smaller error than with* $\lambda = 0$ *if and only if*

$$\sum_{m\in\mathcal{M}} d_m \geq \frac{M^2}{R}\mathrm{Tr}\left(\Sigma_{11}\right) - \frac{2M^2}{R^2}\mathcal{B}\left(\Sigma_{11}\right) + \frac{2M}{R}\mathcal{B}\left(\Sigma_{12}\right) - \mathcal{B}\left(\Sigma_{22}\right), \tag{4.9}$$

where $\mathcal{B}\left(A\right)$ *denotes summation of all elements of matrix* A*.*

Proof. See Appendix C.2. □

In words, Proposition 4.2.3 implies that, if noise variances of malicious agents are sufficiently large compared to the cross-correlations among regular and malicious agents (elements of Σ_{12}), a trivial fully competitive approach (agents keep priors constant overtime) yields better performance than the standard consensus protocol. Intuitively, the latter is still able to drive regular agents to a meaningful value when attacks are mild (small variance d_m), while the full-competitive strategy takes over as soon as attacks become sufficiently aggressive (large d_m) so that the drift error experienced by the consensus dynamics is larger than if agents just froze their priors.

With i.i.d. priors and noises, the following corollary readily follows.

Corollary 4.2.4. *If* $\Sigma = I$ *and* $d_m = d\,\forall m \in \mathcal{M}$*, condition* (4.9) *becomes*

$$d > M\left(1 - \frac{2}{R}\right) - 1. \tag{4.10}$$

4.2.3 The Truth Lies in the Middle

After assessing that a competition-based approach may be more resilient than standard consensus protocol, I now wish to characterize the "competitiveness" of the optimal strategy. In particular, this section aims to analytically characterize the optimal parameter, denoted by

$$\lambda^* \doteq \arg\min_{\lambda} e_{\mathcal{R}}. \tag{4.11}$$

Such optimal parameter always exists. Indeed, if $(1-\lambda)W$ is Schur (*i.e.,* $\lambda > 0$), the FJ dynamics induces the steady state

$$x = L\tilde{\theta}, \quad L \doteq \left(I - (1-\lambda)W\right)^{-1}\lambda. \tag{4.12}$$

Matrix L can be interpreted as a generalization of the consensus matrix, and depends on both weights in W and parameter λ. In particular, L has a continuous extension at $\lambda = 0$ given by $\lim_{\lambda\to 0^+} L = \overline{W} \doteq \lim_{k\to+\infty} W^k$ [144]. In words, this means that, as λ approaches zero, the steady state achieved by the FJ dynamics tends to the one obtained by the consensus protocol. Indeed, this is intuitive by looking at update rule (4.8).

Clearly, the continuous extension of L also implies a corresponding continuous extension of $e_{\mathcal{R}}$ at $\lambda = 0$. Hence, extended continuity of $e_{\mathcal{R}}$ in $[0,1]$ and Weierstrass theorem ensure existence of a global minimum within the interval $[0,1]$.

The next result characterizes the case when the optimal resilient strategy is nontrivial, meaning that regular agents should partially trust their neighbors in order to minimize error (4.4). Intuitively, this happens if some minimal interaction can do better than full competition (which translates into Σ being “non-degenerate" is a suitable sense), and if attacks are sufficiently aggressive (*i.e.,* variances of noises are large enough) so that the consensus protocol yields poor performance, analogously to what remarked about Proposition 4.2.3.

Theorem 4.2.5 (Nontrivial communication-computation trade-off)**.** *Let* $\Gamma \doteq \lim_{\lambda\to 0^+} \frac{dL}{d\lambda}$ *with block partition*

$$\Gamma = \left[\begin{array}{c|c} \Gamma_1 & \Gamma_2 \\ \hline 0 & 0 \end{array}\right], \quad \Gamma_1 \in \mathbb{R}^{R\times R}, \tag{4.13}$$

and let $C_R \doteq \frac{\mathbb{1}_R\mathbb{1}_R^\top}{R}$, $C_{RM} \doteq \frac{\mathbb{1}_R\mathbb{1}_M^\top}{M}$. *Then, the optimal parameter* λ^* *belongs to the open interval* $(0,1)$ *if there exists one regular agent* $i \in \mathcal{R}$ *such that* $\sigma_i^2 > \sigma_{ij}\, \forall j \neq i$ *and*

$$\sum_{m\in\mathcal{M}} \alpha_m d_m > \mathrm{Tr}\left(-\Sigma_{11}\Gamma_1^\top C_R - \Sigma_{12}\Gamma_2^\top C_R + \Sigma_{12}^\top\Gamma_1^\top C_{RM} + \Sigma_{22}\Gamma_2^\top C_{RM}\right), \tag{4.14}$$

where $\alpha_m \geq 0$ *is the negative scalar product between the mth columns of* Γ_2 *and* C_{RM}*.*

Proof. See Appendix C.3. □

Corollary 4.2.6. *If* Σ *is diagonal, then* $\lambda^* \in (0,1)$*.*

Remark 4.2.7 (Explicit condition for $\lambda^* > 0$). Explicitly checking when condition (4.14) holds is hard, because it involves the spectrum of W. However, numerical tests show that indeed such a condition is always satisfied in meaningful cases.

Remark 4.2.8 (Optimal parameter with zero noise). Theorem 4.2.5 implies that λ^* may be positive even when noise variances are zero. This is actually consistent with intuition: not only attackers steer regular agents far off from the nominal average consensus value (whereby d_m quantifies intensity of such deception), but also they behave against the prescribed update rule, ruling out full collaboration as an effective strategy – unless cross-correlations between priors of regular and malicious agents are way larger than other cross-correlations, see condition (4.14).

Remark 4.2.9 (Optimal strategy with general matrices). It is worth mentioning that, even though it is assumed no self-loops in the original matrix W to be consistent with the optimization tasks, Theorem 4.2.5 can be generalized to arbitrary doubly-stochastic matrices W. Further, numerical experiments show that all above result also holds for row-stochastic matrices W.

4.2.4 FJ Dynamics *vs.* Attack Aggressiveness

This section studies how performance varies with attack intensity, as quantified by the variances of noises v_m.

The first result is intuitive: more aggressive attacks (with larger noise variances) induce larger error for any λ.

Proposition 4.2.10 (Performance *vs.* attacks aggressiveness). *Error* $e_{\mathcal{R}}(d_1, \dots, d_M)$ *is strictly increasing with* d_m, $m \in \mathcal{M}$.

Proof. See Appendix C.5. □

The next result characterizes what happens to the optimal parameter λ^*. Intuitively, the more the nominal (prescribed) system behavior is disrupted by attacks, the more regular agents benefit from being competitive rather than collaborating with (potential) malicious neighbors. Formally speaking, this requires λ^* to grow proportionally to the noise intensities d_m. However, such a claim is hard to prove analytically because of the structure of the cost function. In particular, studying its second derivative is complicated by the fact that the function $e_{\mathcal{R}}$ is expressed as the trace of a non-positive semidefinite matrix (in fact, not even symmetric), and similarly, uniqueness of the root of its first derivative cannot be proved, in general. In the face of such analytical difficulties, the next results contributes towards this intuition, which is numerically confirmed in Section 4.3.

Proposition 4.2.11 (Optimal strategy *vs.* attack aggressiveness)**.** *Let* $\lambda_{cr}(d_1, \ldots, d_M)$ *a critical point of* $e_{\mathcal{R}}(d_1, \ldots, d_M)$*, then* $\lambda_{cr}(d_1, \ldots, d_M)$ *is strictly increasing with* d_m*,* $m \in \mathcal{M}$*.*

Proof. See Appendix C.6. □

Proposition 4.2.11 implies that all points of local minimum are strictly increasing with noise variances d_m. An immediate consequence is that, if there is a unique critical point for one choice of $\{d_m\}_{m\in\mathcal{M}}$, then such a point is λ^*, is unique for any choice of $\{d_m\}_{m\in\mathcal{M}}$, and is strictly increasing with any d_m. In words, more aggressive attacks force regular agents to progressively become more competitive, in order not to be deceived by malicious agents that can draw them away from nominal average consensus. The next proposition refines this result, describing the limit behavior with "extreme" attacks.

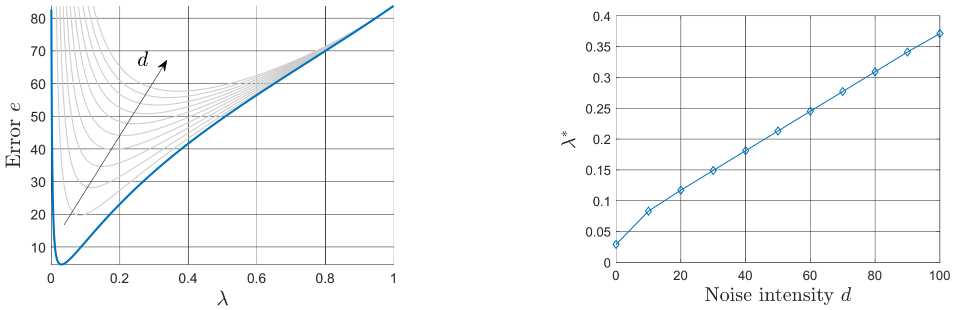
Proposition 4.2.12 (Optimal strategy with extreme attacks)**.** *Let* $\lambda_{cr}(d_1, \ldots, d_M)$ *a critical point of* $e_{\mathcal{R}}(d_1, \ldots, d_M)$*, then* $\lim_{d_m \to +\infty} \lambda_{cr}(d_1, \ldots, d_M) = 1$*,* $m \in \mathcal{M}$*.*

Proof. See Appendix C.7. □

According to intuition, the (trivial) optimal strategy for regular agents is to fully compete against each other when adversarial attacks turn too disruptive. However, numerical simulations in the next section show that λ^* is significantly smaller than 1 in practical scenarios.

4.3 Numerical Experiments

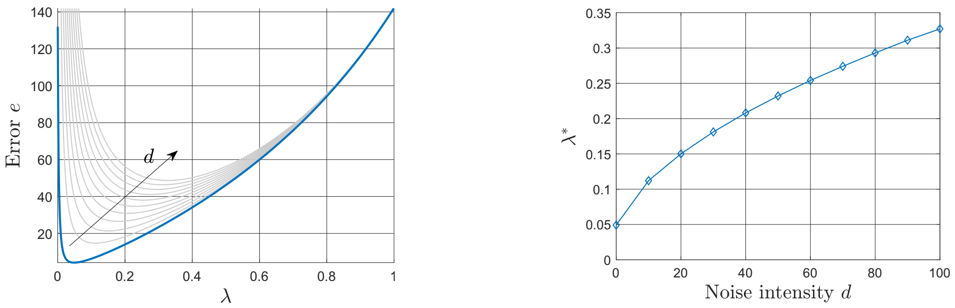
This section, presents numerical experiments on the consensus error $e_{\mathcal{R}}$ aimed to achieving intuition about the behavior of FJ dynamics under different topologies and attack



(a) Average consensus error (4.4)

(b) Optimal λ as a function of d.

Figure 4.2: FJ dynamics consensus error with 3-regular graph, exponential decay of prior covariances, and one malicious agent. The arrow on the left box shows how the error curve varies as the outlier noise intensity d increases.



(a) Average consensus error (4.4)

(b) Optimal λ as a function of d.

Figure 4.3: FJ dynamics consensus error with 3-regular graph, diagonal prior covariance matrix, and one malicious agent.

scenarios, and to drawing insight about effective choices of the parameter λ.

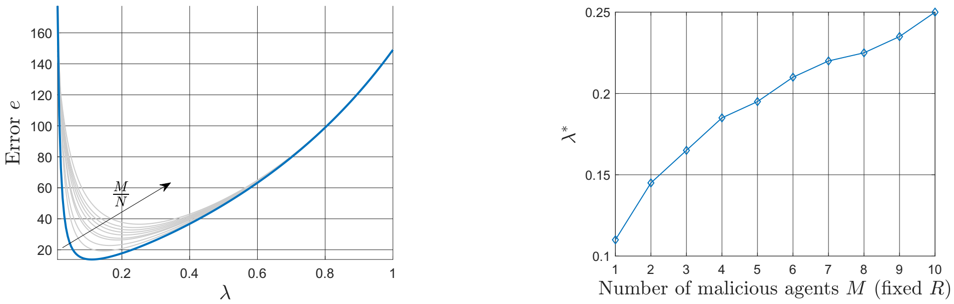
In Fig. 4.2, I consider a 3-regular communication graph with 100 agents and (nominal) uniform weights $W_{ij} = {}^1\!/_3$. Matrix Σ is chosen such that, for each agent i, the cross-covariances obey an exponential decay, $\sigma_{ij} = 10^{-0.2\mathrm{d}(i,j)}$, $\mathrm{d}(i,j)$ being the length of a shortest path between i and j, with $\sigma_i^2 \equiv 1$. Further, I randomly select one malicious agent and vary the noise intensity d within the range $[0, 100]$.

Figure 4.2a shows the error curve as d increases. All curves exhibit a unique point of minimum λ^*, plotted in Fig. 4.2b. Further, both error curve and point of minimum are increasing with the noise intensity d, according to Propositions 4.2.10–4.2.11, showing that competition level needs to increase with d.

Figure 4.3 shows the same experiment but with a diagonal covariance matrix Σ. The monotonic behavior of $e_{\mathcal{R}}$ and λ^* observed above still holds. Further, note that the error curve has a convex shape. In fact, even though it was not possible to prove it formally, all tests performed with diagonal covariance matrices resulted in strictly convex error functions.[1]

Next, I study what happens when increasing the number of malicious agents M. To better visualize changes in the behavior of the system, the set $\mathcal{R}$ is set to be a network composed of $R = 100$ regular agents, and malicious agents are progressively added across the network in a scattered fashion. Figure 4.4 shows the error curve when 10 such agents are progressively introduced. In particular, in this example, malicious agents are introduced so as to affect different portions of the network, which allows λ^* to have relatively low values, see Fig. 4.4b. Conversely, in the opposite scenario, some regular agents may be forced to almost freeze their priors (large λ) to not drive the error too large. Figure 4.5 shows two cases where the added malicious agents are connected to the same

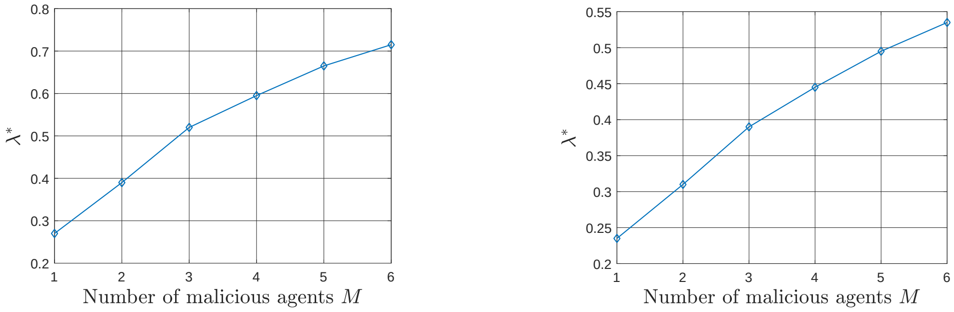
[1] This was checked by means of suitable symbolic software.



(a) Average consensus error (4.4)

(b) Optimal λ as a function of M.

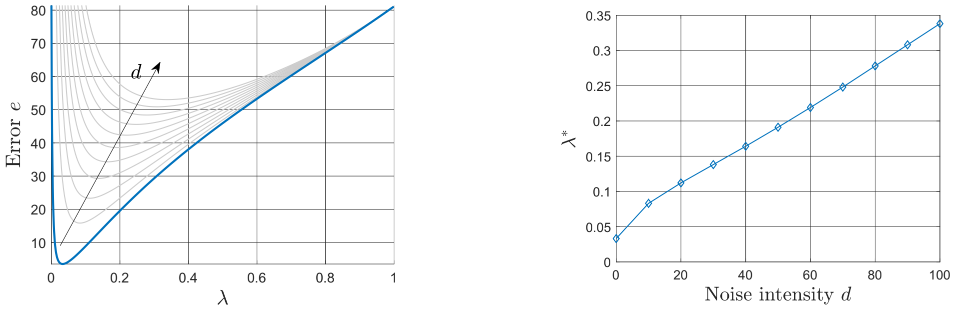
Figure 4.4: FJ dynamics consensus error with 3-regular graph, diagonal prior covariance matrix, and $d = 10$. The arrow on the left box shows how the error varies as the number of malicious nodes M increases (with $R = 100$).



(a) Exponential prior covariances.

(b) Diagonal prior covariance matrix.

Figure 4.5: Optimal λ as a function of M with $d = 10$. Each pair of malicious agents affects one regular agent (*e.g.*, the first two belong to $\mathcal{N}_i$, $i \in \mathcal{R}$).



(a) Average consensus error (4.4)

(b) Optimal λ as a function of d.

Figure 4.6: FJ dynamics consensus error with $(3, 4)$-degree communication graph, exponential decay of prior covariances, and one malicious agent.

regular agents. In particular, each consecutive couple is added to the neighborhood of one regular agent (*e.g.,* the first two malicious agents added to the network are neighbors of agent $1 \in \mathcal{R}$). In this case, λ^* increases faster than Fig. 4.4b, because the regular agents affected by multiple malicious need to keep their error small: in other words, they need to aggressively compete (inducing large value of λ) because of their misbehaving neighbors. Interestingly, λ^* grows faster when priors of regular agents are correlated (Fig. 4.5a), which can be explained because such agents can trust that their states may be similar to each other even before starting dynamical updates, and competing is less risky than collaborating.

Finally, it is interesting to notice that the error behavior observed above is also present when W is just row-stochastic, thus yielding nonzero consensus error even without external attacks. Figure 4.6 shows consensus error and λ^* when each node in the graph has degree 3 or 4 and W has uniform weights.

Many other numerical tests performed with different graphs, prior distributions, and choice of the malicious agents, show the same monotonic and quasi-convex behavior of the error function. This reinforces and extends the scope of the formal analysis, showing that indeed the collaboration-competition trade-off naturally emerges as a resilient mechanism in Networked Control Systems.

Remark 4.3.1 (Value of optimal λ). A remarkable feature of the FJ dynamics, that can be observed from the above numerical experiments, is that the optimal value of λ is relatively small, within the range $[0.1, 0.2]$ for many relevant scenarios. In fact, λ^* reaches 0.3 in Fig. 4.2 when the noise variance of the malicious agent is two order of magnitude larger than the variance of priors. This translates into the practical advantage that adding a little competition is sufficient to achieve substantial performance improvement compared to the standard consensus protocol, which may be attractive to achieve good

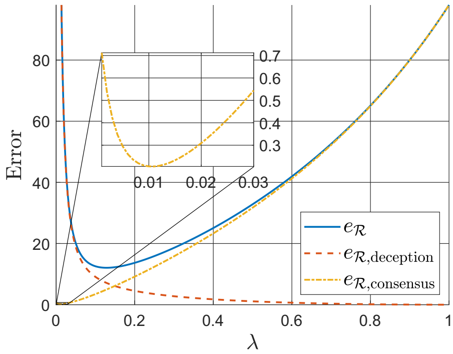


Figure 4.7: Consensus error and its two contributions in (4.15).

level of resilience without forcing too conservative local agent updates.

4.3.1 Competition-Collaboration Trade-off: Analytical Insight

As mentioned earlier, the consensus error function $e_{\mathcal{R}}$ is hard to parse and an exhaustive analysis seems not possible.

Yet, some intuition can be achieved from an exact decomposition of $e_{\mathcal{R}}$, which is analyzed in this section. To keep notation light, a single malicious agent, *i.e.*, $\mathcal{M} = \{m\}$, and diagonal covariance matrix Σ are assumed. In this case, the consensus error can be expanded as follows (cf. (C.6) in Appendix C.3),

$$e_{\mathcal{R}} = \underbrace{\left(\sigma_m^2 + d\right) \left\|L_m^{-m}\right\|^2}_{\doteq e_{\mathcal{R},\text{deception}}} + \underbrace{\sum_{i\in\mathcal{R}} \sigma_i^2 \left\|L_i^{-m} - \frac{\mathbb{1}}{R}\right\|^2}_{\doteq e_{\mathcal{R},\text{consensus}}}, \tag{4.15}$$

where $L_i \in \mathbb{R}^N$ is the ith column of L and $L_i^{-m} \in \mathbb{R}^R$ is obtained from L_i by removing its mth row (corresponding to the malicious agent). The error curves are shown in Fig. 4.7. Equation (4.15) allows for an intuitive interpretation of the error, which leverages the notion of *social power* [34], [194].

In opinion dynamics, the social power is used to quantify how much each agent's opinion affects the opinion of all agents. In particular, when opinions evolve according to the FJ dynamics, the element L_{ij} quantifies the influence of agent j on agent i: as L_{ij} increases, agent i is more affected by agent j's initial opinion. The overall social power of agent j is a symmetric and increasing function of all elements $\{L_{ij}\}_{i\in\mathcal{V}}$.[2]

Borrowing such concepts from opinion dynamics allows to interpret the two contributions highlighted in (4.15). The first contribution, $e_{\mathcal{R},\text{deception}}$, quantifies the impact

[2] For example, [34], [194] use the arithmetic mean of $\{L_{ij}\}_{i\in\mathcal{V}}$.

of the malicious agent on regular agents. The "social power" of m, as quantified by the vector L_m^{-m}, depends on the communication matrix W and on the parameter λ. In particular, each coordinate of L_m^{-m} decreases with λ, intuitively meaning that influence of the malicious agent diminishes as regular agents anchor more tightly to their priors, and becomes exactly zero when $\lambda = 1$, namely, when regular agents have no iterations with their neighbors (see Appendix C.4 for formal analysis).

The second contribution $e_{\mathcal{R},\text{consensus}}$ measures "democracy" among regular agents, *i.e.,* it is proportional to the mismatch between how much each regular agent affects the others and the ideal value $1/R$, which means that each agent affects all others equally. This cost is zero if and only if the submatrix of L corresponding to interactions among regular agents is the consensus matrix: this can happen only if they do not interact with the malicious agent [144], in which case, the vector L_m^{-m} is zero (attacks have no effect). In this special case, $e_{\mathcal{R},\text{consensus}}$ is zero at $\lambda = 0$ and increases monotonically as the network shifts from a democratic system, where agents fully collaborate ($\lambda = 0$), to a disconnected system where agents fully compete ($\lambda = 1$). Conversely, when malicious agents affect regular ones, $e_{\mathcal{R},\text{consensus}}$ is U-shaped. For small λ, the malicious agent rules the dynamics, and interactions among regular agents are negligible. As λ increases, regular agents start collaborating and their interactions become more relevant, making $e_{\mathcal{R},\text{consensus}}$ decrease. However, as λ grows further, agents compete too aggressively, shifting away from democratic system. In practice, numerical tests show that the point of minimum of $e_{\mathcal{R},\text{consensus}}$ is small (zoomed box in Fig. 4.7), meaning that the malicious agent barely affects error due to competition.

Overall, the error has two concurrent causes that generate a phase transition: the collaboration with the malicious agent is critical with small λ, while for large λ the error is mainly due to agents competing against each other, rejecting possibly useful shared information. Indeed, this analysis matches intuition from (4.8), where λ measures conservatism in agent updates.

4.4 The Role of Communication Network

In the previous sections, I discussed the benefits of using a competition-based approach (FJ dynamics) to tame malicious agents. I now shift attention to the communication network, in order to achieve intuition about resilient topologies. Section 4.4.1 introduces a second performance metric which is used to evaluate resilience to attacks. Then, in Section 4.4.2, I observe how performance varies with connectivity.

4.4.1 Performance Metrics

Besides consensus error, it is also interesting to assess energy spent to conduct attacks. For the sake of simplicity, I assume again a single malicious agent, $\mathcal{M} = \{m\}$. Define the following block partition of W,

$$W = \left[\begin{array}{c|c} W_{\mathcal{R}} & W_m \\ \hline 0 & 1 \end{array}\right], \quad W_{\mathcal{R}} \in \mathbb{R}^{R\times R}, \tag{4.16}$$

where $W_{\mathcal{R}}$ corresponds to interactions among regular agents in $\mathcal{R}$ and W_m to neighbors of the malicious agent. Using (4.16), (4.8) can be rewritten as

$$\begin{gathered} x_{\mathcal{R}}(k+1) = Ax_{\mathcal{R}}(k) + Bx_m(k) + \bar{v} \\ A \doteq (1-\lambda)W_{\mathcal{R}}, \quad B \doteq (1-\lambda)W_m, \quad \bar{v} \doteq \lambda\theta_{\mathcal{R}}. \end{gathered} \tag{4.17}$$

I interpret (4.17) as a controlled system where the state $x_{\mathcal{R}}$ stacks all states of regular agents, and the malicious agent can command the control input $x_m(k)$. The controllability Gramian in K steps $\mathcal{W}_K$, defined as

$$\mathcal{W}_K = \sum_{k=0}^{K-1} A^k BB^\top \left(A^\top\right)^k, \tag{4.18}$$

can be used to quantify the control effort: indeed, the trace of $\mathcal{W}_K$ (*controllability index*) is inversely related to the control energy spent in K steps (averaged over the reachable subspace), as shown in the literature [67], [139], [178]. Intuitively, a small controllability index means that the malicious agent consumes a lot of energy to steer $x_{\mathcal{R}}$ to some reachable configuration, which may be desired to drain out adversarial resources and possibly hamper the attack.

Notably, the controllability index can be equivalently written as

$$\operatorname{Tr}\left(\mathcal{W}_K\right) = (1-\lambda)^2 \sum_{k=0}^{K-1} \left\|(1-\lambda)^k W_{\mathcal{R}}^k W_m\right\|^2, \tag{4.19}$$

which resembles the deception error component $e_{\mathcal{R},\text{deception}}$ in (4.15),

$$e_{\mathcal{R},\text{deception}} \propto \left\|\sum_{k=0}^{\infty}(1-\lambda)^k \sum_{j=0}^{k-1} W_{\mathcal{R}}^j W_m\right\|^2. \tag{4.20}$$

Both $\operatorname{Tr}\left(\mathcal{W}_K\right)$ and $e_{\mathcal{R},\text{deception}}$ are decreasing with λ (*i.e.*, the more competition, the

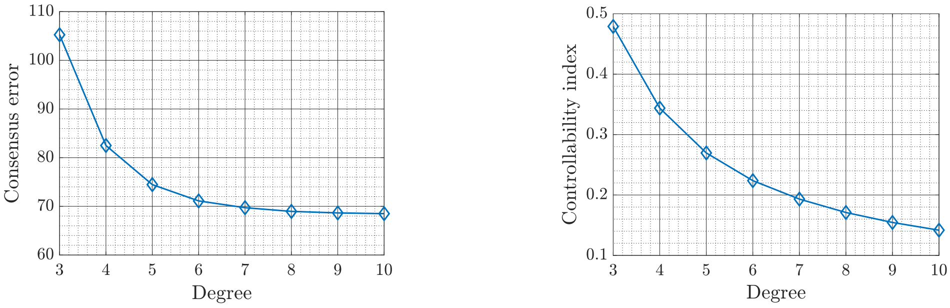


Figure 4.8: Average consensus error (left) and controllability index (right) for regular graphs. Each point is obtained by averaging over 1000 random graphs.

better), and depend on the vectors $W_{\mathcal{R}}^k W_m$ that describe how an attack spreads in k steps. The discount factor $(1-\lambda)^k$ makes the tail of the series in (4.20) negligible, enhancing similarity between those two metrics.

Remark 4.4.1 (Power of adversary). The controllability Gramian addresses richer attack strategies than Assumption 4.1.3, letting the malicious agent drive regular agents to any configuration in the reachable subspace via a suitable trajectory $x_m(k)$.

4.4.2 Network Connectivity *vs.* Resilience

We look at the following worst-case optimization problems,

$$\min_{W} \max_{m\in\mathcal{V}} \quad e_{\mathcal{R}}, \tag{4.21}$$

$$\min_{W} \max_{m\in\mathcal{V}} \quad \mathrm{Tr}\left(\mathcal{W}_K\right). \tag{4.22}$$

The internal maximization (worst case) selects the agent that either causes the largest consensus error (4.21) or makes the smallest control effort (4.22), with K being the reachability index. The external minimization addresses the network design.

In the following, rather than solving (4.21)–(4.22), I examine performance of some simple, but significant, classes of graphs. In particular, I aim to achieve intuition about some core properties of the network, such as connectivity and balance of node degrees. A broader optimization study is deferred to future work. As a first step, I consider regular graphs with 100 nodes and uniform weights. Note that regular graphs are commonly found in applications [24], [82], [143], [216], [226]. To assess the role played by connectivity, I evaluate the worst-case performance of Δ-regular graphs, $\Delta\in\{3,\dots,10\}$, by averaging $\max_{m\in\mathcal{V}} e_{\mathcal{R}}$ (for (4.21)) and $\max_{m\in\mathcal{V}} \mathrm{Tr}\left(\mathcal{W}_K\right)$ (for (4.22)) over 1000 random graphs. The "worst" malicious agent is found via brute force. Results are shown in Fig. 4.8.

A first insight is that increasing the graph connectivity mitigates attacks with respect to both metrics. Intuitively, this is because high degree induces many interactions among regular agents that the malicious agent cannot directly control.

In a real system, the number of communication links is subject to practical constraints. To study how performance varies when the total amount of communication links is limited, I consider *almost-regular* graphs, namely, where nodes have degree either Δ or $\Delta - 1$ for some fixed Δ. Intuitively, almost-regular graphs correspond to "middle-ways" between Δ- and $(\Delta - 1)$-regular graphs, that, looking at Fig. 4.8, could be ideally placed between two consecutive ticks (degrees) Δ and $\Delta - 1$ on the x-axis.[3]

Hence, given a Δ-regular graph as starting point, I progressively prune edges and observe performance variations. More specifically, I iteratively remove one edge at a time so as to minimize performance degradation at each removal. This corresponds to the following simplification of (4.21)–(4.22),

$$\min_{e\in\mathcal{E}} \max_{m\in\mathcal{V}} \quad e_{\mathcal{R}}(\mathcal{E} \setminus \{e\}), \tag{4.23}$$

$$\min_{e\in\mathcal{E}} \max_{m\in\mathcal{V}} \quad \mathrm{Tr}\left(\mathcal{W}_K(\mathcal{E} \setminus \{e\})\right), \tag{4.24}$$

where $\mathcal{E}$ is the set of edges defining the communication network (*i.e.,* the support of W), and W always has uniform weights (before and after removal of each edge). To keep the graph almost regular, it suffices to impose that at most one edge be removed per node in minimization problems (4.23)–(4.24). This additional constraint is also motivated by numerical tests showing that, in non-regular graphs, the "worst" malicious agent exploits a highly connected hub to spread damage more quickly.

Figures 4.9–4.10 show performances obtained starting from a 4-regular graph with 50 nodes (100 edges in total, corresponding to the rightmost point in the plots) and gradually pruning edges according to the above discussion (proceeding leftwards on the x-axis). Also, performances with a 3-regular graphs obtained by removing perfect matchings from the initial 4-regular graphs are shown for comparison.[4] Remarkably, performance degrades (almost) monotonically for both indices as edges are removed. This may be explained by a combination of lower connectivity and degree unbalance, which allows the adversarial to exploit highly connected agents to make more damage against lowly connected regular agents.

[3] Note that an almost-regular graph implies a row-stochastic matrix W.

[4] A matching is a set of edges that do not share nodes. A maximum matching is a matching of maximal cardinality, and a perfect matching is a maximum matching such that each node is incident to one edge (*total coverage*). Note that our greedy edge removal may not be able to remove exactly one edge for each node in the graph, because of the enforced constraint that keeps the resulting graphs almost regular. Indeed, some leftmost points on the x-axes of Figs. 4.9–4.10 have no corresponding makers on the blue curve.

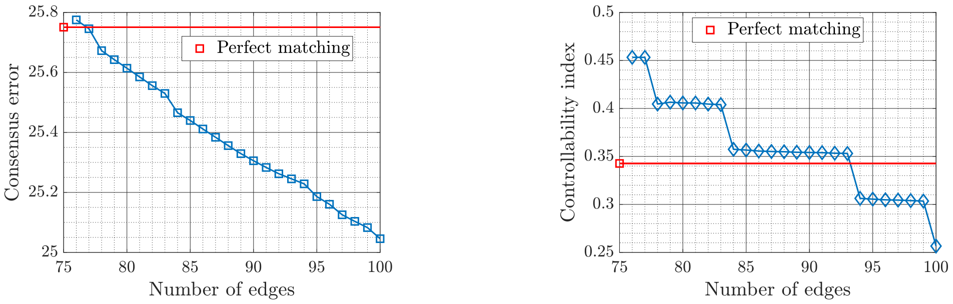


Figure 4.9: Consensus error (left) and controllability index (right) with greedy edge removal for different topologies starting from a 4-regular graph, with $\lambda = 0.7$. In the plots, edge removal iterations (blue diamonds) proceed from right (initially, all 100 edges are present) towards left. At each iteration, one edge is removed so as to minimize performance degradation according to (4.23)–(4.24) while enforcing that no node has fewer than three neighbors (*i.e.,* degree is either three or four for each node). At the last iteration (leftmost diamonds), most or all nodes have degree three, with possibly a few nodes left with degree four (because of the enforced degree-balance constraint). The red squares show the performance metrics for a 3-regular graph obtained by removing a perfect matching (set of edges) from the initial 4-regular graph.

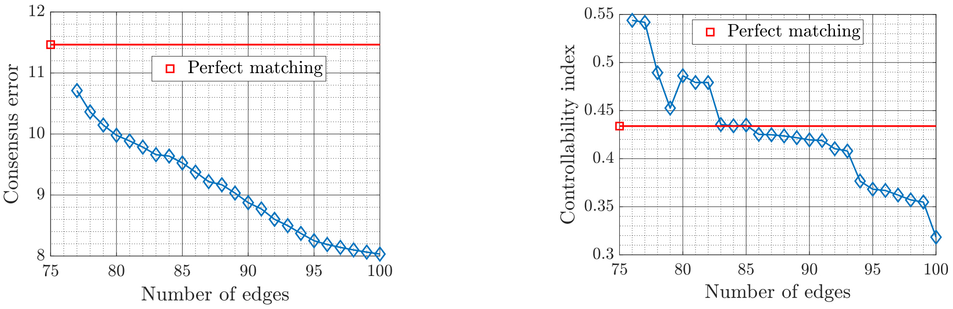


Figure 4.10: Consensus error (left) and controllability index (right) with greedy edge removal starting from a 4-regular graph (100 edges), with $\lambda = 0.2$.

Interestingly, while the consensus error increases quite smoothly as more edges are removed, the controllability index exhibits sharp "jumps". This is especially evident with large λ, as Fig. 4.9 shows. Such behavior suggests the presence of critical subsets of edges, and may give indication about which links should be primarily kept or may be removed.

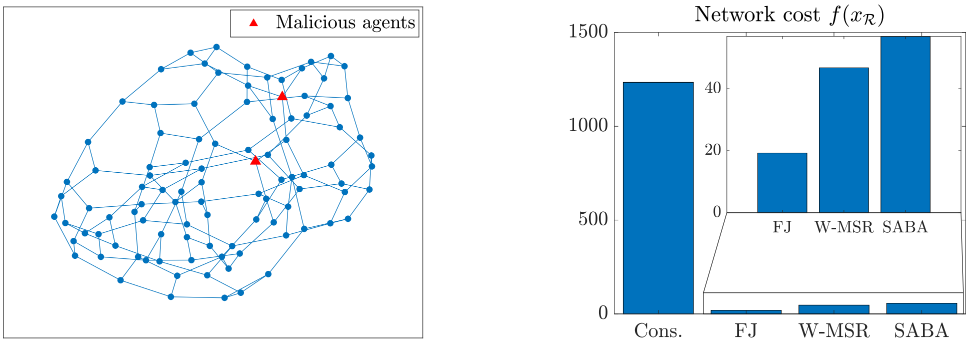
Further, in almost all tests performed with different values of λ and random graphs, the 3-regular graph obtained by removing a perfect matching yielded a performance improvement compared to the last edge removal (leftmost point on blue curve). This shows that increasing connectivity may not be beneficial if it entails a loss in balance: in Fig. 4.9, the 3-regular graph reduces the controllability index by 22% w.r.t. the last graph obtained by greedily pruning edges (0.34 against 0.45), which has a single node with degree 4 and all others with degree 3, and has comparable performance with graphs having most nodes with degree 4. This suggests a phase transition in the network design, whereby it is not convenient to add edges until a certain degree balance is met. However, as shown in Fig. 4.10, a regular graph of degree $\Delta - 1$ obtained by removing a perfect matching (which is not related to performance metrics) from a Δ-regular graph may yield much worse performance than even the more unbalanced graph obtained by greedy pruning edges. This gives further insight: removing edges arbitrarily may perform substantially worse compared to a careful removal strategy.

4.5 Comparison with Existing Literature

In this section, I test our proposed protocol and compare its performance with other approaches in the literature.

Many techniques have been proposed to mitigate malicious attacks in optimization and consensus. However, they usually focus on reaching generic consensus, possibly while keeping the states of regular agents within a safe region (usually defined by initial conditions), and do not consider robustness with respect to *average consensus*, which here is key to the distributed optimization task, as argued in Section 4.1. Indeed, most *resilient consensus* strategies aim to get the agents agree on, *e.g.,* a common position (as robot rendez-vouz) with some level of resilience, not necessarily tying consensus to initial conditions.

I compare two strategies: Weighted Mean Subsequence Reduced (W-MSR) [94] and Secure Accepting and Broadcasting Algorithm (SABA) [50]. As noted in the introduction, many resilient algorithms consist in adaptations of W-MSR to various scenarios, and their core behavior and guarantees are the same. W-MSR suffers from two main limitations related to r-robustness, which is the cornerstone of all theoretical results. First, while



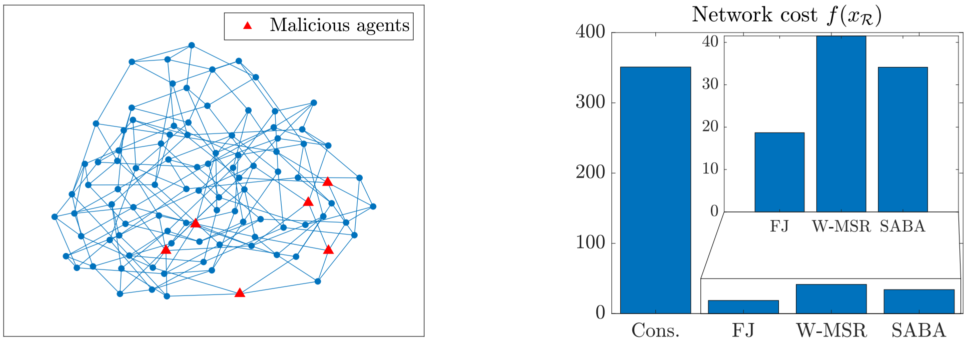
(a) Communication network with 100 agents (two malicious agents).

(b) Consensus error (4.4) and network cost (4.3) of regular agents.

Figure 4.11: Comparison among standard consensus, FJ, W-MSR [94], and SABA [50] with 3-regular communication graph and two adversaries.

sufficient conditions for resilient consensus are clear, often there is no clue about necessary conditions. This translates into unknown behavior when robustness requirements are not met. While r-robustness has proved a good characterization for update rules based on W-MSR, such fact raises practical limitations. On the one hand, the communication network may be fixed but not robust enough. On the other hand, checking r-robustness is computationally intractable for large-scale graphs [185]. Thus, in some cases, for example with sparse architectures, more conservative behaviors with provable performance bounds may be preferred. Also, W-MSR requires to estimate the number of malicious agents affecting the network. This may also be an issue: if the estimate is too low, regular agents may be deceived and average consensus disrupted, whereas, if it is too high, the algorithm may be too conservative, possibly preventing convergence to consensus. Further, agent failures could happen in a time-varying fashion and make the algorithm fail at times, yielding poor overall performance. SABA needs not estimate the number of malicious agents, but stores all received values in a buffer and processes them with a voting strategy. However, this design may impose impractical memory requirements, and further, the authors show that their algorithm is still subject to r-robustness conditions.

In the next simulations, I consider sparse communication graphs, whose low connectivity hampers W-MRS and SABA, and matrices W with homogeneous weights. For each scenario, the parameter λ^* is chosen by finely sampling the interval $(0,1)$ and picking λ minimizing the theoretical error function $e_{\mathcal{R}}$ (4.4). As performance metric, I consider the expected cost of the distributed optimization task, which equals $e_{\mathcal{R}}$ up to an additive constant (see (4.3)). Nominal priors are drawn as $\theta_i \sim \mathcal{N}(0, 0.1), i \in \mathcal{V}$, and each malicious agent m is assigned actual prior $\tilde{\theta}_m = \theta_m + k$, $k \in [2,4]$, which is kept constant according to Assumption 4.1.3.



(a) Communication network with 100 agents (six malicious agents).

(b) Consensus error (4.4) and network cost (4.3) of regular agents.

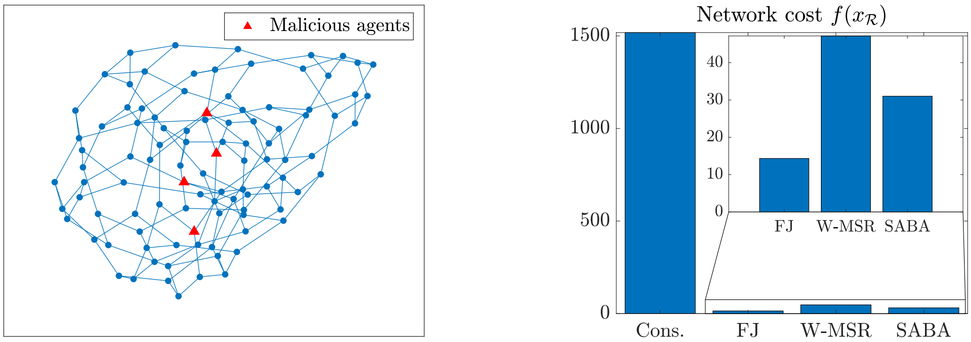
Figure 4.12: Comparison among standard consensus, FJ, W-MSR [94], and SABA [50] with 4-regular communication graph and six adversaries.

Figure 4.11 illustrates a network composed of 100 agents interacting through a 3-regular graph (Fig. 4.11a) with two malicious agents (red triangles). It is worth noting that r-robustness of 3-regular graphs is not sufficient to tolerate malicious agents, and therefore theoretical guarantees of MSR-based approaches do not hold. W-MRS is implemented assuming that each regular agent communicates with at most one adversary, because larger values make updates trivial, *i.e.,* $x_i(k) \equiv x_i(0)$. Such limitations allow FJ dynamics to outperform both algorithms, as shown by Fig. 4.11b.

In the second experiment, I use a denser regular network with degree four attacked by six malicious agents (Fig. 4.12a). Some of the latter communicate with the same regular agent (*e.g.,* the two in the bottom-right portion of the graph), making this scenario more challenging. While both SABA and W-MSR still perform poorly (Fig. 4.12b), FJ dynamics can mitigate attacks by increasing the value of λ^*, as already suggested by Fig. 4.5.

Finally, in Fig. 4.13 I consider a network where nodes have degree three or four (Fig. 4.13a), with W being a row-stochastic matrix with uniform weights. In this case, one may question whether a doubly-stochastic matrix could improve performance of the standard consensus protocol, in light of its optimality under nominal conditions. However, in the presence of malicious agents, it is easy to show that standard consensus converges to the centroid of the states of the latter regardless of weights in W (cf. Assumption 4.1.3 and (C.3) in Appendix). Conversely, Fig. 4.13b, shows that FJ dynamics is a robust strategy against misbehaving agents even though it cannot retrieve the optimal solution under nominal conditions.

Remark 4.5.1 (Benefits of FJ dynamics). The above experiments highlight some advantages of the proposed approach. Firstly, the presence of a tunable parameter makes



(a) Communication network with 100 agents (four malicious agents).

(b) Consensus error (4.4) and network cost (4.3) of regular agents.

Figure 4.13: Comparison among standard consensus, FJ, W-MSR [94], and SABA [50] with (3, 4)-degree communication graph and four adversaries.

the algorithm flexible, as it can smoothly adapt to different attack intensities while still providing decent performance bounds. Further, while the optimal parameterization requires exact knowledge of the adversary, which may not be reasonably assumed, yet the proposed approach proves pretty robust to the choice of a specific λ, as the plots in Section 4.3 show. This also works with row-stochastic matrices, as shown in Fig. 4.13b, enabling simple weighting rules to be implemented locally. In contrast, in other approaches the cost function may be highly sensitive to some design parameters, *e.g.,* the estimated number of malicious agents in W-MSR. Further, most results in literature do not describe system behavior when resilient consensus is not guaranteed. In fact, they usually either ensure that agent's states remain inside the safety region (which in practice may not be better than setting $\lambda = 1$ in the proposed FJ dynamics-based approach), or let agents reach consensus but potentially be steered far away from initial conditions [22]. Finally, computational complexity and memory requirements are minimal, which may be desirable for resource-constrained devices or time-critical applications.

4.6 Conclusion

This chapter proposes a competition-based update protocol based on Friedkin-Johnsen dynamics to mitigate adversarial attacks disrupting a quadratic distributed optimization task. Performance of the proposed approach and optimal parametrization are characterized by formal results and numerical experiments, and numerical simulations performed on sparse communication graphs show that the former can outperform state-of-the-art resilient consensus techniques. Further, the competition-collaboration trade-off is discussed with analytical arguments that are insightful in understanding the overall behavior of the system and the interaction among regular and malicious agents. Finally, I have addressed design of the communication network and explored how to improve performance with respect to regular graphs and link budget, looking at both global cost of the optimization problem and energy spent by the attacker.

This opens several avenues for future research. Firstly, it is desirable to address an effective design of parameters λ_i's in the realistic case where knowledge about the attack is scarce. This may also involve online reweighing of protocol parameters, in the realm of recent work where the authors that build on the concept of trust [22], [218].

Secondly, the more general and challenging scenario of distributed optimization ought to be extensively studied. In this case, the standard approach is to alternate local descent steps to consensus updates to steer all agents towards a common solution [213]. Here, the additional descent steps may critically impact performance even if consensus steps are made resilient.

A third research avenue involves zero-sum games [100], [102] to alternatively model the system dynamics. In particular, asymmetric zero-sum games let one player have more knowledge than other, which may be a suitable model for worst-case attacks. In this case, a research challenge is determining the optimal game strategies for both players, ultimately to derive effective algorithms in the presence of intelligent adversaries.

Finally, it is interesting to deeply investigate optimization of communication topology. While graph robustness to node or edge failures has been addressed in various domains [27], [159], [176], [203], the novel element given by competitive dynamics calls for studying that from a different perspective, as heuristically motivated in Section 4.4. Also, in the spirit of a game-theoretic approach, a comparison between classical centrality measures and maximum-damage attacker nodes may be drawn to get insights about which agents deserve higher attention.

5 Conclusion

Throughout this thesis, I have addressed and tackled several fundamental performance trade-offs arising in Networked Control Systems as a consequence of intrinsic limitations of single nodes, of communication channels connecting them, or of the very interconnected structure that underlies the dynamical dependencies of the system.

The first chapter addresses computation and communication latency associated with local processing of data collected at the network nodes (referred to as smart sensors), and investigates an optimal estimation problem attempting to optimally select a subset of available sensors and to choose, for each of those, an optimal local processing strategy (quantified by computational delay and accuracy of output data). The main conclusion stemming from analysis and application of heuristic selection algorithms is apparently counterintuitive: in contrast to widespread conventions in sensing design for control, robotics, and Edge Computing applications, it is optimal to both select only a fraction of all available sensors and to let nodes locally process acquired data for a limited amount of time. This is due to computational inefficiency both to locally process measurements at nodes and to gather and post-process all sensory data at a central base station.

The second chapter investigates the challenge of choosing the optimal architecture of a distributed controller. This design problem is indeed NP-hard due to its combinatorial nature, which can prevent to find both an effective design and useful intuition about the optimal design. In particular, I consider the case when communication latency affecting inter-node transmissions in non-negligible: in this case, the nominal dynamics of the system change, as local control loops are closed with outdated feedback information that makes considerably more complicated analysis and synthesis. What is more, and crucially affects the controller design, such communication latency is modeled as an increasing function of the overall number of links, namely, of the controller architecture complexity. Hence, an optimal control design problem must take into account the fact that increasing controller complexity not only enhances information share which is beneficial for feedback,

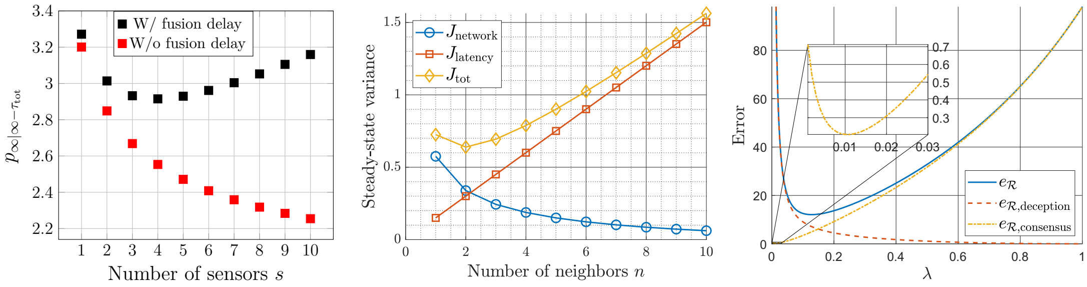


Figure 5.1: Fundamental performance trade-offs investigated throughout this thesis.

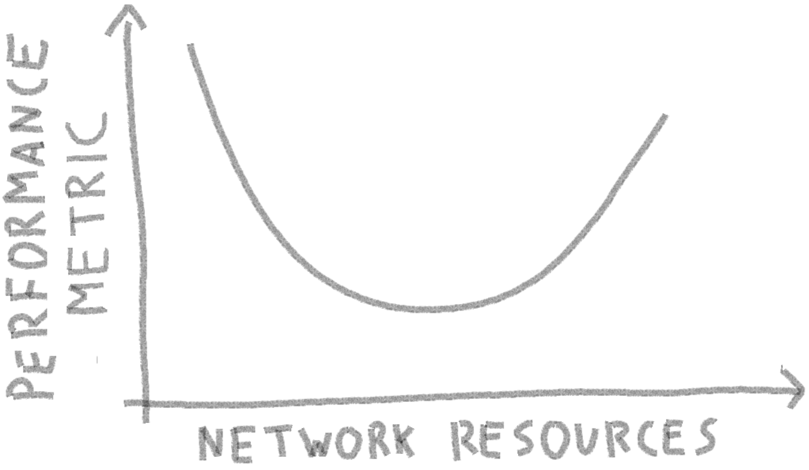
but also makes communication delays longer, weakening the ability of a controller to stabilize the system. To achieve intuition towards the optimal controller design in this scenario, I consider a number of options for the controller architecture, ranging from dense architectures providing long delays to sparse architectures inducing low delays. For each of these, I solve an associated optimal control design problem, which minimizes the steady-state variance of a stochastically forced dynamical system. The fundamental result, which emerges both from an analytical study and from numerical experiments involving several kinds of dynamics, is that the optimal controller architecture is in general distributed, and may even be the sparsest possible if communication delays increase sufficiently fast. This is in stark contrast with the common belief according to which the optimal controller features the centralized architecture (or all-to-all interactions).

Finally, the third chapter consider aspect related to both performance, safety and security, and tackles the presence of agents that can misbehave, namely they do not obey the prescribed update protocol. The most common approach in this case is to force, locally at each node, a filtering of incoming messages sent by neighbors, in order to not deviate too much from the desired behavior. Here, the problem is tackled under a different perspective, that lends itself to a compelling game-theoretic interpretation. Specifically, each agent is allowed to partially trust neighbors, accepting and suitably weighing the information received from them, and to partially retain local information, which serves as an anchor to mitigate the negative effect of interactions with misbehaving agents. The results presented, from both formal and numerical standpoints, corroborate the validity of that approach, showing the presence of a fundamental performance trade-off: the optimal resilient update rule makes each agent choose an specific value of "competitiveness", that varies with the amount and intensity of attacks. This approach, which is here presented with essentially preliminary and analytical results, has the potential to breach a new frontier in the design of resilient control strategies, possibly overcoming fundamental theoretical limitations that arise with standard filtering techniques.

Table 5.1: Potential follow-up research avenues inspired by the proposed results.

	Distributed sensing	**Controller architecture**	**Resilient networks**
Outlook	Realistic latency models Distributed decision-making Adaptive resource allocation	Realistic dynamics and communication delays Complex control tasks Integration with other control paradigms (*e.g.*, SLS)	Competition *vs.* collaboration in other control tasks Network topology design Game-theoretic approaches

Table 5.1 exposes some future work directions that may be pursued to improve and expand the research developed throughout the thesis. Overall, the crucial point that ought to be further explored is the presence of performance trade-offs determined by allocation of network resources: even though in some cases the optimal choice is obvious, the findings presented in this thesis show that indeed in other situations such trade-offs are far from trivial. This suggests that carefully considering non-idealities arising within Networked Control Systems (such as latency or agent misbehavior) is key in order to achieve correct insight about the actual system behavior, which can subsequently play a role in guiding the control design. Ultimately, the results exposed in the previous chapters aim to encourage future research that broadly explores the relationship between available network resources and performance of control tasks towards effective resource allocation and design methods. This means replacing the quantities on the x-axis in Fig. 5.1 with other features of Networked Control Systems – communication bandwidth, hardware and software resources of processing units, mobility, device weight and dimension – and the quantities on the y-axis with related performance metrics – convergence rate, mean-square error, safety, robustness.



Beware of performance trade-offs in Networked Control Systems!

A
Proofs of Chapter 2

A.1 Proof of Theorem 2.2.3

The steady-state error variance for the outdated estimate $\hat{x}_{t-\tau_{tot}}(\tau)$ is the solution of the continuous-time ARE where all sensors are considered:

$$2ap_\infty(\tau) - \sigma_w^2 + \frac{\tau}{\tilde{b}} p_\infty^2(\tau) = 0 \tag{A.1}$$

An open-loop prediction of length τ_{tot} then computes the current-time estimate $\hat{x}_t(\tau)$. The error associated with the prediction has dynamics

$$d\tilde{x}_s(\tau) = a\tilde{x}_s(\tau)ds + dw_s, \qquad t - \tau_{tot} \leq s \leq t \tag{A.2}$$

The error at time t is then given by integrating (A.2) with initial condition $\tilde{x}_{t-\tau_{tot}}(\tau)$:

$$\tilde{x}_t(\tau) = \mathrm{e}^{a\tau_{tot}} \tilde{x}_{t-\tau_{tot}}(\tau) + \bar{w}(\tau_{tot}) \tag{A.3}$$

where $\bar{w}(\tau_{tot})$ is the stochastic integral of w_s in the interval $[t - \tau_{tot},\, t]$. The steady-state error variance is then

$$\begin{aligned} p_{\infty|\infty-\tau_{\text{tot}}}(\tau) &\overset{(i)}{=} \operatorname{var}(\mathrm{e}^{a\tau_{tot}} \tilde{x}_{t-\tau_{tot}}(\tau)) + \operatorname{var}(\bar{w}(\tau_{tot})) = \\ &= \mathrm{e}^{2a\tau_{tot}} p_\infty(\tau) + \frac{\sigma_w^2}{2a}\left(\mathrm{e}^{2a\tau_{tot}} - 1\right) \end{aligned} \tag{A.4}$$

where (i) is motivated by uncorrelated terms. Indeed, $\tilde{x}_{t-\tau_{tot}} \in \operatorname{span}\{x_{t_0}, w_s, v_s : t_0 \leq s \leq t - \tau_{tot}\}$, while $\bar{w}(\tau_{tot}) \in \operatorname{span}\{w_s : t - \tau_{tot} \leq s \leq t\}$, whose intersection has zero measure. The variance $p_{\infty|\infty-\tau_{\text{tot}}}(\tau)$ is quasi-convex with both constant and τ-varying communication and fusion delays. This can be proved, e.g., with a graphical analysis. In virtue of both this fact and limits (2.18), the point of minimum τ_{opt} exists unique and is strictly

positive.
With constant delays $\tau_{\mathrm{c}}(\tau)$, $\tau_{\mathrm{f}}(\tau)$, standard computations show that τ_{opt} must satisfy (2.19).

A.2 Alternative Processing Models

We consider two alternative models to the measurement noise covariance (2.12). These involve a coefficient γ that can be understood as the convergence rate of an anytime algorithm.

Corollary A.2.1 (Non-ideal processing)**.** *Given system* (2.10)*–*(2.11) *and hypotheses as per Theorem 2.2.3 with*

$$\sigma_v^2(\tau) = \frac{b}{\tau^\gamma}, \qquad \gamma > 0, \tag{A.5}$$

the steady-state error variance $p_{\infty|\infty-\tau_{tot}}(\tau)$ has a unique global minimum $\tau_{opt} > 0$.

Proof. It can be seen that limits (2.18) still hold and $p_{\infty|\infty-\tau_{\mathrm{tot}}}(\tau)$ is strictly quasi-convex on $\mathbb{R}_+$ (*e.g.,* via graphical analysis) with both models (2.13a)–(2.13b). □

The second model comes into play with anytime algorithms with exponential convergence, as the ones shown in [161].

Corollary A.2.2 (Exponential-convergence anytime algorithms)**.** *Given system* (2.10)*–*(2.11) *and hypotheses as per Theorem 2.2.3 with*

$$\sigma_v^2(\tau) = be^{-\gamma\tau}, \qquad \gamma > 0, \tag{A.6}$$

the steady-state error variance $p_{\infty|\infty-\tau_{tot}}(\tau)$ has a unique global minimum $\tau_{opt} > 0$:

- *with constant delays as per* (2.13a)*, if and only if*

$$\gamma > 2\sqrt{\frac{\sigma_w^2}{\tilde{b}} + a^2}. \tag{A.7}$$

- *with τ-varying delays as per* (2.13b)*, always.*

Proof. We address the two cases separately.

Constant delays. With model (2.13a), τ_{opt} can be computed in closed form by setting $p'_{\infty|\infty-\tau_{\text{tot}}}(\tau) = 0$. This has the unique solution

$$\tau_{opt} = \frac{1}{\gamma}\left[\ln\left(\frac{\gamma^2}{4} - a^2\right) + \ln\left(\frac{\tilde{b}}{\sigma_w^2}\right)\right], \tag{A.8}$$

which is strictly positive if and only if (A.7) holds.

τ-varying delays. With model (2.13b), $p_{\infty|\infty-\tau_{\text{tot}}}(\tau)$ is quasi-convex (easily verifiable, e.g., via graphical analysis) for any γ.

□

A.3 Proof of Proposition 2.2.6

For convenience, we recall the statement of the implicit function theorem, which is used in the proof.

Theorem A.3.1 (Implicit function)**.** *Let F be a continuously differentiable function on some open $D \subset \mathbb{R}^2$. Assume that there exists a point $(\bar{x}, \bar{y}) \in D$ such that:*

- *$F(\bar{x}, \bar{y}) = 0$;*
- *$\frac{\partial F}{\partial y}(\bar{x}, \bar{y}) \neq 0$.*

Then, there exist two positive constant a, b and a function $f : I_{\bar{x}} := (\bar{x} - a, \bar{x} + a) \mapsto J_{\bar{y}} := (\bar{y} - b, \bar{y} + b)$ such that

$$F(x, y) = 0 \iff y = f(x) \quad \forall x \in I_{\bar{x}},\ \forall y \in J_{\bar{y}}.$$

Moreover, $f \in \mathcal{C}^1(I_{\bar{x}})$ and

$$f'(x) = -\frac{F_x(x, f(x))}{F_y(x, f(x))} \quad \forall x \in I_{\bar{x}}, \tag{A.9}$$

where $F_x(x, f(x)) = \frac{\partial F}{\partial x}(x, f(x))$.

Consider now (2.19), which we rewrite as:

$$\rho\tau_{opt}^3 + a^2\tau_{opt}^2 - \frac{1}{4} = 0. \tag{A.10}$$

We can see the left-hand term in the previous equation as a parametric function of two positive-valued variables, namely

$$F : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R},\ (\pi, \tau) \mapsto F(\pi, \tau) = \rho\tau^3 + a^2\tau^2 - \frac{1}{4}, \tag{A.11}$$

where π, which is either ρ or a^2, is a variable, and the other coefficient is a parameter. Given a solution $(\bar{\pi}, \bar{\tau}_{opt})$ of (A.10), it holds:

- $F(\bar{\pi}, \bar{\tau}_{opt}) = 0$, by construction;
- $F_\tau(\bar{\rho}, \bar{\tau}_{opt}) = 3\bar{\rho}\bar{\tau}_{opt}^2 + 2a^2\bar{\tau}_{opt} > 0$, as $\bar{\rho}, \tau_{opt} > 0$;
- $F_\tau(\bar{a^2}, \bar{\tau}_{opt}) = 3\rho\bar{\tau}_{opt}^2 + 2\bar{a^2}\bar{\tau}_{opt} > 0$, as $\rho, \tau_{opt} > 0$.

Then Theorem A.3.1 applies and there exists a function $\tau(\pi)$ such that $F(\pi, \tau_{opt}) = 0 \iff \tau_{opt} = \tau(\pi)$, with π in some open neighborhood of $\bar{\pi}$. Since we did not pose constraints on $\bar{\pi}$, such a function is defined on the positive real line. We can then compute the first derivative of $\tau(\pi)$ according to (A.9).

$\boldsymbol{\pi = \rho}$ The first derivative of $\tau(\pi) = \tau(\rho)$ is

$$\tau'(\rho) = -\frac{F_\rho(\rho, \tau(\rho))}{F_\tau(\rho, \tau(\rho))} = -\frac{\tau(\rho)^2}{3\rho\tau(\rho) + 2a^2} < 0. \tag{A.12}$$

$\boldsymbol{\pi = a^2}$ The first derivative of $\tau(\pi) = \tau(a^2)$ is

$$\tau'(a^2) = -\frac{F_{a^2}(a^2, \tau(a^2))}{F_\tau(a^2, \tau(a^2))} = -\frac{\tau(a^2)}{3\rho\tau(a^2) + 2a^2} < 0. \tag{A.13}$$

Hence, τ_{opt} is strictly decreasing with both ρ and a^2.

A.4 Sensor Fusion with Kalman Filter in Information Form and Packet Loss

In the following, we drop the dependencies on processing delays for the sake of exposition. According to [167], when the correct reception of a measurement from ith sensor is a binary random variable with success probability λ_i, the optimal steady-state estimator with constant gains has the following dynamics for the expected error covariance:

$$P = APA^\top + Q - APC_\lambda^\top \left(C_\lambda PC_\lambda^\top + P_\lambda + R_\lambda\right)^{-1} C_\lambda PA^\top, \tag{A.14}$$

with

$$\begin{aligned} C_\lambda &= \left[\lambda_1 C_1^\top \dots \lambda_s C_s^\top\right]^\top \\ P_\lambda &= \operatorname{diag}\left(\lambda_1(1-\lambda_1)C_1PC_1^\top, ..., \lambda_s\left(1-\lambda_s\right)C_sPC_s^\top\right) \\ R_\lambda &= \operatorname{diag}\left(\lambda_1 R_1, ..., \lambda_s R_s\right), \end{aligned} \tag{A.15}$$

where $\operatorname{diag}(\cdot)$ denotes a block-diagonal matrix with variables as diagonal blocks, and $\mathcal{S} = \{1, ...s\}$ is the set of sensors involved in the update with measurements. Exploiting the matrix inversion lemma, (A.14) can be rewritten as follows,

$$P = A\left(P^{-1} + \tilde{\Gamma}\right)^{-1} A^\top + Q, \tag{A.16}$$

where the modified information matrix is

$$\tilde{\Gamma} = C_\lambda^\top \left(P_\lambda + R_\lambda\right)^{-1} C_\lambda. \tag{A.17}$$

Finally, we get

$$\begin{aligned} \tilde{\Gamma} &= \sum_{i\in\mathcal{S}} \lambda_i^2 C_i^\top \left(\lambda_i(1-\lambda_i)C_iPC_i^\top + \lambda_i R_i\right)^{-1} C_i \\ &= \sum_{i\in\mathcal{S}} \lambda_i^2 C_i^\top \left[\lambda_i(1-\lambda_i)\left(C_iPC_i^\top + \frac{R_i}{1-\lambda_i}\right)\right]^{-1} C_i \\ &\overset{(i)}{=} \sum_{i\in\mathcal{S}} \lambda_i C_i^\top \left[R_i^{-1} - (1-\lambda_i)R_i^{-1}C_i\left(P^{-1} + \left(1-\lambda_i\right)C_i^\top R_i^{-1}C_i\right)^{-1} C_i^\top R_i^{-1}\right] C_i \\ &= \sum_{i\in\mathcal{S}} \lambda_i \left[C_i^\top R_i^{-1}C_i - \left(1-\lambda_i\right)C_i^\top R_i^{-1}C_i\left(P^{-1} + \left(1-\lambda_i\right)C_i^\top R_i^{-1}C_i\right)^{-1} C_i^\top R_i^{-1}C_i\right] \\ &\overset{(ii)}{=} \sum_{i\in\mathcal{S}} \lambda_i \left[\Gamma_i - \left(1-\lambda_i\right)\Gamma_i\left(P^{-1} + \left(1-\lambda_i\right)\Gamma_i\right)^{-1}\Gamma_i\right], \end{aligned} \tag{A.18}$$

where (i) follows from the matrix inversion lemma, and (ii) from the definition of Γ_i.

A.5 Proof of Theorem 2.3.3

According to [166, Section 3], the estimation starts from the most recent state for which the maximum information possible is available. The former has timestamp $k - \tilde{\tau}_s - \tau_{\mathrm{f,tot}}$, being $\tilde{\tau}_s$ the delay gathered by the most-delayed-sensor data when they are received at the central station. The expected error covariance for such estimate converges to the solution of the ARE (2.31) where all sensors are considered, that is, at steady state the following holds,

$$P_{k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_s|k-1-\tau_{\mathrm{f,tot}}-\tilde{\tau}_s}(\mathcal{T}) = P_{k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_s+1|k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_s}(\mathcal{T}) = P_\infty(\mathcal{T}). \tag{A.19}$$

When computing the state estimates of more recent times, only data from some sensors are available for fusion. In particular, the measurement update for the estimate of the state with delay $\delta + \tau_{\mathrm{f,tot}}$ can only use sensors in $\mathcal{S}(\delta)$,

$$P_{k-\tau_{\mathrm{f,tot}}-\delta|k-\tau_{\mathrm{f,tot}}-\delta}\left(\mathcal{T}\right) = \mathcal{U}\left(P_{k-\tau_{\mathrm{f,tot}}-\delta|k-\tau_{\mathrm{f,tot}}-\delta-1}\left(\mathcal{T}\right), \mathcal{T}_\delta\right). \tag{A.20}$$

According to Assumption 2.3.1, the multi-step KF iteration processing data in the interval $[k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_{i+1} + 2,\ k - \tau_{\mathrm{f,tot}} - \tilde{\tau}_i + 1]$ involves the sensor subset $\mathcal{S}(\tilde{\tau}_i) = \{1, ..., i\}$. The resulting expected error covariance for such iteration is, according to (2.29),

$$P_{k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_i+1|k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_i}(\mathcal{T}) = \mathcal{I}^{\tilde{\tau}_{i+1}-\tilde{\tau}_i}\left(P_{k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_{i+1}+1|k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_{i+1}}\left(\mathcal{T}\right), \mathcal{T}_{\tilde{\tau}_i}\right). \tag{A.21}$$

The multi-step KF iteration involving all the processed dataset (2.1.1) is written as $\mathcal{I}^{\tilde{\tau}_s-\tilde{\tau}_1}\left(P_\infty\left(\mathcal{T}\right), \mathcal{T}_{\tilde{\tau}_s-1}\right)$: starting from $P_\infty\left(\mathcal{T}\right)$, it computes $P_{k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_1+1|k-\tau_{\mathrm{f,tot}}-\tilde{\tau}_1}(\mathcal{T})$ through the multi-step KF iterations (A.21), each involving one sensor less than the previous one. The multi-step prediction $\mathcal{P}^{\tau_{\mathrm{pred}}}(\cdot)$ eventually computes the estimate of the current state, where the remaining delay is $\tau_{\mathrm{pred}} = \tau_{\mathrm{f,tot}} + \tilde{\tau}_1 - 1$.

B

Proofs of Chapter 3

In the following, τ_n is replaced with τ for the sake of readability.

B.1 Proof of Corollary 3.2.2

Convexity of steady-state variance. Because the value of τ does not impact convexity, let $\tau = 1$. The second derivative of $\sigma_I^2(\lambda)$ is

$$\frac{d^2\sigma_I^2(\lambda)}{\lambda} = \frac{(1 + 2\sin\lambda + \lambda\cos\lambda - \cos(2\lambda))\lambda^2\cos^2\lambda}{\lambda^4\cos^4\lambda} + \frac{(\lambda - \cos\lambda + \lambda\sin\lambda - \sin\lambda\cos\lambda)(-2\lambda\cos^2\lambda + 2\lambda^2\cos\lambda\sin\lambda)}{\lambda^4\cos^4\lambda}, \quad \text{(B.1)}$$

which is positive if and only if

$$\lambda^3\cos^3\lambda + 2\lambda\cos^3\lambda + 2\lambda^3\cos\lambda\sin\lambda + 2\lambda\cos^3\lambda\sin\lambda + {} + 2\lambda^3\cos\lambda\sin^2\lambda - 2\lambda^2\cos^2\lambda\sin\lambda - 2\lambda^2\cos^2\lambda > 0. \quad \text{(B.2)}$$

The constraint (3.14) ensures that λ, $\cos\lambda$ and $\sin\lambda$ are positive. I now consider the three possible cases for λ and show that the two negative monomials in (B.2) are always outbalanced by the positive ones.

Case $\lambda > 1$:

$$2\lambda^3\cos\lambda\sin^2\lambda > 2\lambda^2\cos^2\lambda\sin\lambda, \quad \text{(B.3)}$$

$$2\lambda^3\cos\lambda\sin\lambda > 2\lambda^2\cos^2\lambda. \quad \text{(B.4)}$$

Case $\cos\lambda < \lambda \le 1$:

$$2\lambda^3\cos\lambda\sin\lambda > 2\lambda^2\cos^2\lambda\sin\lambda, \tag{B.5}$$

$$2\lambda^3\cos\lambda\sin^2\lambda + 2\lambda\cos^3\lambda \ge 2\lambda^3\cos\lambda > 2\lambda^2\cos^2\lambda. \tag{B.6}$$

Case $\lambda \le \cos\lambda$:

$$2\lambda\cos^3\lambda\sin\lambda \ge 2\lambda^2\cos^2\lambda\sin\lambda, \tag{B.7}$$

$$2\lambda\cos\lambda^3 \ge 2\lambda^2\cos^2\lambda. \tag{B.8}$$

Point of minimum. Uniqueness of λ^* follows from strict convexity. The derivative of the variance $\sigma_I^2(\lambda)$ is

$$\frac{d\sigma_I^2(\lambda)}{d\lambda} = \frac{\tau\lambda - \cos(\tau\lambda) + \tau\lambda\sin(\tau\lambda) - \cos(\tau\lambda)\sin(\tau\lambda)}{2\lambda^2\cos^2(\tau\lambda)}, \tag{B.9}$$

which is equal to zero if and only if

$$(1+\sin(\tau\lambda))(\tau\lambda - \cos(\tau\lambda)) = 0. \tag{B.10}$$

The stability constraint (3.14) makes (B.10) equivalent to $\tau\lambda = \cos(\tau\lambda)$. Applying the change of variable $\beta \leftarrow \tau\lambda$, the resulting equation admits a unique solution β^* in $(0, \pi/2)$.

B.2 Proof of Proposition 3.2.3

The error dynamics equation with agent model (3.17) reads

$$\begin{gathered} dx(t) = (A_0x(t) + A_1x(t-1))\,dt + Bd\bar{w}(t), \\ A_0 = \begin{bmatrix} 0 & I \\ 0 & -\eta I \end{bmatrix}, \; A_1 = \begin{bmatrix} 0 & 0 \\ -\eta K & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ I \end{bmatrix}, \end{gathered} \tag{B.11}$$

with $\bar{w}(t)$ standard N-dimensional Brownian motion. The decoupling (3.19) is obtained from (B.11) through the change of basis $x(t) = (T \otimes I_2)\tilde{x}(t)$. Rewriting (3.19) as a double integrator in state-space form with state $\tilde{s}_j(\cdot)$ yields

$$d\tilde{s}_j(t) = (F_0\tilde{s}_j(t) + F_{1j}\tilde{s}_j(t-1))\,dt + Gd\bar{w}_j(t),$$
$$F_0 = \begin{bmatrix} 0 & 1 \\ 0 & -\eta \end{bmatrix}, \; F_{1j} = \begin{bmatrix} 0 & 0 \\ -\eta\lambda_j & 0 \end{bmatrix}, \; G = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{B.12}$$

Stability of (B.11) is equivalent to that of (B.12) for all j. In the following, the subscript j is dropped for the sake of readability. For positive eigenvalues λ, (B.12) is mean-square asymptotically stable if $\alpha_0 < 0$ and unstable if $\alpha_0 > 0$ [208], where the *spectral abscissa* is defined as

$$\alpha_0 \doteq \sup\left\{\Re(z) : z \in \mathbb{C},\; h(z) = 0\right\}, \tag{B.13}$$

and the *characteristic polynomial* of (B.12) is

$$h(z) \doteq \det\left(zI - F_0 - F_1 e^{-z}\right) = z^2 + \eta z + \eta\lambda e^{-z}. \tag{B.14}$$

A sufficient and necessary condition for all roots of $h(z)$ to lie in the open left-hand half-plane is derived in [21].

Theorem B.2.1 ([21], Theorem 2.1)**.** *Let the 2-vectors $v(b) = \left(pb, q - b^2\right), w(b) = (\cos b, \sin b), b \geq 0$, be given. If $r > 0$, a necessary and sufficient condition for all roots of the equation $h(z) = (z^2 + pz + q)e^z + r = 0$ to have negative real part is that the orthogonality condition $v(b) \cdot w(b) = 0$, with $b \in \cup_{k=0}^{\infty}(2k\pi, (2k+1)\pi)$, implies $|v(b)| > r$.*

From Theorem B.2.1, (B.12) is asymptotically stable if the following implication holds for $b \in \cup_{k=0}^{\infty}(2k\pi, (2k+1)\pi)$,

$$\eta b\cos b - b^2 \sin b = 0 \implies \eta^2 b^2 + b^4 > \eta^2\lambda^2. \tag{B.15}$$

In view of $b \geq 0$ and $\sin b \geq 0$, (B.15) leads to (3.20) after standard algebraic manipulations, where b is replaced with $\beta = \min b \in (0, \pi/2)$. The inequality can be rewritten as

$$\lambda < \frac{\beta}{\sin\beta} \doteq \phi(\eta), \tag{B.16}$$

where the definition of $\phi(\cdot)$ follows from the implicit function theorem applied to $F(\eta,\beta) \doteq \beta\tan\beta - \eta$, which states that $F(\eta,\beta) = 0$ if and only if $\beta = \varphi(\eta)$ and

$$\varphi'(\eta) = \frac{\cos^2(\varphi(\eta))}{\varphi(\eta) + \sin(\varphi(\eta))\cos(\varphi(\eta))}. \tag{B.17}$$

Tedious but straightforward calculations on the first and second derivatives show that

$\phi(\eta)$ is concave increasing for any $\eta > 0$. The limits at 0 and $+\infty$ can be easily computed by noting that

$$\beta_0 \doteq \varphi(0) = 0, \quad \beta_\infty \doteq \lim_{\eta \to +\infty} \varphi(\eta) = \frac{\pi}{2}. \tag{B.18}$$

B.3 Reduced Model of Continuous-Time Double Integrators

Consider (B.12) with state $\tilde{s}(t) = [\tilde{x}(t), \tilde{z}(t)]^\top$. Assuming that the feedback gain η is large, the variable $\tilde{z}(t)$ evolves faster than $\tilde{x}(t)$. Then, by separation of time scales [84], the dynamics of $\tilde{z}(t)$ can be approximated by letting $\tilde{x}(t-1) \equiv x_0$ be constant overtime,

$$d\tilde{z}(t) = (-\eta\tilde{z}(t) - \eta\lambda x_0)\, dt + dw(t). \tag{B.19}$$

Equation B.19 defines a standard Ornstein–Uhlenbeck process,

$$\tilde{z}(t) \sim \mathcal{N}\left(\mathrm{e}^{-\eta t}(\tilde{z}(0) + \lambda x_0) - \lambda x_0, \frac{1}{2\eta}\left(1 - \mathrm{e}^{-2\eta t}\right)\right). \tag{B.20}$$

In view of the time-scale separation, (B.20) holds (with $\tilde{x}(t-1)$ constant) till $\tilde{z}(t)$ settles at steady state,

$$\lim_{t \to +\infty} \tilde{z}(t) = \tilde{z}_\infty \sim \mathcal{N}\left(-\lambda x_0, \frac{1}{2\eta}\right). \tag{B.21}$$

Using (B.21), the dynamics of $\tilde{x}(t)$ can be approximated by assuming that $\tilde{z}(t)$ reaches the steady state instantaneously,

$$d\tilde{x}(t) \approx \tilde{z}_\infty dt = -\lambda\tilde{x}(t-1)dt + dn(t), \tag{B.22}$$

where the diffusion is embedded into the Brownian noise $n(t)$ with variance proportional to $^1/_\eta$. In particular, as $\eta \to +\infty$, $\tilde{z}_\infty \xrightarrow{a.s.} -\lambda x_0$ and (B.22) tends to deterministic dynamics.

B.4 Stability Conditions for Discrete-Time Systems

General Case. The characteristic polynomial $h(z)$ of single-integrator decoupled subsystem (3.27) is obtained by applying the lag operator z such that $\tilde{x}(k)h(z) = \tilde{w}(k)$,

$$h(z) = z - 1 + \lambda z^{-\tau}. \tag{B.23}$$

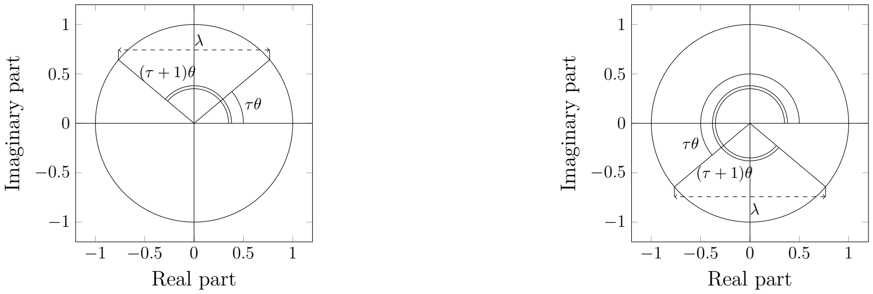


Figure B.1: Two solutions of (B.26) with $\sin(\tau\theta) > 0$ (left) and $\sin(\tau\theta) < 0$ (right).

Similarly, the characteristic polynomial of double-integrator decoupled subsystem (3.30) is

$$h(z) = z - 2 + \eta + (1-\eta)z^{-1} + \eta\lambda z^{-\tau-1}. \tag{B.24}$$

For positive λ, stability of (3.27)–(3.30) can be assessed via the Jury stability criterion, which provides necessary and sufficient conditions for the roots of (B.23) and (B.24) to lie inside the unit circle in the form of inequalities involving the coefficients of $h(z)$. Being the latter polynomial in η and λ, the Jury criterion yields $\Theta(N\tau)$ polynomial inequalities in the feedback gains, which can be computed, *e.g.,* through symbolic software tools.

Proof of Proposition 3.3.1. Equation B.23 can be studied as a root locus by varying the gain λ. In particular, $\lambda = 0$ yields a multiple root at $z_1^* = 0$ and a simple root at $z_2^* = 1$. Negative values of λ are discarded as they push the latter root outside the unit circle. As λ increases, the branches leave the unit ball along their asymptotes. The admissible values for λ are upper bounded by a threshold gain λ_{th} beyond which some roots leave the unit ball. In particular, stability is determined by the minimum gain for which at least one root lies exactly on the unit circle, *i.e.,* $z = \mathrm{e}^{j\theta}$ for some phase θ, and

$$\mathrm{e}^{j(\tau+1)\theta} - \mathrm{e}^{j\tau\theta} + \lambda = 0. \tag{B.25}$$

Equation B.25 can be equivalently written as the system

$$\begin{cases} \cos((\tau+1)\theta) - \cos(\tau\theta) + \lambda = 0 \\ \sin((\tau+1)\theta) = \sin(\tau\theta). \end{cases} \tag{B.26}$$

Fig. B.1 depicts two solutions of system (B.26). being the case $\sin(\tau\theta) < 0$ analogous to the case $\sin(\tau\theta) > 0$, this proof focuses on latter without loss of generality. Further, the solution $(\tau+1)\theta = \tau\theta$ can be discarded from the discussion, because it implies $\lambda = 0$ and thus prevents asymptotic stability. From elementary trigonometric arguments (c.f. Fig. B.1), the second equation in (B.26) implies

$$\tau\theta + \frac{\theta}{2} = \frac{\pi}{2} + 2k\pi \longrightarrow \theta = \frac{\pi + 4k\pi}{2\tau + 1}, \tag{B.27}$$

where I impose $\theta \in [0, \pi]$ and thus $k \in \{0, \dots, \lfloor \tau/2 \rfloor\}$. This includes all possible cases, because the roots of (B.23) appear in complex conjugates pairs. From (B.27), the first equation in (B.26), and the fact $\cos((\tau+1)\theta) = -\cos(\tau\theta)$, it follows

$$\lambda = 2\cos\left(\frac{\pi\tau + 4k\pi\tau}{2\tau + 1}\right). \tag{B.28}$$

The right-hand term in (B.28) is monotone increasing in k. Indeed, taking the argument of the cosine modulus 2π yields

$$\frac{\pi\tau + 4k\pi\tau}{2\tau + 1} \text{ mod } 2\pi = \frac{\pi\tau - 2k\pi}{2\tau + 1} \in \left[0, \frac{\pi}{2}\right), \tag{B.29}$$

which is nonnegative and monotone decreasing in k for any τ. Finally, the upper bound for the gain λ is given by

$$\lambda_{\text{th}} = \min_k 2\cos\left(\frac{\pi\tau + 4k\pi\tau}{2\tau + 1}\right) = 2\cos\left(\frac{\pi\tau}{2\tau + 1}\right). \tag{B.30}$$

B.5 Variance Computation for Discrete-Time Systems

Wiener–Kintchine Formula. Given fixed delay and feedback gains, the steady-state variance $\sigma_I^2(\lambda)$ or $\sigma_{II}^2(\eta, \lambda)$ of the decoupled subsystems can be computed numerically by

$$\frac{1}{2\pi}\int_{-\pi}^{+\pi} \frac{d\theta}{|h(e^{j\theta})|^2}, \tag{B.31}$$

where the characteristic polynomial $h(z)$ is (B.23) or (B.24).

Single Integrator Model. The moment-matching method applied to (3.27) yields a linear system of equations in the variables $(\rho_0, ..., \rho_\tau)$, where $\rho_t \doteq \mathbb{E}[\tilde{x}(k)\tilde{x}(k \pm t)]$:

$$\rho_0 = \mathbb{E}[\tilde{x}(k+1)^2] = \rho_0 + \lambda^2\rho_0 + 1 - 2\lambda\rho_\tau \tag{B.32a}$$

$$\rho_1 = \mathbb{E}[\tilde{x}(k+1)\tilde{x}(k)] = \rho_0 - \lambda\rho_\tau \tag{B.32b}$$

$$\vdots$$

$$\rho_\tau = \rho_{\tau-1} - \lambda\rho_1, \tag{B.32c}$$

where (B.32b)–(B.32c) are the Yule-Walker equations. System (B.32) can be written compactly as $A^{(\tau)}\rho = e_1$, where $\rho^\top = [\rho_0, \ldots, \rho_\tau]$, e_1 is the canonical vector in $\mathbb{R}^{\tau+1}$ with nonzero first coordinate and $A^{(\tau)} \in \mathbb{R}^{(\tau+1)\times(\tau+1)}$ with

$$A^{(\tau)} = \begin{bmatrix} -\lambda^2 & & & & 2\lambda \\ 1 & -1 & & & -\lambda \\ & \ddots & \ddots & \cdot^{\cdot^{\cdot}} & \\ & & & & \\ & -\lambda & & 1 & -1 \end{bmatrix}. \tag{B.33}$$

In particular, when τ is odd, the $(\lceil \tau/2 \rceil + 1)$-th row is

$$\begin{bmatrix} 0 & \ldots & 0 & 1 & -1-\lambda & 0 & \ldots & 0 \end{bmatrix}, \tag{B.34}$$

while, when τ is even, the $(\tau/2 + 2)$-th row is

$$\begin{bmatrix} 0 & \ldots & 0 & 1-\lambda & -1 & 0 & \ldots & 0 \end{bmatrix}. \tag{B.35}$$

Notice that $A^{(\tau)}$ is full rank for all $\tau \geq 1$ and thus (B.32) can be solved uniquely. In particular, the steady-state variance of interest $\sigma_I^2(\lambda)$ coincides with the autocorrelation ρ_0, which is given by the ratio between the minor associated with the top-left element of $A^{(\tau)}$, denoted by $n_\tau \doteq M_{1,1}^{(\tau)}$, and the determinant $d_\tau \doteq \det(A^{(\tau)})$. Hence, ρ_0 is a rational function of λ and can be computed by a symbolic solver given any value of τ.

Further, n_τ and d_τ can be computed by leveraging the following nested structure of the matrix $A^{(\tau)}$:

$$A^{(\tau)} = \begin{bmatrix} -\lambda^2 & & & -2\lambda \\ 1 & -1 & & \\ & 1 & \boxed{\begin{matrix} -1 & & & -\lambda \\ 1 & & & \\ & \boxed{\quad \tilde{A}^{(\tau-4)} \quad} & & \\ -\lambda & & 1 & -1 \end{matrix}} & \\ & -\lambda & & 1 \end{bmatrix}, \tag{B.36}$$

where $\tilde{A}^{(\tau)}$ is the submatrix of $A^{(\tau)}$ obtained by removing its first row and column such that $M_{1,1}^{(\tau)} = \det(\tilde{A}^{(\tau)})$, and the matrices $\tilde{A}^{(\tau-2)}$ and $\tilde{A}^{(\tau-4)}$ are framed in (B.36).

The solution obeys the following recursive expression in τ:

$$n_\tau = \begin{cases} (-1-\lambda)n_{\tau-1} + \tilde{n}_{\tau-1} & \text{if } \tau \text{ odd} \\ -(1-\lambda)n_{\tau-1} - \lambda\tilde{n}_{\tau-1} & \text{if } \tau \text{ even,} \end{cases} \tag{B.37a}$$

$$\tilde{n}_\tau = (2-\lambda^2)\tilde{n}_{\tau-2} - \tilde{n}_{\tau-4}, \tag{B.37b}$$

$$d_\tau = d_{\tau-2} - \lambda^2\left(n_\tau + n_{\tau-2}\right), \tag{B.37c}$$

$$\tilde{n}_{-3} = -1+\lambda^2,\ \tilde{n}_{-2} = \lambda^2,\ \tilde{n}_{-1} = -1,\ \tilde{n}_0 = 0, \tag{B.37d}$$

$$n_{-1} = 0,\ n_0 = 1,\ d_{-1} = -2\lambda,\ d_0 = 2\lambda - \lambda^2. \tag{B.37e}$$

Equation B.37 can be proved by an inductive argument on the delay τ.

Numerator. The formula is proved for odd delays $\tau = 2k+1, k \in \mathbb{N}$. The other case can be obtained similarly and is thus omitted.

Consider the submatrix $\tilde{A}^{(\tau)} \in \mathbb{R}^{\tau\times\tau}$ obtained by removing the first row and column of A, such that $n_\tau = \det(\tilde{A}^{(\tau)})$. Replacing the $(\lfloor \tau/2 \rfloor)$-th column with the sum of $(\lfloor \tau/2 \rfloor)$-th and $(\lceil \tau/2 \rceil)$-th columns yields

$$\det\left(\tilde{A}^{(\tau)}\right) = \left|\begin{array}{c|c|c} \tilde{A}^{(\tau-1)}_{11} & & \tilde{A}^{(\tau-1)}_{12} \\ \hline \ldots\ \ 0\ \ -\lambda & -1-\lambda & \\ \hline & 1 & \\ \tilde{A}^{(\tau-1)}_{21} & 0 & \tilde{A}^{(\tau-1)}_{22} \\ & \vdots & \end{array}\right|, \tag{B.38}$$

from which it follows $n_\tau = (-1-\lambda)n_{\tau-1} - \det(R^{(\tau)})$ where $R^{(\tau)} \in \mathbb{R}^{(\tau-1)\times(\tau-1)}$ and the base case is $n_1 = -1-\lambda$. This expression corresponds to (B.37a) with $\tilde{n}_{\tau-1} = -\det(R^{(\tau)})$. Manipulations of the second term yield a further recursive expression for $\tilde{n}_{\tau-1}$. Consider

$$\det\left(R^{(\tau)}\right) = \begin{vmatrix} -1 & & & -\lambda \\ 1 & \boxed{\begin{matrix} -1 & & & -\lambda \\ 1 & \boxed{\begin{matrix} & & \\ & R^{(\tau-4)} & \\ & & \end{matrix}} & & \\ \lambda & & 1 & -1 \end{matrix}} & & \\ -\lambda & & 1 & -1 \end{vmatrix}, \tag{B.39}$$

where the two inner boxes highlight $R^{(\tau-2)}$ and $R^{(\tau-4)}$, respectively. Straightforward calculations yield

$$\det\left(R^{(\tau)}\right) = \det\left(R^{(\tau-2)}\right) + \lambda \begin{vmatrix} 1 & \boxed{\begin{matrix} -1 & & & -\lambda \\ 1 & \boxed{\begin{matrix} & & \\ & R^{(\tau-4)} & \\ & & \end{matrix}} & & \\ -\lambda & & 1 & -1 \end{matrix}} \\ -\lambda & 1 \end{vmatrix}. \tag{B.40}$$

The determinant in the second addend is computed as

$$-\lambda \det\left(R^{(\tau-2)}\right) + \begin{vmatrix} 1 & \boxed{\begin{matrix} & & \\ & R^{(\tau-4)} & \\ & & \end{matrix}} \\ -\lambda & 1 \end{vmatrix}, \tag{B.41}$$

and the second addend in the above equation has the same structure as the determinant in the second addend in (B.40). Thus, an easy inductive argument proves

$$\begin{aligned} \det\left(R^{(\tau)}\right) = \det\left(R^{(\tau-2)}\right) + \lambda \Big(-\lambda \det\left(R^{(\tau-2)}\right) \\ -\lambda \det\left(R^{(\tau-4)}\right) - \cdots - \lambda \det\left(R^{(3)}\right) - \lambda\Big), \end{aligned} \tag{B.42}$$

where the base case is $\det\left(R^{(3)}\right) = -\lambda^2$. Equation B.37b is retrieved by noting

$$\det\left(R^{(\tau-2)}\right) - \left(1-\lambda^2\right) \det\left(R^{(\tau-4)}\right) = \lambda\left(-\lambda \det\left(R^{(\tau-6)}\right) - \cdots - \lambda\right), \tag{B.43}$$

and thus the tail of the infinite summation in (B.42) can be replaced by the left-hand term in (B.43).

Denominator. The denominator of ρ_0 is computed as the determinant of A. Letting $A^{(\tau)} \doteq A$, from (B.33) it follows

$$\det\left(A^{(\tau)}\right) = -\lambda^2 M_{1,1}^{(\tau)} - 2\lambda \begin{vmatrix} 1 & -1 & & \\ & 1 & \boxed{\begin{matrix} -1 & & & -\lambda \\ 1 & & & \\ & \boxed{\quad \tilde{A}^{(\tau-4)} \quad} & & \\ -\lambda & & 1 & -1 \end{matrix}} & \\ & -\lambda & & 1 \end{vmatrix}, \tag{B.44}$$

where $\tilde{A}^{(\tau-2)}$ and $\tilde{A}^{(\tau-4)}$ are framed in the second addend above. The latter can be computed as the following sum,

$$\lambda M_{1,1}^{(\tau-2)} + \begin{vmatrix} 1 & -1 & & \\ & 1 & \boxed{\quad \tilde{A}^{(\tau-4)} \quad} & \\ & -\lambda & & 1 \end{vmatrix}, \tag{B.45}$$

where the same structure is repeated recursively in the second addend above. Thus, an easy inductive argument proves

$$d_\tau = -\lambda^2 n_\tau - 2\lambda\left(\lambda n_{\tau-2} + \lambda n_{\tau-4} + \cdots + \lambda n_1 + 1\right), \tag{B.46}$$

where the base case is $d_1 = -\lambda^2(-1-\lambda) - 2\lambda$. Equation B.37c is retrieved by noting

$$\begin{aligned} -2\lambda\left(\lambda n_{\tau-2} + \lambda n_{\tau-4} + \cdots + 1\right) &= -\lambda^2 n_{\tau-2} - \lambda^2 n_{\tau-2} - 2\lambda\left(\lambda n_{\tau-4} + \cdots + 1\right) \\ &= -\lambda^2 n_{\tau-2} + d_{\tau-2}. \end{aligned} \tag{B.47}$$

Given τ, convexity of ρ_0 in λ can be assessed by checking the sign of the second derivative in the stability region. This reduces to a system of inequalities which can be solved, *e.g.*, by `solve_rational_inequalities` in Python. The variance was proved strictly convex for all tried delays.

Double Integrator Model. The moment-matching system associated with (3.30) has

$\tau + 2$ variables $(\rho_0, \dots, \rho_{\tau+1})$ and is composed of the following equations:

$$\begin{aligned}\rho_0 &= (2-\eta)^2\rho_0 + (1-\eta)^2\rho_0 + \eta^2\lambda^2\rho_0 + 1 - 2(2-\eta)(1-\eta)\rho_1 \\ &\quad - 2(2-\eta)\eta\lambda\rho_{\tau+1} + 2(1-\eta)\eta\lambda\rho_\tau \end{aligned} \tag{B.48a}$$

$$\rho_1 = (2-\eta)\rho_0 - (1-\eta)\rho_1 - \eta\lambda\rho_{\tau+1} \tag{B.48b}$$

$$\rho_2 = (2-\eta)\rho_1 - (1-\eta)\rho_0 - \eta\lambda\rho_\tau \tag{B.48c}$$

$$\vdots$$

$$\rho_{\tau+1} = (2-\eta)\rho_\tau - (1-\eta)\rho_{\tau-1} - \eta\lambda\rho_1, \tag{B.48d}$$

where (B.48b)–(B.48d) are the Yule-Walker equations associated with (3.30). Analogous considerations to the single-integrator model can be done in this case.

B.6 Proof of Proposition 3.4.1

Because the eigenvalues of a circulant matrix are the Discrete Fourier Transform (DFT) of its first row, from linearity of the DFT and Plancherel theorem, problem (3.33) can be rewritten as follows,

$$\begin{aligned} \tilde{k}^* = \operatorname*{arg\,min}_{k\in\mathbb{R}^n} \quad & \|r(k) - \lambda^* e_1\|_2^2 \\ \text{s.t.} \quad & \lambda_M(k) < \frac{\pi}{2\tau_n}, \end{aligned} \tag{B.49}$$

where $r(k)^\top$ is the first row of K and e_ℓ is the ℓth canonical vector in $\mathbb{R}^N$. Given a set of integers $\mathcal{I} \subset \{1, \dots, N\}$, consider now the problem

$$\begin{aligned} x^* = \operatorname*{arg\,min}_{x\in\mathbb{R}^N} \quad & \|x - e_\ell\|_2^2 \\ \text{s.t.} \quad & x_i = 0 \quad \forall i \in \mathcal{I} \\ & \sum_{i\notin\mathcal{I}} x_i = 0 \end{aligned} \tag{B.50}$$

with $\ell \notin \mathcal{I}$ and $|\mathcal{I}| = N - n - 1$. I first show that $x_i^* \equiv \bar{x}^* \, \forall i \notin \mathcal{I} \cup \{\ell\}$, which implies that the suboptimal gains $\tilde{k}_i^*, i = 1, \dots, n$, in $r(\tilde{k}^*)$ are equal. Assume there exists $j \notin \mathcal{I} \cup \{\ell\}$ such that $x_j \neq x_i \equiv \bar{x}$ for all $i \notin \mathcal{I} \cup \{j, \ell\}$. The cost of x is

$$C_x = \|x - e_\ell\|_2^2 = (x_\ell - 1)^2 + (n-1)\bar{x}^2 + x_j^2. \tag{B.51}$$

Let $\tilde{x}$ such that $\tilde{x}_\ell = x_\ell$, $\tilde{x}_i \equiv \bar{\tilde{x}}$ for all $i \notin \mathcal{I} \cup \{\ell\}$. We have

$$\begin{aligned} x_\ell &= -x_j - \sum_{i \neq \ell, j} x_i = -x_j - (n-1)\bar{x} \\ \tilde{x}_\ell &= -\sum_{i \neq \ell} \tilde{x}_i = -n\bar{\tilde{x}}, \end{aligned} \tag{B.52}$$

and the cost associated with $\tilde{x}$ is

$$C_{\tilde{x}} = (\tilde{x}_\ell - 1)^2 + n\bar{\tilde{x}}^2 = (x_\ell - 1)^2 + \frac{[(n-1)\bar{x} + x_j]^2}{n}. \tag{B.53}$$

We then have the following chain of inequalities,

$$\begin{aligned} [(n-1)\bar{x} + x_j]^2 &< n[(n-1)\bar{x}^2 + x_j^2] \\ 2(n-1)\bar{x}x_j &< (n-1)\left(\bar{x}^2 + x_j^2\right) \\ 0 &< (\bar{x} - x_j)^2, \end{aligned} \tag{B.54}$$

which implies $C_{\tilde{x}} < C_x$ for any $x, \tilde{x}$.

Once proved that all suboptimal feedback gains have to be equal, the suboptimal value $\tilde{k}^*$ can be computed by considering a simplified version of (3.33). According to (3.37) and setting all feedback gains $k_i \equiv \tilde{k}$, each eigenvalue λ_j of matrix K can be written as $\lambda_j = g_j\tilde{k}$ for a constant g_j. The cost function in (3.33) can then be rewritten as

$$J(\tilde{k}) = \sum_{j=2}^{N} \left(g_j\tilde{k} - \lambda^*\right)^2. \tag{B.55}$$

Convexity and differentiability of (B.55) allow to find the global minimum by setting

$$\frac{dJ(\tilde{k})}{d\tilde{k}} = 0 \quad \longrightarrow \quad \sum_{j=2}^{N} g_j \left(g_j\tilde{k} - \lambda^*\right) = 0, \tag{B.56}$$

which admits the unique solution

$$\tilde{k}^* = \frac{\sum_{j=2}^{N} g_j}{\sum_{j=2}^{N} g_j^2}\,\lambda^*. \tag{B.57}$$

The coefficients g_j's are the eigenvalues of K when $\tilde{k} = 1$. Hence, it holds

$$\sum_{i=2}^{N} g_i = \sum_{i=1}^{N} g_i = N \sum_{i=1}^{N} r_i(k) = 2Nn \tag{B.58}$$

$$\sum_{i=2}^{N} g_i^2 = \sum_{i=1}^{N} g_i^2 = N \sum_{i=1}^{N} r_i^2(k) = N(4n^2 + 2n), \tag{B.59}$$

where (B.58) comes from the definition of inverse DFT and (B.59) from Plancherel theorem. The final expression of $\tilde{k}^*$ follows by substituting (B.58)–(B.59) in (B.57).

It is left to check if such solution satisfies the stability constraint. First, note that $\lambda^* < \pi/4\tau_n$, which follows by studying the sign of (B.9). We then have the following relations for the largest eigenvalue,

$$\begin{aligned} \tilde{\lambda}_M^* &= g_M \tilde{k}^* = 2\tilde{k}^* \left(n - \sum_{\ell=1}^{n} \cos\left(\frac{2\pi(M-1)\ell}{N} \right) \right) \\ &= \frac{2\lambda^*}{2n+1} \left(n - \sum_{\ell=1}^{n} \cos\left(\frac{2\pi(M-1)\ell}{N} \right) \right) < \frac{\pi}{4\tau_n} \frac{4n}{2n+1} < \frac{\pi}{2\tau_n}. \end{aligned} \tag{B.60}$$

C
Proofs of Chapter 4

C.1 Useful Lemmas

I report next standard facts in linear algebra that are used in the following proofs.

Lemma C.1.1. *Let $\alpha \in \mathbb{R}$ and $A, B \in \mathbb{R}^{n\times n}$ differentiable functions of α, then the derivative of* $\mathrm{Tr}\left(A^\top B\right)$ *is*

$$\frac{d\mathrm{Tr}\left(A^\top B\right)}{d\alpha} = \mathrm{Tr}\left(\frac{dA^\top}{d\alpha}B\right) + \mathrm{Tr}\left(A^\top \frac{dB}{d\alpha}\right). \tag{C.1}$$

Lemma C.1.2. *Let $\alpha \in \mathbb{R}$ and $A \in \mathbb{R}^{n\times n}$ invertible and differentiable function of α, then the derivative of A^{-1} is*

$$\frac{dA^{-1}}{d\alpha} = -A^{-1}\frac{dA}{d\alpha}A^{-1}. \tag{C.2}$$

Lemma C.1.3. *Let $A \in \mathbb{R}^{n\times n}$ invertible with eigenvalue-eigenvector couple (λ, v), then A^{-1} has eigenvalue-eigenvector couple (λ^{-1}, v).*

Corollary C.1.4. *If $A \in \mathbb{R}^{n\times n}$ is diagonalizable, then A and A^{-1} are simultaneously diagonalizable.*

Lemma C.1.5. *Let $A \in \mathbb{R}^{n\times n}$ with eigenvalue-eigenvector couple (λ, v), then $(I - \alpha A)$ has eigenvalue-eigenvector couple $((1-\alpha\lambda), v)$.*

C.2 Proof of Proposition 4.2.3

I first compute the consensus error induced by FJ dynamics with $\lambda = 0$. By virtue of Assumption 4.1.3, the steady-state consensus value induced by W is the average of the priors of malicious agents, *i.e.*, $\bar{\tilde{\theta}}_{\mathcal{M}} \doteq \frac{1}{M}\sum_{m\in\mathcal{M}}\tilde{\theta}_m$, $M \doteq |\mathcal{M}|$. The consensus error $e_{\mathcal{R}}$ is

$$\begin{aligned}
e^C &= \mathbb{E}\left[\left\|\mathbb{1}_R\frac{1_{\mathcal{M}}^\top}{M}\tilde{\theta} - \mathbb{1}_R\frac{1_{\mathcal{R}}^\top}{R}\tilde{\theta}\right\|^2\right]\\
&= \mathbb{E}\left[\left\|\mathbb{1}_R\frac{1_{\mathcal{M}}^\top}{M}\tilde{\theta}\right\|^2\right] + \mathbb{E}\left[\left\|\mathbb{1}_R\frac{1_{\mathcal{R}}^\top}{R}\tilde{\theta}\right\|^2\right] - 2\mathbb{E}\left[\tilde{\theta}^\top\frac{1_{\mathcal{R}}}{R}\mathbb{1}_R^\top\mathbb{1}_R\frac{1_{\mathcal{M}}^\top}{M}\tilde{\theta}\right]\\
&= \frac{R\mathrm{Tr}\left(\tilde{\Sigma}1_{\mathcal{M}}1_{\mathcal{M}}^\top\right)}{M^2} + \frac{\mathrm{Tr}\left(\tilde{\Sigma}1_{\mathcal{R}}1_{\mathcal{R}}^\top\right)}{R} - \frac{2\mathrm{Tr}\left(\tilde{\Sigma}1_{\mathcal{R}}1_{\mathcal{M}}^\top\right)}{M}\\
&= \frac{R}{M^2}\sum_{m\in\mathcal{M}} d_m + \frac{R}{M^2}\sum_{m\in\mathcal{M}}\left(\sigma_m^2 + \sum_{\substack{n\in\mathcal{M}\\ n\neq m}}\sigma_{mn}\right)\\
&\quad + \frac{1}{R}\sum_{i\in\mathcal{R}}\left(\sigma_i^2 + \sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}\right) - \frac{2}{M}\sum_{i\in\mathcal{R}}\sum_{m\in\mathcal{M}}\sigma_{im}, \qquad \text{(C.3)}
\end{aligned}$$

where $1_{\mathcal{M}} \in \mathbb{R}^N$ and $1_{\mathcal{R}} \in \mathbb{R}^N$ are the indicator vectors of sets $\mathcal{M}$ and $\mathcal{R}$, respectively. On the other hand, the FJ dynamics with $\lambda = 1$ simply freezes all regular agents' priors, yielding consensus error

$$\begin{aligned}
e_1^{FJ} &= \mathbb{E}\left[\|\theta_{\mathcal{R}} - C_R\theta_{\mathcal{R}}\|^2\right] = \mathbb{E}\left[\|(I_R - C_R)\theta_{\mathcal{R}}\|^2\right]\\
&= \mathrm{Tr}\left(\mathbb{E}\left[\theta_{\mathcal{R}}\theta_{\mathcal{R}}^\top\right](I - C_R)\right) = \frac{R-1}{R}\sum_{i\in\mathcal{R}}\sigma_i^2 - \frac{1}{R}\sum_{i\in\mathcal{R}}\sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}, \qquad \text{(C.4)}
\end{aligned}$$

which depends only on the nominal covariance matrix of priors Σ. By comparing the final expressions in (C.3) and (C.4), it follows that $e^C > e_1^{FJ}$ is equivalent to the following inequality,

$$\begin{aligned}
\sum_{m\in\mathcal{M}} d_m > &-\sum_{m\in\mathcal{M}}\left(\sigma_m^2 + \sum_{\substack{n\in\mathcal{M}\\ n\neq m}}\sigma_{mn}\right) + \frac{M^2}{R}\sum_{i\in\mathcal{R}}\sigma_i^2\\
&- \frac{2M^2}{R^2}\sum_{i\in\mathcal{R}}\left(\sigma_i^2 + \sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}\right) + \frac{2M}{R}\sum_{i\in\mathcal{R}}\sum_{m\in\mathcal{M}}\sigma_{im}, \qquad \text{(C.5)}
\end{aligned}$$

which leads to condition (4.9).

C.3 Proof of Theorem 4.2.5

Part one: $\lambda^* < 1$. Let matrix $S_R \in \mathbb{R}^{R\times N}$ maps x to $x_{\mathcal{R}}$, and matrix $C_R \doteq \frac{1}{R}\mathbb{1}_R\mathbb{1}_R^\top$ the consensus matrix for the regular agent subset. According to the labeling discussed in Section 4.1.1, it holds $S_R = [I_R|0]$. Then, the error (4.4) can be written as

$$e_{\mathcal{R}} = \operatorname{Tr}\left(\tilde{\Sigma}E^\top E\right), \quad E \doteq S_R L - C_R S_R, \tag{C.6}$$

and its derivative with respect to λ is (up to constants)

$$\frac{de_{\mathcal{R}}}{d\lambda} = \frac{1}{\lambda}\operatorname{Tr}\left(\tilde{\Sigma}L^\top\left(I - W^\top L^\top\right)S_R^\top E\right), \tag{C.7}$$

where Lemmas C.1.1–C.1.2 were used. At $\lambda = 1$, (C.7) takes value

$$\left.\frac{de_{\mathcal{R}}}{d\lambda}\right|_{\lambda=1} = \operatorname{Tr}\left(\tilde{\Sigma}\left(I - W^\top\right)S_R^\top\left(S_R - C_R S_R\right)\right). \tag{C.8}$$

Straightforward computations show that the argument of the trace in (C.8) takes form

$$\tilde{\Sigma}\left(I - W^\top\right)S_R^\top\left(S_R - C_R S_R\right) = \left[\begin{array}{c|c} A & 0 \\ \hline \star & 0 \end{array}\right], \qquad A \in \mathbb{R}^{R\times R}, \tag{C.9}$$

and the ith diagonal element of A, associated with $i \in \mathcal{R}$, is

$$\begin{aligned} a_i = \sigma_i^2 + \frac{1}{R}\sum_{m\in\mathcal{M}}\sigma_{im}\left(1 - \sum_{\substack{m'\in\mathcal{M}\\ m'\neq m}} W^o_{m'm}\right) - \frac{1}{R}\sigma_i^2\sum_{m\in\mathcal{M}}W^o_{mi} \\ - \sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}W_{ij} - \frac{1}{R}\sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}\sum_{m\in\mathcal{M}}W^o_{mj} - \sum_{m\in\mathcal{M}}\sigma_{im}W^o_{mi}, \end{aligned} \tag{C.10}$$

where W^o_{ij} is the weight of the directed edge from j to i in the original matrix W^o (without malicious agents), and $W_{mj} = \delta_{mj}$ according to Assumption 4.1.3, δ_{mj} being the Kronecker delta. Note that $W^o_{ij} = W_{ij}$ for $i, j \in \mathcal{R}$. Being W^o doubly stochastic, it holds

$$\begin{aligned} \frac{1}{R}\sigma_i^2\sum_{m\in\mathcal{M}}W^o_{mi} + \sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}W_{ij} + \frac{1}{R}\sum_{\substack{j\in\mathcal{R}\\ j\neq i}}\sigma_{ij}\sum_{m\in\mathcal{M}}W^o_{mj} + \sum_{m\in\mathcal{M}}\sigma_{im}W^o_{mi} \\ \leq \max\left\{\frac{1}{R}\sigma_i^2 + \sigma_{im^*}, \sigma_{ij^*}\right\}, \end{aligned} \tag{C.11}$$

where $j^* \doteq \arg\max_{j \in \mathcal{R} \setminus \{i\}} \sigma_{ij}$ and $m^* \doteq \arg\max_{m \in \mathcal{M}} \sigma_{im}$.

Case $\boldsymbol{\frac{1}{R}\sigma_i^2 + \sigma_{im^*}^2 \geq \sigma_{ij^*}^2}$: it follows

$$a_i \geq \sigma_i^2 + \frac{1}{R}\sigma_{im^*} - \frac{1}{R}\sigma_i^2 - \sigma_{im^*} = \left(\sigma_i^2 - \sigma_{im^*}\right)\left(1 - \frac{1}{R}\right) \geq 0. \tag{C.12}$$

Case $\boldsymbol{\frac{1}{R}\sigma_i^2 + \sigma_{im^*}^2 < \sigma_{ij^*}^2}$: it follows

$$a_i \geq \sigma_i^2 - \sigma_{ij^*} + \frac{1}{R}\sum_{m \in \mathcal{M}} \sigma_{im}\left(1 - \sum_{\substack{m' \in \mathcal{M} \\ m' \neq m}} W^o_{m'm}\right) \geq 0. \tag{C.13}$$

Being $a_i \geq 0 \, \forall i \in \mathcal{R}$, and because $\exists i \in \mathcal{R} : a_i > 0$ according to the hypothesis, the error derivative at $\lambda = 1$ (C.8) is strictly positive, hence the consensus error (4.4) is increasing in a left neighborhood of 1. By virtue of continuity of (C.7) for $\lambda > 0$, the point of minimum of $e_{\mathcal{R}}$ satisfies $\lambda^* < 1$.

Part two: $\boldsymbol{\lambda^* > 0}$. By virtue of continuity of the error derivative in L, we can compute the limit of (C.7) as

$$\begin{aligned}
\lim_{\lambda \to 0^+} \frac{de_{\mathcal{R}}}{d\lambda} &= \mathrm{Tr}\left(\widetilde{\Sigma} \lim_{\lambda \to 0^+} \frac{dL}{d\lambda}^\top S_R^\top \lim_{\lambda \to 0^+} E\right) = \mathrm{Tr}\left(\widetilde{\Sigma}\Gamma^\top S_R^\top \left(S_R \overline{W} - C_R S_R\right)\right) \\
&= \mathrm{Tr}\left(\widetilde{\Sigma}\Gamma^\top S_R^\top \left[-C_R \,|\, C_{RM}\right]\right) \\
&= \mathrm{Tr}\left(-\widetilde{\Sigma}_{11}\Gamma_1^\top C_R - \widetilde{\Sigma}_{12}\Gamma_2^\top C_R + \widetilde{\Sigma}_{12}^\top \Gamma_1^\top C_{RM} + \widetilde{\Sigma}_{22}\Gamma_2^\top C_{RM}\right),
\end{aligned} \tag{C.14}$$

where the steady-state consensus matrix $\overline{W} \doteq \lim_{\lambda \to 0^+} W$ has block partition (cf. Assumption 4.1.3)

$$\overline{W} = \left[\begin{array}{c|c} 0 & C_{RM} \\ 0 & I_M \end{array}\right], \tag{C.15}$$

where $\Gamma_1, \widetilde{\Sigma}_{11} \in \mathbb{R}^{R \times R}$ and $\Gamma_2, \widetilde{\Sigma}_{22} \in \mathbb{R}^{M \times M}$. Matrix Γ can be computed from the spectral decomposition of W. In particular, its elements are finite, Γ_1 is nonnegative, and Γ_2 is nonpositive (see details in Appendix C.4). Hence, limit (C.14) is negative if and only if the following inequality holds,

$$\mathrm{Tr}\left(V_{\mathcal{M}}(-\Gamma_2^\top C_{RM})\right) > \mathrm{Tr}\left(-\Sigma_{11}\Gamma_1^\top C_R - \Sigma_{12}\Gamma_2^\top C_R + \Sigma_{12}^\top \Gamma_1^\top C_{RM} + \Sigma_{22}\Gamma_2^\top C_{RM}\right), \tag{C.16}$$

which leads to condition (4.14). It follows that, if (C.16) holds, $e_{\mathcal{R}}$ is strictly decreasing in a right neighborhood of $\lambda = 0$. By virtue of continuity, it follows that $\lambda^* > 0$.

C.4 Computation of Matrix Γ

For the sake of simplicity, in the following I assume that the original weight matrix W^o (*i.e.,* with weights not corrupted by malicious agents) is symmetric, which implies that W is diagonalizable also after malicious agents modify their weights according to Assumption 4.1.3. If W is not diagonalizable, a similar derivation can be carried out by considering the Jordan canonical form of W. This is because a straightforward extension of Lemma C.1.3 shows that W and Γ share the same (chain of) generalized eigenvectors.

Computation of Γ. The derivative of L is (Lemma C.1.2)

$$\frac{dL}{d\lambda} = \tilde{L} - \lambda\tilde{L}\frac{d\tilde{L}^{-1}}{d\lambda}\tilde{L} = \tilde{L} - \lambda\tilde{L}W\tilde{L}, \tag{C.17}$$

where $\tilde{L} \doteq (I - (1-\lambda)W)^{-1}$. Let λ_W and v_W an eigenvalue of W and its associated eigenvector, respectively, from Lemmas C.1.3–C.1.5 it follows that $\tilde{L}$ has eigenvalue $(1-(1-\lambda)\lambda_W)^{-1}$ with associated eigenvector v_W. Hence, straightforward computations yield

$$\frac{dL}{d\lambda}v_W = \frac{1-(1-(1-\lambda)\lambda_W)^{-1}\lambda\lambda_W}{(1-(1-\lambda)\lambda_W)}v_W. \tag{C.18}$$

In particular, the dominant eigenvector $v_W = \mathbb{1}$ (associated with $\lambda_W = 1$) is in the kernel of ${}^{dL}\!/_{d\lambda}$ for any λ. As for the other eigenvectors, by letting λ go to zero in (C.18), one gets

$$\Gamma v_W = (1-\lambda_W)^{-1} v_W. \tag{C.19}$$

Finally, the eigendecomposition of Γ is obtained from eigenvectors v_W and eigenvalues $(1-\lambda_W)^{-1}$, plus the kernel.

Sign of Γ_1 and Γ_2. As regards Γ_1, note that the upper-left block in $\overline{W}$ is identically zero, and that L is a stochastic matrix for any value of λ: hence, as λ becomes larger than zero, (some) elements in L_1 become positive, and thus their derivative at $\lambda = 0^+$ is also positive.

As for Γ_2, define the following block partitions,

$$L = \left[\begin{array}{c|c} L_1 & L_2 \\ \hline 0 & I_M \end{array}\right], \tag{C.20}$$

with $W_1, L_1 \in \mathbb{R}^{R\times R}$ and $W_2, L_2 \in \mathbb{R}^{R\times M}$. Then, it holds

$$\frac{dL}{d\lambda} = \frac{1}{\lambda} L\,(I - WL) = \left[\begin{array}{c|c} \star & -L_1W_1L_2 - L_1 \\ \hline 0 & 0 \end{array}\right], \tag{C.21}$$

which implies, for any $\lambda \in (0, 1)$,

$$\frac{dL_{im}}{d\lambda} \leq 0,\ i \in \mathcal{R}, m \in \mathcal{M}. \tag{C.22}$$

In particular, the limit of the derivative of element L_{im} at $\lambda = 0^+$ is nonpositive by virtue of the theorem of sign permanence.

C.5 Proof of Proposition 4.2.10

Computing the partial derivative of $e_{\mathcal{R}}(d_1, \ldots, d_M)$ (C.6) yields

$$\frac{\partial e_{\mathcal{R}}(d_1, \ldots, d_M)}{\partial d_m} = \mathrm{Tr}\left(\left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & S_m \end{array}\right] E^\top E\right), \tag{C.23}$$

where

$$\frac{\partial \widetilde{\Sigma}(d_1, \ldots, d_M)}{\partial d_m} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & S_m \end{array}\right] \tag{C.24}$$

and $S_m \in \mathbb{R}^{M\times M}$ has all zero elements except for the mth diagonal element equal to 1. Hence, the argument of the trace in (C.23) has all zero rows except for the $(R+m)$th row, which equals the $(R+m)$th row of

$$\left[\begin{array}{c|c} (L_1 - C_R)^2 & (L_1 - C_R)L_2 \\ \hline L_2^\top(L_1 - C_R) & L_2^\top L_2 \end{array}\right]. \tag{C.25}$$

The trace then selects the mth diagonal element of $L_2^\top L_2$, which has all positive elements (see [144] and discussion in Section 4.3.1). Hence, it follows that the partial derivative in (C.23) is strictly positive for any $m \in \mathcal{M}$.

C.6 Proof of Proposition 4.2.11

The partial derivative of the error with respect to λ and to d_m is

$$\frac{\partial^2 e_{\mathcal{R}}(\lambda, d_1, \ldots, d_M)}{\partial d_m \partial \lambda} = \frac{1}{\lambda} \mathrm{Tr}\left(L \frac{\partial \widetilde{\Sigma}(d_1, \ldots, d_M)}{\partial d_m} L^\top \left(I - W^\top L^\top \right) S_R^\top S_R \right). \tag{C.26}$$

It holds

$$L \frac{\partial \widetilde{\Sigma}(d_1, \ldots, d_M)}{\partial d_m} = \left[\begin{array}{c|c} 0 & L_2 S_m \\ \hline 0 & S_m \end{array} \right] \tag{C.27}$$

$$M \doteq I - W^\top L^\top = \left[\begin{array}{c|c} I_R - W_1^\top L_1^\top & 0 \\ \hline -W_2^\top L_1^\top - L_2^\top & 0 \end{array} \right] \tag{C.28}$$

$$L^\top M S_R^\top S_R = \left[\begin{array}{c|c} \star & 0 \\ \hline -L_2^\top W_1^\top L_1^\top - W_2^\top L_1^\top & 0 \end{array} \right], \tag{C.29}$$

and the argument of the trace in (C.26) is

$$\left[\begin{array}{c|c} -L_2 S_m L_2^\top W_1^\top L_1^\top - L_2 S_m W_2^\top L_1^\top & 0 \\ \hline \star & 0 \end{array} \right], \tag{C.30}$$

whose upper-left block is a negative matrix for all $\lambda \in (0, 1)$, and is the zero matrix for $\lambda = 1$. Hence, the error derivative with respect to λ (C.7) is strictly decreasing with d_m for any $\lambda \in (0, 1)$, and does not depend on d_m at $\lambda = 1$. By virtue of continuity of (C.7) in λ, the critical points of $e_{\mathcal{R}}$ are strictly increasing with d_m.

C.7 Proof of Proposition 4.2.12

I first expand (C.7) to highlight dependence on d_m. Note that

$$\frac{d e_{\mathcal{R}}}{d\lambda} = \frac{1}{\lambda} \mathrm{Tr}\left(V_{\mathcal{M}} N \right) + k(\lambda, W, \Sigma), \tag{C.31}$$

where N is a nonpositive matrix defined as

$$N \doteq -\left(L_2^\top W_1^\top + W_2^\top \right) L_1^\top L_2, \tag{C.32}$$

and $k(\lambda, W, \Sigma) > 0$ does not depend on any d_m, $m \in \mathcal{M}$. Then, it follows

$$\frac{de_{\mathcal{R}}}{d\lambda} = \frac{1}{\lambda} \sum_{m \in \mathcal{M}} N_{mm} d_m + k(\lambda, W, \Sigma). \tag{C.33}$$

Note that $N_{mm} \neq 0$, because the opposite implies that the mth malicious agent has no (even indirect) interactions with regular agents. It follows that, for any $m \in \mathcal{M}$, there always exists $d_m > 0$ such that the error derivative (C.31) is negative, for any $\lambda < 1$. In fact, given λ, the minimal such value of d_m can be obtained from the following inequality,

$$d_m > -\frac{\lambda}{N_{mm}} k(\lambda, W, \Sigma) - \sum_{\substack{m' \in \mathcal{M} \\ m' \neq m}} \frac{N_{m'm'}}{N_{mm}} d_{m'} > 0. \tag{C.34}$$

The claim follows by combining (C.34) with Proposition 4.2.11.

References

[1] W. Abbas, A. Laszka, and X. Koutsoukos, “Improving network connectivity and robustness using trusted nodes with application to resilient consensus,” *IEEE Control Netw. Syst.*, vol. 5, no. 4, pp. 2036–2048, 2018 (Cited in page 14).

[2] N. Agmon and D. Peleg, “Fault-tolerant gathering algorithms for autonomous mobile robots,” *SIAM J. Comput.*, vol. 36, no. 1, pp. 56–82, 2006 (Cited in page 73).

[3] N. Allegra, B. Bamieh, P. Mitra, and C. Sire, “Phase transitions in distributed control systems with multiplicative noise,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2018, no. 1, p. 013 405, 2018 (Cited in page 12).

[4] S. Amin, A. A. Cárdenas, and S. S. Sastry, “Safe and secure networked control systems under denial-of-service attacks,” in *Hybrid Systems: Computation and Control*, R. Majumdar and P. Tabuada, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 31–45 (Cited in page 12).

[5] M. Amir and T. Givargis, “Priority neuron: A resource-aware neural network for cyber-physical systems,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 11, pp. 2732–2742, 2018 (Cited in page 26).

[6] J. Anderson, J. C. Doyle, S. H. Low, and N. Matni, “System level synthesis,” *Annual Reviews Control*, vol. 47, pp. 364–393, 2019 (Cited in pages 10, 55).

[7] M. Anvaripour, M. Saif, and M. Ahmadi, “A Novel Approach to Reliable Sensor Selection and Target Tracking in Sensor Networks,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 171–182, Jan. 2020 (Cited in page 3).

[8] J. Arbelaiz, B. Bamieh, A. E. Hosoi, and A. Jadbabaie, “Distributed Kalman filtering for spatially-invariant diffusion processes: The effect of noise on communication requirements,” in *Proc. IEEE CDC*, Dec. 2020, pp. 622–627 (Cited in page 9).

[9] J. Arbelaiz, B. Bamieh, A. E. Hosoi, and A. Jadbabaie, “Optimal structured controllers for spatially invariant systems: A convex reformulation,” in *Proc. IEEE CDC*, Dec. 2021, pp. 3374–3380 (Cited in page 9).

[10] M. S. Bahavarnia and N. Motee, “Sparse memoryless LQR design for uncertain linear time-delay systems,” *Proc. IFAC World Congress*, vol. 50, no. 1, pp. 10 395–10 400, 2017 (Cited in pages 9, 55).

[11] M.-F. Balcan, F. Constantin, and S. Ehrlich, “The Snowball Effect of Uncertainty in Potential Games,” in *Internet Netw. Econom.* N. Chen, E. Elkind, and E. Koutsoupias, Eds., vol. 7090, Springer Berlin Heidelberg, 2011, pp. 1–12 (Cited in page 15).

[12] L. Ballotta, G. Como, J. S. Shamma, and L. Schenato, “Can competition outperform collaboration? The role of malicious agents,” *arXiv e-prints*, no. arXiv:2207.01346, Jul. 2022, (submitted to IEEE Trans. Autom. Control) (Cited in page 23).

[13] L. Ballotta, G. Como, J. S. Shamma, and L. Schenato, “Competition-based resilience in distributed quadratic optimization,” in *Proc. IEEE CDC*, (to appear), 2022 (Cited in page 23).

[14] L. Ballotta, M. R. Jovanović, and L. Schenato, “Can decentralized control outperform centralized? The role of communication latency,” *arXiv e-prints*, no. arXiv:2109.00359, Jul. 2022, (submitted to IEEE Control Netw. Syst.) (Cited in page 23).

[15] L. Ballotta, M. R. Jovanović, and L. Schenato, “Optimal network topology of multi-agent systems subject to computation and communication latency,” in *Proc. Mediterranean Conf. Control Autom.*, 2021, pp. 249–254 (Cited in page 23).

[16] L. Ballotta, G. Peserico, and F. Zanini, “A reinforcement learning approach to sensing design in resource-constrained wireless networked control systems,” in *Proc. IEEE CDC*, (to appear), 2022 (Cited in pages 28, 44).

[17] L. Ballotta, G. Peserico, F. Zanini, and P. Dini, “To compute or not to compute? Adaptive smart sensing in resource-constrained edge computing,” *arXiv e-prints*, no. arXiv:2209.02166, Sep. 2022, (submitted to IEEE Trans. Netw. Sci. Eng.) (Cited in pages 28, 44).

[18] L. Ballotta, L. Schenato, and L. Carlone, “Computation-communication trade-offs and sensor selection in real-time estimation for processing networks,” *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2952–2965, 2020 (Cited in pages 23, 28).

[19] L. Ballotta, L. Schenato, and L. Carlone, “From sensor to processing networks: Optimal estimation with computation and communication latency,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 11 024–11 031, 2020, 21st IFAC World Congress (Cited in pages 23, 35).

[20] B. Bamieh, M. R. Jovanović, P. Mitra, and S. Patterson, “Coherence in large-scale networks: Dimension dependent limitations of local feedback,” *IEEE Trans. Automat. Control*, vol. 57, no. 9, pp. 2235–2249, 2012 (Cited in pages 9, 58).

[21] M. Baptistini and P. Táboas, “On the stability of some exponential polynomials,” *J. Math. Anal. Appl.*, vol. 205, no. 1, pp. 259–272, 1997 (Cited in page 111).

[22] J. S. Baras and X. Liu, “Trust is the cure to distributed consensus with adversaries,” in *Proc. MED*, 2019, pp. 195–202 (Cited in pages 14, 78, 96, 97).

[23] J. Barreiro-Gomez, H. Tembine, L. Stella, D. Bauso, and P. Colaneri, “Risk-aware control and games in engineering,” in *Proc. IEEE Conf. Decis. Control*, 2020, pp. 3860–3870 (Cited in page 75).

[24] M. A. Batalin and G. S. Sukhatme, “The design and analysis of an efficient local algorithm for coverage and exploration based on sensor network deployment,” *IEEE Trans. Robot.*, vol. 23, no. 4, pp. 661–675, 2007 (Cited in page 90).

[25] L. Berezansky, J. Diblík, Z. Svoboda, and Z. Šmarda, “Simple uniform exponential stability conditions for a system of linear delay differential equations,” *Appl. Math. Comput.*, vol. 250, pp. 605 –614, 2015 (Cited in page 54).

[26] C. Bisdikian, L. M. Kaplan, and M. B. Srivastava, “On the quality and value of information in sensor networks,” *ACM Trans. Sen. Netw.*, vol. 9, no. 4, 48:1–48:26, Jul. 2013 (Cited in page 5).

[27] A. Bojchevski and S. Günnemann, “Certifiable robustness to graph perturbations,” in *Proc. NeurIPS*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché Buc, E. Fox, and R. Garnett, Eds., vol. 32, 2019 (Cited in pages 12, 97).

[28] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, “Adaptive neural networks for efficient inference,” in *Proc. Int. Conf. Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proc. Mach. Learn. Research, vol. 70, PMLR, 2017, pp. 527–536 (Cited in page 5).

[29] K. Bonawitz, H. Eichner, W. Grieskamp, *et al.*, “Towards federated learning at scale: System design,” in *Proc. Machine Learning Syst.*, A. Talwalkar, V. Smith, and M. Zaharia, Eds., vol. 1, 2019, pp. 374–388 (Cited in page 74).

[30] L. Borghese and S. Braithwaite. “Hackers block italian covid-19 vaccination booking system in ’most serious cyberattack ever’.” C. Business, Ed. (web). () (Cited in page 73).

[31] V. S. Borkar and S. K. Mitter, “LQG Control with Communication Constraints,” in *Communications, Computation, Control, and Signal Processing*, A. Paulraj, V. Roychowdhury, and C. D. Schaper, Eds., Springer US, 1997, pp. 365–373 (Cited in page 8).

[32] A. Brunello, A. Urgolo, F. Pittino, A. Montvay, and A. Montanari, “Virtual Sensing and Sensors Selection for Efficient Temperature Monitoring in Indoor Environments,” *Sensors*, vol. 21, no. 8, p. 2728, Jan. 2021 (Cited in page 3).

[33] L. Carlone and S. Karaman, “Attention and anticipation in fast visual-inertial navigation,” *IEEE Trans. Robot.*, vol. 35, no. 1, pp. 1–20, 2019 (Cited in page 4).

[34] D. Cartwright *et al.*, *Studies in social power* (Publications of the Institute for Social Research: Research Center for Group Dynamics Series). Research Center for Group Dynamics, Institute for Social Research, University of Michigan, 1959 (Cited in page 87).

[35] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, “Network intrusion detection for iot security based on learning techniques,” *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2671–2701, 2019 (Cited in pages 73, 74).

[36] J. P. Champati, M. H. Mamduhi, K. H. Johansson, and J. Gross, “Performance Characterization Using AoI in a Single-loop Networked Control System,” in *Proc. IEEE INFOCOM WKSHPS*, Apr. 2019, pp. 197–203 (Cited in page 5).

[37] H. Chehardoli and A. Ghasemi, “Adaptive centralized/decentralized control and identification of 1-d heterogeneous vehicular platoons based on constant time headway policy,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 10, pp. 3376–3386, 2018 (Cited in page 8).

[38] H. Chehardoli and A. Ghasemi, “Formation control of longitudinal vehicular platoons under generic network topology with heterogeneous time delays,” *J. Vib. Control*, vol. 25, no. 3, pp. 655–665, 2019 (Cited in pages 8, 19, 54).

[39] S. Chen, H. Wen, J. Wu, W. Lei, W. Hou, W. Liu, A. Xu, and Y. Jiang, “Internet of Things Based Smart Grids Supported by Intelligent Edge Computing,” *IEEE Access*, vol. 7, pp. 74 089–74 102, 2019 (Cited in page 5).

[40] J. Cheng, M. Pavone, S. Katti, S. Chinchali, and A. Tang, “Data Sharing and Compression for Cooperative Networked Control,” in *Proc. NeurIPS*, vol. 34, Curran Associates, Inc., 2021, pp. 5947–5958 (Cited in page 6).

[41] M. Cheng, C. Yin, J. Zhang, S. Nazarian, J. Deshmukh, and P. Bogdan, “A General Trust Framework for Multi-Agent Systems,” in *Proc. AAMAS*, May 2021, pp. 332–340 (Cited in page 15).

[42] S. Chinchali, A. Sharma, J. Harrison, A. Elhafsi, D. Kang, E. Pergament, E. Cidon, S. Katti, and M. Pavone, "Network offloading policies for cloud robotics: A learning-based approach," *Auton. Robot*, vol. 45, no. 7, pp. 997–1012, Oct. 2021 (Cited in pages 6, 7).

[43] A. Chiuso, N. Laurenti, L. Schenato, and A. Zanella, "LQG cheap control subject to packet loss and SNR limitations," in *Proc. ECC*, Jul. 2013, pp. 2374–2379 (Cited in page 8).

[44] E. Clark, S. L. Brunton, and J. N. Kutz, "Multi-Fidelity Sensor Selection: Greedy Algorithms to Place Cheap and Expensive Sensors With Cost Constraints," *IEEE Sensors J.*, vol. 21, no. 1, pp. 600–611, Jan. 2021 (Cited in page 4).

[45] L. Damonte, G. Como, and F. Fagnani, "Systemic risk and network intervention," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 2856–2861, 2020, 21st IFAC World Congress (Cited in page 75).

[46] C. Davis, "All convex invariant functions of Hermitian matrices," *Archiv der Mathematik*, vol. 8, no. 4, pp. 276–278, 1957 (Cited in page 68).

[47] T. Devos, M. Kirchner, J. Croes, W. Desmet, and F. Naets, "Sensor selection and state estimation for unobservable and non-linear system models," *Sensors*, vol. 21, no. 22, 2021 (Cited in page 3).

[48] S. Dezfulian, Y. Ghaedsharaf, and N. Motee, "On performance of time-delay linear consensus networks with directed interconnection topologies," in *Proc. ACC*, 2018, pp. 4177–4182 (Cited in pages 9, 19, 54).

[49] S. M. Dibaji and H. Ishii, "Consensus of second-order multi-agent systems in the presence of locally bounded faults," *Syst. Control. Lett.*, vol. 79, pp. 23–29, 2015 (Cited in page 14).

[50] S. M. Dibaji, M. Safi, and H. Ishii, "Resilient distributed averaging," in *Proc. ACC*, 2019, pp. 96–101 (Cited in pages 22, 78, 93–96).

[51] F. Dörfler, M. R. Jovanović, M. Chertkov, and F. Bullo, "Sparsity-promoting optimal wide-area control of power networks," *IEEE Trans. Power Syst.*, vol. 29, no. 5, pp. 2281–2291, 2014 (Cited in page 55).

[52] W. Fang, Y. Zhang, B. Yu, and S. Liu, "FPGA-based ORB feature extraction for real-time visual SLAM," in *Proc. ICFPT*, Dec. 2017, pp. 275–278 (Cited in page 5).

[53] M. Fardad, F. Lin, and M. R. Jovanović, "Design of optimal sparse interconnection graphs for synchronization of oscillator networks," *IEEE Trans. Autom. Control*, vol. 59, no. 9, pp. 2457–2462, 2014 (Cited in page 10).

[54] M. Fardad, M. R. Jovanovic, and B. Bamieh, “Frequency analysis and norms of distributed spatially periodic systems,” *IEEE Trans. Autom. Control*, vol. 53, no. 10, pp. 2266–2279, 2008 (Cited in page 9).

[55] M. Fardad and M. R. Jovanović, “On the design of optimal structured and sparse feedback gains via sequential convex programming,” in *Proc. ACC*, 2014, pp. 2426–2431 (Cited in page 9).

[56] M. Fardad, F. Lin, and M. R. Jovanović, “On the optimal design of structured feedback gains for interconnected systems,” in *Proc. IEEE CDC & Chin. Control Conf.*, 2009, pp. 978–983 (Cited in page 9).

[57] A. Forootani, R. Iervolino, M. Tipaldi, and S. Dey, “Transmission scheduling for multi-process multi-sensor remote estimation via approximate dynamic programming,” *Automatica*, vol. 136, p. 110 061, 2022 (Cited in page 5).

[58] M. Franceschelli, A. Giua, and A. Pisano, “Finite-Time Consensus on the Median Value With Robustness Properties,” *IEEE Trans. Autom. Contr.*, vol. 62, no. 4, pp. 1652–1667, Apr. 2017 (Cited in page 14).

[59] N. E. Friedkin and E. C. Johnsen, “Social influence and opinions,” *J. Math. Sociol.*, vol. 15, no. 3-4, pp. 193–206, 1990. eprint: https://doi.org/10.1080/0022250X.1990.9990069 (Cited in pages 75, 79).

[60] M. I. Friswell, “The derivatives of repeated eigenvalues and their associated eigenvectors,” *J. Vib. Acoust.*, vol. 118, no. 3, pp. 390–397, 1996. eprint: https://asmedigitalcollection.asme.org/vibrationacoustics/article-pdf/118/3/390/5625239/390_1.pdf (Cited in page 68).

[61] F. Gama, Q. Li, E. Tolstaya, A. Prorok, and A. Ribeiro, “Synthesizing decentralized controllers with graph neural networks and imitation learning,” *IEEE Trans. Signal Process.*, vol. 70, pp. 1932–1946, 2022 (Cited in page 9).

[62] F. Gama and S. Sojoudi, “Distributed linear-quadratic control with graph neural networks,” *Signal Processing*, vol. 196, p. 108 506, 2022 (Cited in page 9).

[63] E. Garcia, Y. Cao, and D. W. Casbeer, “Periodic event-triggered synchronization of linear multi-agent systems with communication delays,” *IEEE Trans. Autom. Control*, vol. 62, no. 1, pp. 366–371, 2016 (Cited in pages 19, 55).

[64] B. Ghosh, S. Chinchali, and P. S. Duggirala, “Interpretable Trade-offs Between Robot Task Accuracy and Compute Efficiency,” in *Proc. IEEE/RSJ IROS*, Sep. 2021, pp. 5364–5371 (Cited in pages 6, 7).

[65] M. A. Gomez, A. V. Egorov, S. Mondié, and W. Michiels, "Optimization of the $\mathcal{H}_2$ norm for single-delay systems, with application to control design and model approximation," *IEEE Trans. Autom. Control*, vol. 64, no. 2, pp. 804–811, 2019 (Cited in pages 54, 68).

[66] R. M. Gray, "Toeplitz and circulant matrices: A review," *Found. Trends Commun. Inf. Theory*, vol. 2, no. 3, pp. 155–239, 2006 (Cited in pages 66, 71).

[67] S. Gu, F. Pasqualetti, M. Cieslak, Q. K. Telesford, A. B. Yu, A. E. Kahn, J. D. Medaglia, J. M. Vettel, M. B. Miller, S. T. Grafton, *et al.*, "Controllability of structural brain networks," *Nature communications*, vol. 6, no. 1, pp. 1–10, 2015 (Cited in page 89).

[68] S. Gupta, R. Kambli, S. Wagh, and F. Kazi, "Support-vector-machine-based proactive cascade prediction in smart grid using probabilistic framework," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2478–2486, 2015 (Cited in page 73).

[69] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray, "On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage," *Automatica*, vol. 42, no. 2, pp. 251–260, Feb. 2006 (Cited in page 3).

[70] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press, 2004 (Cited in pages 26, 29, 32).

[71] S. Hassan-Moghaddam and M. R. Jovanović, "Topology design for stochastically-forced consensus networks," *IEEE Control Netw. Syst.*, vol. 5, no. 3, pp. 1075–1086, 2018 (Cited in page 10).

[72] C. Huang, R. Zhang, and S. Cui, "Optimal power allocation for wireless sensor networks with outage constraint," *IEEE Trans. Commun. Lett.*, vol. 3, no. 2, pp. 209–212, 2014 (Cited in page 73).

[73] K. Imagane, K. Kanai, J. Katto, and T. Tsuda, "Evaluation and analysis of system latency of edge computing for multimedia data processing," in *Proc. IEEE Global Conf. Consumer Electron.*, 2016, pp. 1–2 (Cited in page 6).

[74] H. Jaleel, W. Abbas, and J. S. Shamma, "Robustness Of Stochastic Learning Dynamics To Player Heterogeneity In Games," in *Proc. IEEE CDC*, Dec. 2019, pp. 5002–5007 (Cited in page 15).

[75] S. Jawaid and S. Smith, "Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems," *Automatica*, vol. 61, pp. 282–288, 2015 (Cited in page 5).

[76] M. Jilg, J. Tonne, and O. Stursberg, “Design of distributed $\mathcal{H}_2$-optimized controllers considering stochastic communication link failures,” in *Proc. ACC*, 2015, pp. 3540–3545 (Cited in page 9).

[77] S. Joshi and S. Boyd, “Sensor selection via convex optimization,” *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 451–462, 2009 (Cited in pages 3, 34).

[78] M. R. Jovanović and N. K. Dhingra, “Controller architectures: Tradeoffs between performance and structure,” *Eur. J. Control*, vol. 30, pp. 76–91, 2016 (Cited in pages 8, 10, 18, 55).

[79] Z. Junhui, Y. Tao, G. Yi, W. Jiao, and F. Lei, “Power control algorithm of cognitive radio based on non-cooperative game theory,” *China Communications*, vol. 10, no. 11, pp. 143–154, 2013 (Cited in page 75).

[80] I. Kadota and E. Modiano, “Minimizing the Age of Information in Wireless Networks with Stochastic Arrivals,” *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 1173–1185, Mar. 2021 (Cited in page 4).

[81] Y.-H. Kao, B. Krishnamachari, M.-R. Ra, and F. Bai, “Hermes: Latency Optimal Task Assignment for Resource-constrained Mobile Computing,” *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3056–3069, Nov. 2017 (Cited in page 6).

[82] S. Kar, S. Aldosari, and J. M. F. Moura, “Topology for distributed inference on graphs,” *IEEE Trans. Signal Process.*, vol. 56, no. 6, pp. 2609–2613, 2008 (Cited in page 90).

[83] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *Int. J. Robot. Research*, vol. 30, no. 7, pp. 846–894, 2011 (Cited in page 26).

[84] H. K. Khalil, *Nonlinear Systems* (Pearson Education). Prentice Hall, 2002 (Cited in page 112).

[85] M. F. Khan, M. Bibi, F. Aadil, and J.-W. Lee, “Adaptive Node Clustering for Underwater Sensor Networks,” *Sensors*, vol. 21, no. 13, p. 4514, Jun. 2021 (Cited in page 4).

[86] M. J. Khojasteh, A. Khina, M. Franceschetti, and T. Javidi, “Learning-Based Attacks in Cyber-Physical Systems,” *IEEE Control Netw. Syst.*, vol. 8, no. 1, pp. 437–449, Mar. 2021 (Cited in page 15).

[87] S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch, and S. Martinez, "Tutorial on dynamic average consensus: The problem, its applications, and the algorithms," *IEEE Control Syst. Mag.*, vol. 39, no. 3, pp. 40–72, 2019 (Cited in page 74).

[88] R. Kieckhafer and M. Azadmanesh, "Reaching approximate agreement with mixed-mode faults," *IEEE Trans. Parallel Distrib. Syst.*, vol. 5, no. 1, pp. 53–63, 1994 (Cited in page 13).

[89] M. Klügel, M. H. Mamduhi, S. Hirche, and W. Kellerer, "AoI-Penalty Minimization for Networked Control Systems with Packet Loss," in *Proc. IEEE INFOCOM WKSHPS*, Apr. 2019, pp. 189–196 (Cited in page 5).

[90] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "The Cost of Delay in Status Updates and Their Value: Non-Linear Ageing," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 4905–4918, Aug. 2020 (Cited in page 5).

[91] W. W. Kywe, D. Fujiwara, and K. Murakami, "Scheduling of image processing using anytime algorithm for real-time system," in *18th International Conf. on Pattern Recognition (ICPR'06)*, vol. 3, 2006, pp. 1095–1098 (Cited in page 26).

[92] U. Küchler and B. Mensch, "Langevins stochastic differential equation extended by a time-delayed term," *Stochastics and Stochastic Reports*, vol. 40, no. 1-2, pp. 23–42, 1992. eprint: https://doi.org/10.1080/17442509208833780 (Cited in page 61).

[93] J. Le Ny, E. Feron, and M. A. Dahleh, "Scheduling continuous-time kalman filters," *IEEE Trans. Autom. Control*, vol. 56, no. 6, pp. 1381–1394, 2011 (Cited in page 3).

[94] H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram, "Resilient asymptotic consensus in robust networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 4, pp. 766–781, 2013 (Cited in pages 13, 22, 74, 79, 93–96).

[95] A. S. Leong, S. Dey, and D. E. Quevedo, "Sensor scheduling in variance based event triggered estimation with packet drops," *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 1880–1895, 2017 (Cited in page 5).

[96] F. Li, M. C. De Oliveira, and R. E. Skelton, "Integrating Information Architecture and Control or Estimation Design," *SICE JCMSI*, vol. 1, no. 2, pp. 120–128, Mar. 2008 (Cited in pages 3, 34).

[97] F. Li, B. R. Upadhyaya, and S. R. P. Perillo, "Fault Diagnosis of Helical Coil Steam Generator Systems of an Integral Pressurized Water Reactor Using Optimal Sensor Selection," *IEEE Trans. Nucl. Sci.*, vol. 59, no. 2, pp. 403–410, Apr. 2012 (Cited in page 4).

[98] H. Li and Y. Shi, "Network-Based Predictive Control for Constrained Nonlinear Systems With Two-Channel Packet Dropouts," *IEEE Trans. Ind. Electron.*, vol. 61, no. 3, pp. 1574–1582, Mar. 2014 (Cited in page 8).

[99] L. Li, C. Langbort, and J. Shamma, "An LP Approach for Solving Two-Player Zero-Sum Repeated Bayesian Games," *IEEE Trans. Automat. Contr.*, vol. 64, no. 9, pp. 3716–3731, Sep. 2019 (Cited in pages 15, 78).

[100] L. Li, C. Langbort, and J. Shamma, "An lp approach for solving two-player zero-sum repeated bayesian games," *IEEE Trans. Autom. Control*, vol. 64, no. 9, pp. 3716–3731, 2019 (Cited in page 97).

[101] L. Li and J. S. Shamma, "Efficient Strategy Computation in Zero-Sum Asymmetric Information Repeated Games," *IEEE Trans. Automat. Contr.*, vol. 65, no. 7, pp. 2785–2800, Jul. 2020 (Cited in pages 15, 78).

[102] L. Li and J. S. Shamma, "Efficient strategy computation in zero-sum asymmetric information repeated games," *IEEE Trans. Autom. Control*, vol. 65, no. 7, pp. 2785–2800, 2020 (Cited in page 97).

[103] N. Li and J. R. Marden, "Designing games for distributed optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 2, pp. 230–242, 2013 (Cited in page 75).

[104] S. Li, Z. Zhang, R. Mao, J. Xiao, L. Chang, and J. Zhou, "A Fast and Energy-Efficient SNN Processor With Adaptive Clock/Event-Driven Computation Scheme and Online Learning," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 4, pp. 1543–1552, Apr. 2021 (Cited in page 5).

[105] F. Lian, A. Chakrabortty, and A. Duel-Hallen, "Game-theoretic multi-agent control and network cost allocation under communication constraints," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 2, pp. 330–340, 2017 (Cited in page 55).

[106] B. Lin, F. Zhu, J. Zhang, J. Chen, X. Chen, N. N. Xiong, and J. Lloret Mauri, "A Time-Driven Data Placement Strategy for a Scientific Workflow Combining Edge Computing and Cloud Computing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4254–4265, Jul. 2019 (Cited in page 6).

[107] F. Lin, M. Fardad, and M. R. Jovanovic, “Augmented lagrangian approach to design of structured optimal state feedback gains,” *IEEE Trans. Autom. Control*, vol. 56, no. 12, pp. 2923–2929, 2011 (Cited in page 9).

[108] F. Lin, M. Fardad, and M. R. Jovanovic, “Optimal control of vehicular formations with nearest neighbor interactions,” *IEEE Trans. Autom. Control*, vol. 57, no. 9, pp. 2203–2218, 2011 (Cited in page 8).

[109] F. Lin, M. Fardad, and M. R. Jovanović, “Design of optimal sparse feedback gains via the alternating direction method of multipliers,” *IEEE Trans. Autom. Control*, vol. 58, no. 9, pp. 2426–2431, 2013 (Cited in pages 10, 55).

[110] Y. Liu, P. Ning, and M. K. Reiter, “False data injection attacks against state estimation in electric power grids,” *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, 2011 (Cited in page 12).

[111] M. C. Lucic, H. Ghazzai, A. Alsharoa, and Y. Massoud, “A Latency-Aware Task Offloading in Mobile Edge Computing Network for Distributed Elevated LiDAR,” in *Proc. IEEE ISCAS*, Oct. 2020, pp. 1–5 (Cited in page 6).

[112] N. A. Lynch, *Distributed algorithms.* San Mateo, CA, USA: Elsevier, 1996 (Cited in page 12).

[113] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, and R. P. Liu, “Energy-Efficient Admission of Delay-Sensitive Tasks for Mobile Edge Computing,” *IEEE Trans. Commun.*, vol. 66, no. 6, pp. 2603–2616, Jun. 2018 (Cited in page 6).

[114] J. Maestre, D. M. de la Peña, A. J. Losada, E. A. Durán, and E. Camacho, “An application of cooperative game theory to distributed control,” *Proc. IFAC World Congress*, vol. 44, no. 1, pp. 9121–9126, 2011, 18th IFAC World Congress (Cited in page 10).

[115] D. Maity, D. Hartman, and J. S. Baras, “Sensor scheduling for linear systems: A covariance tracking approach,” *Automatica*, vol. 136, p. 110 078, Feb. 2022 (Cited in page 5).

[116] L. Mao, L. Jackson, and B. Davies, “Effectiveness of a Novel Sensor Selection Algorithm in PEM Fuel Cell On-Line Diagnosis,” *IEEE Trans. Ind. Electron.*, vol. 65, no. 9, pp. 7301–7310, Sep. 2018 (Cited in page 4).

[117] J. R. Marden, G. Arslan, and J. S. Shamma, “Cooperative control and potential games,” *IEEE Trans. Syst., Man, Cybern., Part B (Cybern.)*, vol. 39, no. 6, pp. 1393–1407, 2009 (Cited in pages 75, 79).

[118] J. R. Marden and J. S. Shamma, "Chapter 16 - game theory and distributed control," in *Handbook of Game Theory with Economic Applications*, H. P. Young and S. Zamir, Eds., vol. 4, Elsevier, 2015, pp. 861–899 (Cited in page 75).

[119] E. Masero, J. M. Maestre, A. Ferramosca, M. Francisco, and E. F. Camacho, "Robust coalitional model predictive control with predicted topology transitions," *IEEE Control Netw. Syst.*, vol. 8, no. 4, pp. 1869–1880, 2021 (Cited in page 10).

[120] P. R. Massenio, G. Rizzello, D. Naso, F. L. Lewis, and A. Davoudi, "Data-driven optimal structured control for unknown symmetric systems," in *Proc. IEEE CASE*, 2020, pp. 179–184 (Cited in page 9).

[121] N. Matni, "Communication delay co-design in $\mathcal{H}_2$-distributed control using atomic norm minimization," *IEEE Control Netw. Syst.*, vol. 4, no. 2, pp. 267–278, 2017 (Cited in page 10).

[122] N. Matni and V. Chandrasekaran, "Regularization for design," *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 3991–4006, 2016 (Cited in pages 10, 55).

[123] H. Medeiros, J. Park, and A. Kak, "Distributed object tracking using a cluster-based kalman filter in wireless camera networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 2, no. 4, pp. 448–463, 2008 (Cited in page 50).

[124] W. Michiels, G. Hilhorst, G. Pipeleers, and J. Swevers, "Model order reduction for time-delay systems, with application to fixed-order $\mathcal{H}_2$ optimal controller design," in *Recent Results on Time-Delay Systems: Analysis and Control.* Springer International Publishing, 2016, pp. 45–66 (Cited in page 54).

[125] D. G. Mikulski, F. L. Lewis, E. Y. Gu, and G. R. Hudas, "Trust method for multi-agent consensus," in *Unmanned Systems Technology XIV*, R. E. Karlsen, D. W. Gage, C. M. Shoemaker, and G. R. Gerhart, Eds., International Society for Optics and Photonics, vol. 8387, 2012, pp. 146 –159 (Cited in page 14).

[126] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Cham: Springer International Publishing, 2020, pp. 405–421 (Cited in page 5).

[127] R. Mitchell, J. Blumenkamp, and A. Prorok, "Gaussian Process Based Message Filtering for Robust Multi-Agent Cooperation in the Presence of Adversarial Communication," *arXiv e-prints*, no. arXiv:2012.00508, Dec. 2020 (Cited in page 15).

[128] Y. Mo, T. H.-J. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber–physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012 (Cited in page 12).

[129] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2009, pp. 911–918 (Cited in page 12).

[130] R. Moarref, M. Fardad, and M. R. Jovanovic, "Perturbation analysis of eigenvalues of a class of self-adjoint operators," in *Proc. ACC*, 2008, pp. 955–960 (Cited in page 9).

[131] M. M. Morato and J. E. Normey-Rico, "A novel unified method for time-varying dead-time compensation," *ISA Trans.*, vol. 108, pp. 78–95, 2021 (Cited in page 54).

[132] F. J. Muros, J. M. Maestre, E. Algaba, T. Alamo, and E. F. Camacho, "Networked control design for coalitional schemes using game-theoretic methods," *Automatica*, vol. 78, pp. 320–332, 2017 (Cited in pages 10, 75).

[133] F. J. Muros, J. M. Maestre, C. Ocampo-Martinez, E. Algaba, and E. F. Camacho, "A game theoretical randomized method for large-scale systems partitioning," *IEEE Access*, vol. 6, pp. 42 245–42 263, 2018 (Cited in page 10).

[134] N. Muslim, S. Islam, and J.-C. Grégoire, "Reinforcement Learning Based Offloading Framework for Computation Service in the Edge Cloud and Core Cloud," *JAIT*, vol. 13, no. 2, 2022 (Cited in page 6).

[135] U. Münz, A. Papachristodoulou, and F. Allgöwer, "Delay robustness in consensus problems," *Automatica*, vol. 46, no. 8, pp. 1252–1265, 2010 (Cited in pages 9, 19, 54).

[136] G. N. Nair and R. J. Evans, "Stabilizability of Stochastic Linear Systems with Finite Feedback Data Rates," *SIAM J. Control Optim.*, vol. 43, no. 2, pp. 413–436, Jan. 2004 (Cited in page 8).

[137] M. Nakanoya, S. Chinchali, A. Anemogiannis, A. Datta, S. Katti, and M. Pavone, "Co-Design of Communication and Machine Inference for Cloud Robotics," in *Robotics: Science and Systems*, Robotics: Science and Systems Foundation, Jul. 2021 (Cited in page 6).

[138] R. B. Nelson, "Simplified calculation of eigenvector derivatives," *AIAA J.*, vol. 14, no. 9, pp. 1201–1205, 1976. eprint: https://doi.org/10.2514/3.7211 (Cited in page 68).

[139] E. Nozari, F. Pasqualetti, and J. Cortés, "Heterogeneity of central nodes explains the benefits of time-varying control scheduling in complex dynamical networks," *J. Complex Netw.*, vol. 7, no. 5, pp. 659–701, Feb. 2019. eprint: https://academic.oup.com/comnet/article-pdf/7/5/659/30157079/cnz001.pdf (Cited in page 89).

[140] E. Nozari, F. Pasqualetti, and J. Cortés, "Time-invariant versus time-varying actuator scheduling in complex networks," in *Proc. ACC*, 2017, pp. 4995–5000 (Cited in page 3).

[141] F. de Oliveira Souza, L. A. B. Torres, L. A. Mozelli, and A. A. Neto, "Stability and formation error of homogeneous vehicular platoons with communication time delays," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4338–4349, 2020 (Cited in page 54).

[142] T. Z. Ornee and Y. Sun, "Sampling and Remote Estimation for the Ornstein-Uhlenbeck Process Through Queues: Age of Information and Beyond," *IEEE/ACM Trans. Netw.*, vol. 29, no. 5, pp. 1962–1975, Oct. 2021 (Cited in page 5).

[143] R. Paluch, L. G. Gajewski, J. A. Hołyst, and B. K. Szymanski, "Optimizing sensors placement in complex networks for localization of hidden signal source: A review," *Future Gener. Comput. Syst.*, vol. 112, pp. 1070–1092, 2020 (Cited in page 90).

[144] S. E. Parsegov, A. V. Proskurnikov, R. Tempo, and N. E. Friedkin, "Novel multidimensional models of opinion dynamics in social networks," *IEEE Trans. Autom. Control*, vol. 62, no. 5, pp. 2270–2285, 2017 (Cited in pages 81, 88, 128).

[145] F. Pasqualetti, A. Bicchi, and F. Bullo, "Consensus computation in unreliable networks: A system theoretic approach," *IEEE Trans. Autom. Control*, vol. 57, no. 1, pp. 90–104, 2012 (Cited in page 14).

[146] M. Pezzutto, L. Schenato, and S. Dey, "Transmission power allocation for remote estimation with multi-packet reception capabilities," *Automatica*, vol. 140, p. 110 257, 2022 (Cited in page 5).

[147] M. Pezzutto, F. Tramarin, S. Dey, and L. Schenato, "Adaptive transmission rate for LQG control over Wi-Fi: A cross-layer approach," *Automatica*, vol. 119, p. 109 092, Sep. 2020 (Cited in page 8).

[148] E. Phaisangittisagul and H. T. Nagle, "Sensor Selection for Machine Olfaction Based on Transient Feature Extraction," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 2, pp. 369–378, Feb. 2008 (Cited in page 4).

[149] U. Prells and M. I. Friswell, “Calculating derivatives of repeated and nonrepeated eigenvalues without explicit use of eigenvectors,” *AIAA J.*, vol. 38, no. 8, pp. 1426–1436, 2000. eprint: https://doi.org/10.2514/2.1119 (Cited in page 68).

[150] A. Prorok, M. Malencia, L. Carlone, G. S. Sukhatme, B. M. Sadler, and V. Kumar, “Beyond Robustness: A Taxonomy of Approaches towards Resilient Multi-Robot Systems,” *arXiv e-prints*, no. arXiv:2109.12343, Sep. 2021 (Cited in pages 15, 73).

[151] A. V. Proskurnikov and R. Tempo, “A tutorial on modeling and analysis of dynamic social networks. part i,” *Annual Reviews in Control*, vol. 43, pp. 65–79, 2017 (Cited in page 75).

[152] J. Qi, J. Wang, and K. Sun, “Efficient estimation of component interactions for cascading failure analysis by em algorithm,” *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 3153–3161, 2018 (Cited in page 73).

[153] M. A. Rahimian, A. Ajorlou, and A. G. Aghdam, “Characterization of link failures in multi-agent systems under the agreement protocol,” in *2012 American Control Conference (ACC)*, 2012, pp. 5258–5263 (Cited in page 12).

[154] M. Rahnamay-Naeini and M. M. Hayat, “Cascading failures in interdependent infrastructures: An interdependent markov-chain approach,” *IEEE Trans. Smart Grid*, vol. 7, no. 4, pp. 1997–2006, 2016 (Cited in page 73).

[155] M. M. Rana, L. Li, and S. W. Su, “Cyber attack protection and control in microgrids using channel code and semidefinite programming,” in *Proc. IEEE PESGM*, 2016, pp. 1–5 (Cited in page 73).

[156] S. Raskin. “Energy secretary says enemies are capable of shutting down us power grid.” N. Y. Post, Ed. (web). () (Cited in page 73).

[157] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” *arXiv e-prints*, arXiv:1804.02767, arXiv:1804.02767, 2018. arXiv: 1804.02767 [cs.CV] (Cited in page 26).

[158] H. Ren, G. Zong, L. Hou, and Y. Yang, “Finite-time resilient decentralized control for interconnected impulsive switched systems with neutral delay,” *ISA Trans.*, vol. 67, pp. 19–29, 2017 (Cited in pages 19, 54).

[159] W. Ren and R. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Trans. Autom. Control*, vol. 50, no. 5, pp. 655–661, 2005 (Cited in pages 12, 97).

[160] O. Rippel and L. Bourdev, “Real-time adaptive image compression,” in *Proc. of the 34th International Conf. on Machine Learning - Volume 70*, ser. ICML’17, Sydney, NSW, Australia: JMLR.org, 2017, 2922–2930 (Cited in pages 26, 36).

[161] G. Rudolph, “Convergence rates of evolutionary algorithms for quadratic convex functions with rank-deficient hessian,” in *Adaptive and Natural Computing Algorithms*, M. Tomassini, A. Antonioni, F. Daolio, and P. Buesser, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 151–160 (Cited in page 104).

[162] C. A. Rösinger and C. W. Scherer, “A flexible synthesis framework of structured controllers for networked systems,” *IEEE Control Netw. Syst.*, vol. 7, no. 1, pp. 6–18, 2020 (Cited in page 9).

[163] J. Rüth, R. Glebke, T. Ulmen, and K. Wehrle, “Demo abstract: Towards in-network processing for low-latency industrial control,” in *Proc. IEEE INFOCOM WKSHPS*, Apr. 2018, pp. 1–2 (Cited in page 6).

[164] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, “Distributed federated learning for ultra-reliable low-latency vehicular communications,” *IEEE Trans. Commun.*, vol. 68, no. 2, pp. 1146–1159, 2020 (Cited in page 74).

[165] Z. A. Z. Sanai Dashti, C. Seatzu, and M. Franceschelli, “Dynamic Consensus on the Median Value in Open Multi-Agent Systems,” in *Proc. IEEE CDC*, Dec. 2019, pp. 3691–3697 (Cited in page 14).

[166] L. Schenato, “Optimal estimation in networked control systems subject to random delay and packet drop,” *IEEE Trans. on Automatic Control*, vol. 53, pp. 1311–1317, 2008 (Cited in pages 40, 107).

[167] L. Schenato, “Optimal sensor fusion for distributed sensors subject to random delay and packet loss,” in *Proc. IEEE CDC*, 2007, pp. 1547–1552 (Cited in pages 32, 41, 106).

[168] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, “Foundations of Control and Estimation Over Lossy Networks,” *Proc. IEEE*, vol. 95, no. 1, pp. 163–187, Jan. 2007 (Cited in page 8).

[169] I. D. Schizas, G. Mateos, and G. B. Giannakis, “Distributed lms for consensus-based in-network adaptive processing,” *IEEE Trans. Signal Process.*, vol. 57, no. 6, pp. 2365–2382, 2009 (Cited in page 74).

[170] S. Schön, C. Brenner, H. Alkhatib, *et al.*, “Integrity and Collaboration in Dynamic Sensor Networks,” *Sensors*, vol. 18, no. 7, p. 2400, Jul. 2018 (Cited in page 4).

[171] E. Shafieepoorfard and M. Raginsky, “Rational inattention in scalar LQG control,” in *Proc. IEEE CDC*, Dec. 2013, pp. 5733–5739 (Cited in page 8).

[172] S. Shahrampour and A. Jadbabaie, “Distributed online optimization in dynamic environments using mirror descent,” *IEEE Transactions on Automatic Control*, vol. 63, no. 3, pp. 714–725, 2018 (Cited in page 74).

[173] Y. Shang, “Median-Based Resilient Consensus Over Time-Varying Random Networks,” *IEEE Trans. Circuits Syst. II*, vol. 69, no. 3, pp. 1203–1207, Mar. 2022 (Cited in page 14).

[174] Y. Shang, “Resilient consensus in multi-agent systems with state constraints,” *Automatica*, vol. 122, p. 109 288, 2020 (Cited in page 14).

[175] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, “Kalman filtering with intermittent observations,” *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1453–1464, 2004 (Cited in pages 8, 32, 41).

[176] P. Y. Sohouenou, P. Christidis, A. Christodoulou, L. A. Neves, and D. L. Presti, “Using a random road graph model to understand road networks robustness to link failures,” *Int. J. Crit. Infrastruct. Prot.*, vol. 29, p. 100 353, 2020 (Cited in pages 12, 97).

[177] D. Soudbakhsh, A. Chakrabortty, and A. M. Annaswamy, “A delay-aware cyber-physical architecture for wide-area control of power systems,” *Control Eng. Practice*, vol. 60, pp. 171–182, 2017 (Cited in pages 19, 54).

[178] M. V. Srighakollapu, R. K. Kalaimani, and R. Pasumarthy, “Optimizing network topology for average controllability,” *Syst. Control Lett.*, vol. 158, p. 105 061, 2021 (Cited in page 89).

[179] L. Su and N. Vaidya, “Multi-agent optimization in the presence of byzantine adversaries: Fundamental limits,” in *2016 American Control Conference (ACC)*, 2016, pp. 7183–7188 (Cited in page 12).

[180] L. Su and N. H. Vaidya, “Byzantine-resilient multiagent optimization,” *IEEE Trans. Autom. Control*, vol. 66, no. 5, pp. 2227–2233, 2021 (Cited in page 15).

[181] T. Summers and J. Ruths, “Performance bounds for optimal feedback control in networks,” in *Proc. ACC*, 2018, pp. 203–209 (Cited in page 3).

[182] T. H. Summers, F. L. Cortesi, and J. Lygeros, “On submodularity and controllability in complex dynamical networks,” *IEEE Control Netw. Syst.*, vol. 3, no. 1, pp. 91–101, 2016 (Cited in page 3).

[183] S. Sun, H. Zhang, W. Li, and Y. Wang, "Time-varying delay-dependent finite-time boundedness with $\mathcal{H}_\infty$ performance for markovian jump neural networks with state and input constraints," *Neurocomputing*, vol. 423, pp. 419–426, 2021 (Cited in page 54).

[184] Y. Sun, Y. Polyanskiy, and E. Uysal, "Sampling of the Wiener Process for Remote Estimation Over a Channel With Random Delay," *IEEE Trans. Inf. Theory*, vol. 66, no. 2, pp. 1118–1135, Feb. 2020 (Cited in page 5).

[185] S. Sundaram and B. Gharesifard, "Consensus-based distributed optimization with malicious nodes," in *Proc. Allerton*, 2015, pp. 244–249 (Cited in pages 12, 15, 94).

[186] S. Sundaram and C. N. Hadjicostis, "Distributed function calculation via linear iterative strategies in the presence of malicious agents," *IEEE Trans. Autom. Control*, vol. 56, no. 7, pp. 1495–1508, 2011 (Cited in page 15).

[187] D. Swaroop and J. Hedrick, "String stability of interconnected systems," *IEEE Trans. Autom. Control*, vol. 41, no. 3, pp. 349–357, 1996 (Cited in page 9).

[188] T. Taami, S. Krug, and M. O'Nils, "Experimental Characterization of Latency in Distributed IoT Systems with Cloud Fog Offloading," in *Proc. IEEE WFCS*, May 2019, pp. 1–4 (Cited in page 6).

[189] R. Talak, S. Karaman, and E. Modiano, "Optimizing Information Freshness in Wireless Networks Under General Interference Constraints," *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 15–28, Feb. 2020 (Cited in page 4).

[190] R. Talak and E. H. Modiano, "Age-Delay Tradeoffs in Queueing Systems," *IEEE Trans. Inf. Theory*, vol. 67, no. 3, pp. 1743–1758, Mar. 2021 (Cited in page 4).

[191] T. Tanaka and H. Sandberg, "SDP-based joint sensor and controller design for information-regularized optimal LQG control," in *Proc. IEEE CDC*, 2015, pp. 4486–4491 (Cited in page 4).

[192] S. Tatikonda and S. Mitter, "Control under communication constraints," *IEEE Trans. Autom. Control*, vol. 49, no. 7, pp. 1056–1068, Jul. 2004 (Cited in page 8).

[193] A. Teixeira, S. Amin, H. Sandberg, K. H. Johansson, and S. S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5991–5998 (Cited in page 12).

[194] Y. Tian, P. Jia, A. MirTabatabaei, L. Wang, N. E. Friedkin, and F. Bullo, "Social power evolution in influence networks with stubborn individuals," *IEEE Trans. Autom. Control*, vol. 67, no. 2, pp. 574–588, 2022 (Cited in page 87).

[195] V. Tripathi, L. Ballotta, L. Carlone, and E. Modiano, “Computation and communication co-design for real-time monitoring and control in multi-agent systems,” in *Proc. WiOpt*, 2021, pp. 1–8 (Cited in page 49).

[196] V. Tripathi and E. Modiano, “An Online Learning Approach to Optimizing Time-Varying Costs of AoI,” in *Proc. ACM MOBIHOC*, Jul. 2021, pp. 241–250 (Cited in page 5).

[197] V. Tripathi, R. Talak, and E. Modiano, “Age Optimal Information Gathering and Dissemination on Graphs,” *IEEE Trans. Mobile Comput.*, pp. 1–1, 2021 (Cited in page 4).

[198] V. Tsiatsis, R. Kumar, and M. B. Srivastava, “Computation hierarchy for in-network processing,” *Mob. Netw. Appl.*, vol. 10, no. 4, pp. 505–518, 2005 (Cited in page 6).

[199] V. Tzoumas, A. Jadbabaie, and G. Pappas, “Sensor placement for optimal kalman filtering: Fundamental limits, submodularity, and algorithms,” in *Proc. ACC*, 2016 (Cited in page 3).

[200] V. Tzoumas, L. Carlone, G. J. Pappas, and A. Jadbabaie, “Lqg control and sensing co-design,” *IEEE Trans. Autom. Control*, vol. 66, no. 4, pp. 1468–1483, 2021 (Cited in pages 4, 34, 44).

[201] J. Usevitch and D. Panagou, “Resilient leader-follower consensus to arbitrary reference values in time-varying graphs,” *IEEE Trans. Autom. Control*, vol. 65, no. 4, pp. 1755–1762, 2020 (Cited in page 14).

[202] M. E. Valcher, “Consensus in the presence of communication faults,” in *2019 18th European Control Conference (ECC)*, 2019, pp. 1062–1067 (Cited in page 12).

[203] M. E. Valcher and G. Parlangeli, “On the effects of communication failures in a multi-agent consensus network,” in *Proc. ICSTCC*, 2019, pp. 709–720 (Cited in pages 12, 97).

[204] S. Vanka, V. Gupta, and M. Haenggi, “Power-delay analysis of consensus algorithms on wireless networks with interference,” *Int. J. Syst. Control Commun.*, vol. 2, no. 1-3, pp. 256–274, 2010. eprint: `https://www.inderscienceonline.com/doi/pdf/10.1504/IJSCC.2010.031166` (Cited in pages 19, 55, 56).

[205] S. Vanka, M. Haenggi, and V. Gupta, “Convergence speed of the consensus algorithm with interference and sparse long-range connectivity,” *IEEE J. Sel. Top. Signal Process.*, vol. 5, no. 4, pp. 855–865, 2011 (Cited in pages 19, 55, 56).

[206] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When Edge Meets Learning: Adaptive Control for Resource-Constrained Distributed Machine Learning," in *Proc. IEEE INFOCOM*, Apr. 2018, pp. 63–71 (Cited in page 5).

[207] Y. Wang, H. Ishii, F. Bonnet, and X. Défago, "Resilient consensus against mobile malicious agents," in *Proc. IFAC World Congress*, vol. 53, 2020, pp. 3409–3414 (Cited in pages 14, 78).

[208] Z. Wang, X. Li, and J. Lei, "Second moment boundedness of linear stochastic delay differential equations," *Discrete Contin. Dyn. Syst. - B*, vol. 19, no. 9, pp. 2963 –2991, 2014 (Cited in pages 62, 111).

[209] P. Warden and D. Situnayake, *TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers*. O'Reilly Media, 2019 (Cited in page 5).

[210] S. Wu, K. Ding, P. Cheng, and L. Shi, "Optimal scheduling of multiple sensors over lossy and bandwidth limited channels," *IEEE Control Netw. Syst.*, vol. 7, no. 3, pp. 1188–1200, 2020 (Cited in page 5).

[211] X. Wu and M. R. Jovanović, "Sparsity-promoting optimal control of systems with symmetries, consensus and synchronization networks," *Syst. Control Lett.*, vol. 103, pp. 1–8, 2017 (Cited in page 10).

[212] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007 (Cited in pages 11, 74).

[213] R. Xin, S. Pu, A. Nedić, and U. A. Khan, "A general framework for decentralized optimization with first-order methods," *Proc. IEEE*, vol. 108, no. 11, pp. 1869–1889, 2020 (Cited in pages 74, 97).

[214] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, 2016 (Cited in pages 73, 74).

[215] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *IEEE Trans. Inf. Theory*, vol. 65, no. 3, pp. 1807–1827, 2019 (Cited in page 4).

[216] A. Y. Yazıcıoğlu, M. Egerstedt, and J. S. Shamma, "Formation of robust multi-agent networks through self-organizing random regular graphs," *IEEE Trans. Netw. Sci. Eng.*, vol. 2, no. 4, pp. 139–151, 2015 (Cited in pages 73, 90).

[217] Y. Ye, L. Shi, X. Chu, H. Zhang, and G. Lu, "On the outage performance of swipt-based three-step two-way df relay networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3016–3021, 2019 (Cited in page 73).

[218] M. Yemini, A. Nedi/'c, A. J. Goldsmith, and S. Gil, "Characterizing trust and resilience in distributed consensus for cyberphysical systems," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 71–91, 2022 (Cited in pages 14, 78, 97).

[219] M. Yemini, A. Nedi/'c, A. J. Goldsmith, and S. Gil, "Resilience to malicious activity in distributed optimization for cyberphysical systems," in *Proc. IEEE CDC*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, Eds., (to appear, 2022 (Cited in page 15).

[220] S. Yu, Y. Chen, and S. Kar, "Dynamic Median Consensus Over Random Networks," in *Proc. IEEE CDC*, Dec. 2021, pp. 5695–5702 (Cited in page 14).

[221] D. Yue and Q.-L. Han, "Delayed feedback control of uncertain systems with time-varying input delay," *Automatica*, vol. 41, no. 2, pp. 233–240, 2005 (Cited in page 19).

[222] G. Zardini, A. Censi, and E. Frazzoli, "Co-design of autonomous systems: From hardware selection to control synthesis," in *Proc. ECC*, 2021, pp. 682–689 (Cited in page 4).

[223] G. Zardini, Z. Suter, A. Censi, and E. Frazzoli, "Task-driven modular co-design of vehicle control systems," *arXiv e-prints*, no. arXiv:2203.16640, 2022 (Cited in page 4).

[224] Y. Zhai, Z.-W. Liu, M.-F. Ge, G. Wen, X. Yu, and Y. Qin, "Trusted-region subsequence reduction for designing resilient consensus algorithms," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 259–268, 2021 (Cited in page 14).

[225] H. Zhang and S. Sundaram, "A simple median-based resilient consensus algorithm," in *Proc. Allerton*, Oct. 2012, pp. 1734–1741 (Cited in page 14).

[226] J. Zhang and V. Varadharajan, "Wireless sensor network key management survey and taxonomy," *J. Netw. Comput. Appl.*, vol. 33, no. 2, pp. 63–75, 2010 (Cited in page 90).

[227] Y. Zhao, F. Pasqualetti, and J. Cortés, "Scheduling of control nodes for improved network controllability," in *Proc. IEEE CDC*, 2016, pp. 1859–1864 (Cited in page 5).

[228] B. Zhou and W. Saad, “Joint status sampling and updating for minimizing age of information in the internet of things,” *IEEE Trans. Commun.*, pp. 1–1, 2019 (Cited in page 4).

[229] S. Zilberstein, “Using anytime algorithms in intelligent systems,” *AI Magazine*, vol. 17, no. 3, 1996 (Cited in pages 25, 26).

[230] D. M. Zoltowski, N. Dhingra, F. Lin, and M. R. Jovanović, “Sparsity-promoting optimal control of spatially-invariant systems,” in *Proc. ACC*, 2014, pp. 1255–1260 (Cited in page 10).

[231] X. Zong, T. Li, G. Yin, L. Y. Wang, and J.-F. Zhang, “Stochastic consentability of linear systems with time delays and multiplicative noises,” *IEEE Trans. Autom. Control*, vol. 63, no. 4, pp. 1059–1074, 2018 (Cited in pages 9, 19).

[232] X. Zong, T. Li, and J.-F. Zhang, “Consensus conditions of continuous-time multi-agent systems with time-delays and measurement noises,” *Automatica*, vol. 99, pp. 412–419, 2019 (Cited in pages 9, 19).