RESILIENT SUBMODULAR MAXIMIZATION FOR CONTROL AND SENSING

Vasileios Tzoumas

A DISSERTATION

in

Electrical and Systems Engineering

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2018

Dissertation Supervisor

George J. Pappas, Professor of Electrical and Systems Engineering (UPenn)

Dissertation Co-Supervisor

Ali Jadbabaie, Professor of Engineering, Institute for Data, Systems and Society (MIT)

Graduate Group Chairperson

Alejandro Ribeiro, Associate Professor of Electrical and Systems Engineering (UPenn)

Dissertation Committee

Rakesh Vohra, Professor of Economics and of Electrical and Systems Engineering (UPenn)

Hamed Hassani, Assistant Professor of Electrical and Systems Engineering (UPenn)

Luca Carlone, Assistant Professor of Aeronautics and Astronautics (MIT)

ABSTRACT

RESILIENT SUBMODULAR MAXIMIZATION FOR CONTROL AND SENSING

Vasileios Tzoumas
George J. Pappas
Ali Jadbabaie

Fundamental applications in control, sensing, and robotics, motivate the design of systems by selecting system elements, such as actuators or sensors, subject to constraints that require the elements not only to be a few in number, but also, to satisfy heterogeneity or interdependency constraints (called matroid constraints). For example, consider the scenarios:

- (*Control*) *Actuator placement*: In a power grid, how should we place a few generators both to guarantee its stabilization with minimal control effort, and to satisfy interdependency constraints where the power grid must be controllable from the generators?

- (*Sensing*) *Sensor placement*: In medical brain-wearable devices, how should we place a few sensors to ensure smoothing estimation capabilities?

- (*Robotics*) *Sensor scheduling*: At a team of mobile robots, which few on-board sensors should we activate at each robot —subject to heterogeneity constraints on the number of sensors that each robot can activate at each time— so both to maximize the robots' battery life, and to ensure the robots' capability to complete a formation control task?

In the first part of this thesis we motivate the above design problems, and propose the first algorithms to address them. In particular, although traditional approaches to matroid-constrained maximization have met great success in machine learning and facility location, they are unable to meet the aforementioned problem of actuator placement. In addition, although traditional approaches to sensor selection enable Kalman filtering capabilities, they do not enable smoothing or formation control capabilities, as required in the above problems of sensor placement and scheduling. Therefore, in the first part of the thesis we provide the first algorithms, and prove they achieve the following characteristics: *provable approximation performance*: the algorithms guarantee a solution close to the optimal; *minimal running time*: the algorithms terminate with the same running time as state-of-the-art algorithms for matroid-constrained maximization; *adaptiveness*: where applicable, at each time step the algorithms select system elements based on both the history of selections. We achieve the above ends by taking advantage of a submodular structure of in all aforementioned problems —submodularity is a diminishing property for set functions, parallel to convexity for continuous functions.

But in failure-prone and adversarial environments, sensors and actuators can fail; sensors and actuators can get attacked. Thence, the traditional design paradigms over matroid-constraints become insufficient, and in contrast, resilient designs against attacks or failures become important. However, no approximation algorithms are known for their solution; relevantly, the problem of resilient maximization over matroid constraints is NP-hard.

In the second part of this thesis we motivate the general problem of resilient maximization over matroid constraints, and propose the first algorithms to address it, to protect that way *any* design over matroid constraints, not only within the boundaries of control, sensing, and robotics, but also within machine learning, facility location, and matroid-constrained optimization in general. In particular, in the second part of this thesis we provide the first algorithms, and prove they achieve the following characteristics: *resiliency*: the algorithms are valid for any number of attacks or failures; *adaptiveness*: where applicable, at each time step the algorithms select system elements based on both the history of selections, and on the history of attacks or failures; *provable approximation guarantees*: the algorithms guarantee for any submodular or merely monotone function a solution close to the optimal; *minimal running time*: the algorithms terminate with the same running time as state-of-the-art algorithms for matroid-constrained maximization. We bound the performance of our algorithms by using notions of curvature for monotone (not necessarily submodular) set functions, which are established in the literature of submodular maximization.

In the third and final part of this thesis we apply our tools for resilient maximization in robotics, and in particular, to the problem of active information gathering with mobile robots. This problem calls for the motion-design of a team of mobile robots so to enable the effective information gathering about a process of interest, to support, e.g., critical missions such as hazardous environmental monitoring, and search and rescue. Therefore, in the third part of this thesis we aim to protect such multi-robot information gathering tasks against attacks or failures that can result to the withdrawal of robots from the task. We conduct both numerical and hardware experiments in multi-robot multi-target tracking scenarios, and exemplify the benefits, as well as, the performance of our approach.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1 : INTRODUCTION

1.1. Motivation of submodular maximization in control, sensing, and robotics

Researchers in control, sensing, and robotics envision the design of critical infrastructures and autonomous systems in applications such as:

- (*Control*) *Power-grid stabilization*: Deploy new-technology HVDC generators in power grids to guarantee their stabilization. [1]
- (*Sensing*) *Search and rescue*: Deploy mobile robots to localize people trapped in burning buildings. [2]
- (*Robotics*) *Multi-target coverage*: Deploy aerial micro-robots to monitor targets that move in a cluttered urban environment. [3]

In particular, all the aforementioned applications motivate fundamental set function optimization problems such as:

- (*Control*) *Actuator placement*: In a power grid, how should we place a few generators both to guarantee its stabilization, and to satisfy global-interdependency constraints where the power grid must be controllable from the generators? [4]
- (*Sensing*) *Sensor scheduling*: At a team of mobile robots, which few on-board sensors should we activate at each robot —subject to heterogeneity constraints on the number of sensors each robot can activate— so both to maximize the robots' battery life, and to ensure the robots' capability to complete a formation control task? [5]
- (*Robotics*) *Motion planning*: At a team of aerial robots, how should we select the robots' motions to maximize the team's capability for tracking targets moving in urban environments, subject to heterogeneity constraints where each robot has different motion capabilities? [6]

Specifically, all the above applications motivate the design of systems by selecting system elements, such as actuators, sensors, or movements, subject to complex design constraints that require the system elements not only to be a few in number, but also to possibly satisfy heterogeneity or global-interdependency constraints. Other general fundamental problems that involve such complex design constraints are:

- (*Control*) Sparse actuation design for state reachability or low-control effort [4], or merely for controllability [7] or structural controllability [8]; and synchronization in complex networks for tasks of motion coordination [9].
- (*Sensing*) Sparse sensing design for optimal Kalman filtering [5, 10].
- (*Robotics*) Task allocation in collaborative multi-robot systems for surveillance in urban environments [11].

In more detail, all the aforementioned problems and applications require the solution to an optimization problem of the form:

$$\max_{\mathcal{A}\subseteq\mathcal{V},\, \mathcal{A}\in\mathcal{I}} \quad f(\mathcal{A}), \tag{1.1}$$

where the set $\mathcal{V}$ represent a set of available elements to choose from; the set $\mathcal{I}$ represents the collection of complex design constraints —called *matroids* [12]— that enforce heterogeneity or global-interdependency across the elements in $\mathcal{A}$; and the objective function f is non-decreasing and (possibly) submodular; submodularity is a diminishing returns property. For example, $\mathcal{I}$ may constrain the cardinality of each feasible set in the problem in eq. (1.1), e.g., when $\mathcal{I} = \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, |\mathcal{A}| \leq \alpha\}$, given some positive integer α; an interpretation of the number α is that it captures a resource constraint, such as a limited battery for sensor activation, which limits the number of elements one can select in $\mathcal{A}$ (under the implicit assumption that all the elements in $\mathcal{V}$ consume the same amount of the limited resource). In some cases, however, different elements may consume different amounts of the limited resource; for example, different sensors may have different battery consumption. In such heterogeneity scenarios, $\mathcal{I}$ may constrain the cost of each feasible set in the problem in eq. (1.1), e.g., by being $\mathcal{I} = \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, c(\mathcal{A}) \leq b\}$, given some cost function $c(\mathcal{A})$ over all the possible subsets $\mathcal{A} \subseteq \mathcal{V}$, and given some budget constraint b; that is, the cost function c captures the heterogeneity in the cost of each element in $\mathcal{V}$. More generally, $\mathcal{I}$ may also enforce heterogeneity to the elements in $\mathcal{A}$ by partitioning the elements in $\mathcal{V}$, and permitting the selection of only a few elements from each partition, e.g., when $\mathcal{V} = \mathcal{V}_1 \cup \cdots \cup \mathcal{V}_n$ and $\mathcal{I} = \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, c_i(\mathcal{A} \cap \mathcal{V}_i) \leq b_i, \text{ for all } i = 1, \ldots, n\}$, given a positive integer n, a partition $\mathcal{V}_1, \ldots, \mathcal{V}_n$ of $\mathcal{V}$, cost functions $c_1, \ldots, c_n$, and budget constraints $b_1, \ldots, b_n$. In particular, we may give two interpretations of the heterogeneity introduced by the sets $\mathcal{V}_1, \ldots, \mathcal{V}_n$: the first interpretation considers that the sets $\mathcal{V}_1, \ldots, \mathcal{V}_n$ correspond to the available elements across n different *types* (buckets) of elements, and correspondingly, the budgets $b_1, \ldots, b_n$ constrain the total cost of the elements one can use from each type $1, \ldots, n$; and the second interpretation considers that the sets $\mathcal{V}_1, \ldots, \mathcal{V}_n$ correspond to the available elements across n different *times*, and correspondingly, the budget constraints $b_1, \ldots, b_n$ constrain the total cost of the elements one can use at each time $1, \ldots, n$. Finally, in other complex design scenarios, that call for global-interdependency among the selected elements, $\mathcal{I}$ may require the elements in $\mathcal{A}$ to form, e.g., a spanning tree on a graph associated to $\mathcal{V}$, such as in the aforementioned scenario of leader selection for structural controllability [8].

1.2. State-of-the-art approaches for submodular maximization

Overall, the optimization problem in eq. (1.1) is combinatorial, and, in particular, it is NP-hard [13]; notwithstanding, greedy-like algorithms have been proposed for its solution [12, 14], such as the greedy presented in Algorithm 1. Specifically, Algorithm 1 builds sequentially an approximate solution for the problem in eq. (1.1), by starting with an empty set $\mathcal{A}$ (line 1 of Algorithm 1), and then by adding in $\mathcal{A}$ one element at a time (lines 2-8 of Algorithm 1); in particular, any element that achieves the highest value of $f(\mathcal{A} \cup \{y\})$ among the elements $y \in \mathcal{V}$ that not chosen so far (line 5 of Algorithm 1) and for which the feasibility constraint $\mathcal{A} \cup \{y\} \in \mathcal{I}$ is satisfied (lines 4 of Algorithm 1). Similarly, the rest of the state-of-the-art algorithms for the problem in eq. (1.1), i.e., the proposed algorithms in [14], follow similar

Algorithm 1 Greedy algorithm for problem in eq. (1.1) [12].

Input: Per problem in eq. (1.1), Algorithm 19 receives the inputs:

- a matroid $(\mathcal{V}, \mathcal{I})$;
- a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$.

Output: Set $\mathcal{A}$.

1: $\mathcal{A} \leftarrow \emptyset$; $\mathcal{R} \leftarrow \emptyset$;
2: **while** $\mathcal{R} \neq \mathcal{V}$ **do**
3: $x \in \arg\max_{y \in \mathcal{V} \setminus (\mathcal{A} \cup \mathcal{R})} f(\mathcal{A} \cup \{y\})$;
4: **if** $\mathcal{A} \cup \{x\} \in \mathcal{I}$ **then**
5: $\mathcal{A} \leftarrow \mathcal{A} \cup \{x\}$;
6: **end if**
7: $\mathcal{R} \leftarrow \mathcal{R} \cup \{x\}$;
8: **end while**

steps to the ones in Algorithm 1, and differ only on how they choose which element to add in $\mathcal{A}$ (i.e., they replace the criterion in line 3 of Algorithm 1 with some other).

Notably, the algorithms in [12, 14] are proved to be near-optimal for several instances of the optimization problem in eq. (1.1) [13, 14, 15], and are commonly used in, e.g., statistics, such as, in machine learning [16], and optimization, such as, in facility location [17].

1.3. Need for novel approaches of submodular maximization in control

However, the algorithms in [12, 14], cannot address with provable approximation performance the fundamental control problems of actuator selection discussed above, such as the ones in [4]. In particular, consider the fundamental problem of actuator placement for low control effort in [4], where the objective is to place a few actuators in a dynamical system to minimize the average control effort one needs to drive the system in the state space. In this case, the algorithms in [12, 14] do not exhibit the near-optimal approximation proved in [12, 15], even for average control-effort metrics (which are instances of the objective function f in eq. (1.1)) that are non-decreasing and submodular. To reveal the reason, we next discuss in more detail when a system can be controlled with low effort from an actuator set, and then discuss how the algorithms in [12, 14] may become insufficient in this context: specifically, the control-effort one needs to drive a system in the state space is *infinite* if the system is *not* controllable from the set of placed actuators, i.e., if there exists at least one system state that is not reachable with a finite amount of control effort from the set of placed actuators. In particular, for a system to be controllable typically more than one actuators is needed [18]. Hence, given a system, and *any* set of placed actuators of low enough cardinality, then any metric f that captures the average control effort needed to drive the system in the state space [19] is *infinity* (has infinite value). The latter conclusion is sufficient to reveal why the algorithms in [12, 14] may fail to provide a near-optimal actuator selection for low-effort control: by focusing without loss of generality only on Algorithm 1, recall that Algorithm 1 builds an approximate solution to the optimization problem in eq. (1.1) greedily, by starting with an empty set $\mathcal{A}$, and then by adding elements in $\mathcal{A}$ one-by-one, using the criterion in line 5 of Algorithm 1 to differentiate among the candidate elements to

add; however, since the control effort metrics are infinity insofar only a few actuators have been added in $\mathcal{A}$, Algorithm 1 cannot differentiate among them, and as a result, it picks randomly the element to add in $\mathcal{A}$. The complication of this fact is that even though all the elements finally picked in $\mathcal{A}$ affect the average control effort, a part of $\mathcal{A}$ has been picked randomly, instead for minimizing the average control effort.

In sum, we have exemplified the necessity for novel tools of submodular maximization in control, by presenting the above complications in applying the state-of-the-art algorithms in [12, 14] to the fundamental problem of actuator placement for low control effort.

1.4. Need for novel approaches of submodular maximization in sensing and robotics

Traditional designs in sensing focus on selecting sensors in critical infrastructures, such as in networks of satellites, or in power-grids, with the objective to enable state estimation via Kalman filtering in the presence of resource constraints, such as of limited bandwidth for simultaneous satellite sensor communication [20], or of limited monetary budget for phasor-measurement-unit (PMU) placement in power grids [4].

However, recent advances in the miniaturization of sensors and robots trigger the vision of using swarms of mobile robots to support missions of search and rescue, and of safety and security [2], which all suggest a shift of focus in the sensor selection process beyond Kalman filtering: in particular, a shift from sensor selection for merely state estimation (Kalman filtering) to sensor selection for autonomous navigation. For example, for a swarm of robots to participate in missions of search and rescue in burning buildings, where each robot in the swarm can operate only a subset of its sensors due to limited battery, the primary goal of the sensor selection process is to enable the swarm's capability for autonomous navigation, instead of its capability for only localization (state estimation); that is, such missions of autonomous navigation exemplify the need for *navigation-aware* sensor selection emerges, instead of merely *localization-aware* sensor selection.

At the same time, emergent medical applications require the design of multi-sensor devices, such as of brain wearables, that enable *smoothing estimation* (trajectory estimation), instead of Kalman filtering (state estimation); see [21] and the references therein. Similarly, research in robotics weigh also on *smoothing estimation* to enable exploration missions in unknown environments by the means of simultaneous localization and mapping (SLAM) [22].

In sum, novel sensor selection schemes of submodular maximization are necessitated, that go beyond Kalman filtering to enable a variety of critical applications such as medical applications of brain wearables, and autonomous navigation applications of swarms of robots.

1.5. Need for resilient submodular maximization

At the same time, in all the above critical infrastructures and complex autonomous systems, actuators can fail [23]; sensors and robots can get attacked [24]. Hence, in such failure-prone and adversarial scenarios, *resilient* designs against denial-of-service attacks or failures become important. That is, one needs to introduce *resilient re-formulations* of the problem in eq. (1.1), that go beyond the traditional problem in eq. (1.1), and guard against denial-

of-service attacks and failures, either in an *off-line fashion* (before any attack or failure happens) or in an *on-line fashion* (while any attacks or failures happen).

Evidently, which of the two options is appropriate —off-line or on-line resilient design— depends on the context of the design in hand. For example, off-line protection of designs becomes important in critical infrastructures, such as in power grids, where the design happens once, does not change in time, and needs to withstand future attacks or failures [1, 25]. In contrast, on-line protection of designs becomes important in critical tasks where the design requirements may evolve in time, such as in sensor scheduling for autonomous navigation in search and rescue, where, specifically, different sensors are activated at each time step, and as a result, different sensors may fail or get attacked at each time step.

We discuss in more detail the two options of off-line and on-line resilient design below.

1.5.1. Off-line resilient submodular maximization

An option for an off-line resilient re-formulation of the problem in eq. (1.1) is the following:

$$\max_{\mathcal{A}\subseteq\mathcal{V},\, \mathcal{A}\in\mathcal{I}} \quad \min_{\mathcal{B}\subseteq\mathcal{A},\, \mathcal{B}\in\mathcal{I}'} \quad f(\mathcal{A}\setminus\mathcal{B}). \tag{1.2}$$

where the set $\mathcal{I}'$ represents the collection of possible set-removals $\mathcal{B}$ —attacks or failures— from $\mathcal{A}$, each of some specified cardinality. Hence, the problem in eq. (1.2) maximizes f despite *worst-case* failures that compromise the maximization in eq. (1.1). Therefore, it is suitable in scenarios where there is no prior on the removal mechanism, as well as, in scenarios where protection against worst-case removals is essential, such as in sensor selections for expensive experiment designs.

Particularly, the optimization problem in eq. (1.2) may be interpreted as a 2-stage perfect information sequential game between two players [26, Chapter 4], namely, a "maximization" player (designer), and a "minimization" player (attacker), where the designer plays first, and selects $\mathcal{A}$ to maximize the objective function f, and, in contrast, the attacker plays second, and selects $\mathcal{B}$ to minimize the objective function f. In particular, *the attacker first observes the designer's selection* $\mathcal{A}$, and then, selects $\mathcal{B}$ such that $\mathcal{B}$ is a worst-case set removal from $\mathcal{A}$.

1.5.2. On-line resilient submodular maximization

As mentioned above, the optimization problem in eq. (1.2) enables the off-line protection of system designs against attacks or failures (since in eq. (1.2) the set $\mathcal{A}$ is selected once, and before any attack or failure $\mathcal{B}$ happens); however, for design requirements that evolve in time (such as in sensor scheduling), one may want to go even beyond the off-line resilient objective of the problem in eq. (1.2), and guard *adaptively* against real-time attacks or failures. To this end, an option is to introduce the following on-line re-formulation of the problem in

eq. (1.2) (which for simplicity is presented for the case of merely cardinality constraints):

$$\begin{aligned} &\max_{\mathcal{A}_1 \subseteq \mathcal{V}_1} \min_{\mathcal{B}_1 \subseteq \mathcal{A}_1} \cdots \max_{\mathcal{A}_T \subseteq \mathcal{V}_T} \min_{\mathcal{B}_T \subseteq \mathcal{A}_T} f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_T \setminus \mathcal{B}_T), \\ &\quad \text{such that:} \\ &\quad |\mathcal{A}_t| = \alpha_t \text{ and } |\mathcal{B}_t| \leq \beta_t, \text{ for all } t = 1, \ldots, T, \end{aligned} \tag{1.3}$$

where the number β_t is the number of possible attacks or failures. Hence, the problem in eq. (1.2) maximizes the function f despite real-time *worst-case* failures that compromise the consecutive maximization steps in eq. (1.1). Therefore, similarly to the problem in eq. (1.2), it is suitable in scenarios where there is no prior on the removal mechanism, and in scenarios where protection against worst-case failures is essential, such as in missions of adversarial-target tracking.

Particularly, and similarly to the problem in eq. (1.2), the problem in eq. (1.3) may be interpreted as a T-stage perfect information sequential game between two players [26, Chapter 4], namely, a "maximization" player (designer), and a "minimization" player (attacker), who play sequentially, *both observing all past actions of all players*, and with the designer starting the game. That is, at each time $t = 1, \ldots, T$, both the designer and the attacker *adapt* their set selections to the history of all the players' selections so far, and, in particular, the attacker adapts its selection also to the current (t-th) selection of the designer (since at each step t, the attacker plays after it observes the selection of the designer).

1.6. Thesis goal and approach

Goal. The goal of the thesis is threefold:

- (*Novel theory on submodular maximization*) To address fundamental design problems in control, sensing, and robotics per the problem in eq. (1.1); in particular:
  - (*Control*) We consider two fundamental problems of actuator placement: the problem of *actuator placement for state reachability*, and the problem of *actuator placement for controllability with low control effort*. These problems are important, e.g., in the stabilizability of large-scale systems, such as power grids [27], and the control of complex networks, such as biological networks [28].

    In particular, the objective of *actuator placement for state reachability* is to determine which few nodes we should actuate in a linear dynamical system so to make feasible the state transfer from the system's initial condition to a given final state. And the objective of *actuator placement for controllability with low control effort* is to determine which few nodes we should actuate in a linear dynamical system so to maximize the volume of the system states that are reachable with one unit of control effort from the system's initial condition.
  - (*Sensing* and *robotics*) We consider two fundamental problems of sensor selection: the problem of *sensor selection for batch-state estimation* (smoothing), and the problem of *sensor selection for LQG control* (autonomous navigation). These problems are important in both sensing and robotics applications (see also Sec-

tion 1.4), such as in the design of brain wearables in medical applications [21], and in the design of the control inputs in multi-robot navigation applications [29].

In particular, the objective of *sensor selection for batch-state estimation* is to determine which few sensors we should activate in a linear dynamical system — possibly different sensors at different time steps— so to maximize at each time step the estimation accuracy of the system's observed trajectory so far. And the objective of *sensor selection for LQG control* is to determine which few sensors we should activate in a linear system so to enable the generation of control inputs that minimize the system's deviation from a desired trajectory.

- (*Novel theory on resilient maximization*) To protect against attacks and failures not only the aforementioned fundamental designs, *but also to go beyond control, sensing, and robotics*, and protect *any* design per the problem in eq. (1.1) —e.g., in machine learning, facility location, and optimization in general [16, 17, 30]— by introducing the resilient re-formulation of eq. (1.1) per the eq. (1.2) or the eq. (1.3); in particular:

  - (*Off-line resilient maximization*) The problem in eq. (1.2) goes beyond traditional (non-resilient) optimization [12, 13, 31, 32, 33] by proposing *resilient* optimization; beyond merely cardinality-constrained resilient optimization [34, 35] by proposing *matroid-constrained* resilient optimization; and beyond protection against non-adversarial set-removals [36, 37] by proposing protection against *worst-case* set-removals. Hence, the problem in eq. (1.2) aims to protect the complex design of systems, per *heterogeneity* or *global-interdependency* constraints, against attacks or failures, which is a vital objective for the safety of critical infrastructures, such as power grids [1, 25], or internet service provider networks [38].

  - (*On-line resilient maximization*) The problem in eq. (1.3) goes beyond traditional (non-resilient) optimization [31, 32, 33, 39, 40] by proposing *resilient* optimization; beyond the single-step resilient optimization in [34] or in eq. (1.2) by proposing *multi-step* (sequential) resilient optimization; beyond memoryless resilient optimization [41] by proposing *adaptive* resilient optimization; and beyond protection against non-adversarial attacks [36, 37] by proposing protection against *worst-case* attacks. Hence, the problem in eq. (1.3) aims to protect the system performance over extended periods of time against real-time denial-of-service attacks or failures, which is vital in critical applications, such as multi-target surveillance with teams of mobile robots [6].

- (*Applications of resilient maximization*) To apply the resilient maximization tools we develop herein to the problem of *active information gathering with mobile robots* [42].

  In particular, active information gathering calls for the motion-design of a team of mobile robots so to enable the effective information gathering about a process of interest. For example, this problem aims to support critical missions such as:

  - *Hazardous environmental monitoring*: Deploy a team of mobile robots *to monitor* the radiation flow around a nuclear reactor after an explosion; [43]

- *Adversarial-target tracking*: Deploy a team of agile robots *to track* an adversarial target that aims to escape by moving in a cluttered urban environment; [3]

- *Search and rescue*: Deploy a team of aerial micro-robots *to localize* people trapped in a burning building. [2]

Approach. To achieve the above ends, in this thesis we develop novel algorithms for both submodular and merely monotone maximization, as explained in more detail below.

1.7. Thesis contributions, and organization

The thesis contribution is to realize the aforementioned goals, by developing novel algorithms for both submodular and monotone maximization, that achieve the following characteristics:

- *resiliency*: where applicable, the algorithms are valid for any number of denial-of-service attacks or failures;

- *adaptiveness*: where applicable, at each time step the algorithms select system elements based on both the history of selections, and on the history of attacks or failures;

- *provable approximation guarantees*: the algorithms guarantee for any submodular or merely monotone function a solution close to the optimal;

- *minimal running time*: the algorithms terminate with the same running time as state-of-the-art algorithms for submodular maximization.

In more detail, the thesis contributions per thesis chapter are as follows:

- (Chapters 2-3) *Contributions to submodular maximization in control*: In Chapter 2 and Chapter 3 we address the problems of minimal actuator placement for state reachability and of minimal actuator placement for controllability, respectively.

  In more detail, in Chapters 2-3 we make the following contributions:

  - In Chapter 2 we prove that the problem of actuator placement for state reachability *cannot* be approximated in polynomial or even quasi-polynomial time.

  - In Chapter 3 we prove that the problem of minimal actuator placement for controllability with low control effort is NP-hard, yet we provide novel and near-optimal approximation algorithms for its solution, by overcoming the complications discussed in Section 1.3 regarding the application of state-of-the-art algorithms for the solution of the submodular maximization problem in eq. (1.1).

- (Chapters 4-8) *Contributions to submodular maximization in sensing and robotics*: In Chapters 4-7 we focus on the problem of sensor selection for batch-state estimation, and in Chapter 8 we focus on the problem of sensor selection for LQG control.

  In more detail, in Chapters 4-7 we make the following contributions:

– (*Problem definition*) We formalize problems of sensor selection for batch-state estimation (smoothing) for systems that are either linear (Chapters 4-5), non-linear (Chapter 6), or stochastic (Chapter 7). This is the first work to formalize, address, and demonstrate the importance of these problems.

– (*Solution*) We prove that the problem of sensor selection for batch-state estimation is NP-hard (Chapter 6), yet we provide for its solution near-optimal, on-line approximation algorithms, with minimal running time (equal to those sensor selection algorithms that are employed for Kalman filtering).

– (*Application*) We propose novel designs of multi-sensor brain wearables that rely on electroencephalograms, by determining via our proposed algorithms the sensor location that seems to be the most effective with respect to a pre-specified number of sensors. In particular, we observe that for a variety of tasks the location of sensors currently used in such wearable devices is sub-optimal with respect to the objective smoothing estimation (Chapter 6).

Finally, in Chapter 8 we make the following contributions:

– (*Problem definition*) We formalize the problem of sensor selection for LQG control, in particular, subject to heterogeneous sensor-cost constraints. This is the first work to formalize, address, and demonstrate the importance of this problem.

– (*Solution*) We provide the first algorithms the problem of sensor selection for LQG control, by extending algorithms in the literature on submodular optimization subject to heterogeneous cost constraints. In particular, (i) we provide the first efficient algorithms for the optimization of approximately supermodular functions subject to heterogeneous-cost constraints; and (ii) we improve known sub-optimality bounds that also apply to the optimization of (exactly) supermodular functions: specifically, the proposed algorithm for approximate supermodular optimization with heterogeneous-cost constraints can achieve in the exactly supermodular case the approximation bound $(1 - 1/e)$, which is superior to the previously established bound $1/2(1 - 1/e)$ in the literature [44].

– (*Simulations*) We consider two application scenarios, namely, *sensing-constrained formation control* and *resource-constrained robot navigation*. We present a Monte Carlo analysis for both scenarios, which demonstrates that (i) the proposed algorithm is near-optimal (matches the optimal selection in all tested instances for which the optimal selection could be computed via a brute-force approach), and (ii) a naive selection which attempts to minimize the state estimation covariance [5] (Kalman filtering error rather than the LQG cost) has degraded LQG tracking performance, often comparable to a random selection.

- (Chapters 9-10) *Resilient submodular maximization*: In Chapters 9-10 we go beyond the traditional objective of the optimization problem in eq. (1.1), and introduce its resilient re-formulations in eq. (1.2) and eq. (1.3), so to enable the protection of any system design per eq. (1.1) —e.g., in control, machine learning, and optimization in

general— against any number of attacks or failures.

In more detail, in Chapters 9-10 we make the following contributions:

- (*Problem definition*) We formalize the problems of *off-line resilient maximization over matroid-constraints* per eq. (1.2) (Chapter 9), and of *on-line resilient maximization* per eq. (1.3) (Chapter 10). This is the first work to formalize, address, and demonstrate the importance of these problems.

- (*Solution*) We develop the first algorithms for the solution of the resilient maximization problems in eq. (1.2) and eq. (1.3), and prove that they exhibit the properties described in the beginning of Section 1.7, i.e., the properties of *resiliency*, *adaptiveness* —applicable to the Algorithm in Chapter 10,— *provable approximation performance*, and *minimal running time*.

- (*Simulations*) We demonstrate the necessity for the resilient re-formulation of the problem in eq. (1.1) by conducting numerical experiments in various scenarios of sensing-constrained autonomous robot navigation, varying the number of sensor failures. In addition, via the experiments we demonstrate the benefits of our approach per the resilient problem formulations in eq. (1.2) and eq. (1.3).

- (Chapter 11) *Application of resilient submodular maximization to robotics*: In Chapter 11 we introduce the problem of *resilient active information gathering with mobile robots*, which goes beyond the traditional objective of (non-resilient) active information gathering, and aims to guard the information gathering process from worst-case failures or attacks that can cause not only the withdrawal of robots from the information gathering task, but also the inability of the remaining robots to jointly optimize their motions, due to disruptions to their communication network.

  In more detail, in Chapter 11 we make the following contributions:

  - (*Problem definition*) We formalize the problem of *resilient active information gathering with mobile robots* against attacks or failures. This is the first work to formalize, address, and demonstrate the importance of this problem.

  - (*Solution*) We develop the first algorithm for resilient active information gathering with the following properties:

    * *resiliency*: it is valid for any number of denial-of-service attacks or failures;

    * *provable approximation performance*: for all monotone and (possibly) submodular information gathering objective functions in the active robot set (non-failed robots), it ensures a solution close to the optimal;

    * *minimal communication*: it terminates within the same order of communication rounds as current algorithms for (non-resilient) information gathering.

  - (*Simulations*) We conduct simulations in a variety of multi-robot multi-target

tracking scenarios, varying the number of robots, targets, and failures. Our simulations validate the benefits of our approach.

– (*Experiments*) We conduct hardware experiments of multiple quad-rotors tracking static ground targets, to demonstrate visually the necessity for resilient robot motion design against robotic failures or denial-of-service attacks.

Part I

CONTRIBUTIONS TO SUBMODULAR MAXIMIZATION IN ACTUATION DESIGN

CHAPTER 2 : Minimal Reachability is Hard to Approximate

In this chapter, we consider the problem of choosing which nodes of a linear dynamical system should be actuated so that the state transfer from the system's initial condition to a given final state is possible. Assuming a standard complexity hypothesis, we show that this problem cannot be efficiently solved or approximated in polynomial, or even quasi-polynomial, time.[1]

2.1. Introduction

During the last decade, researchers in systems, optimization, and control have focused on questions such as:

- *(Actuator Selection)* How many nodes do we need to actuate in a gene regulatory network to control its dynamics? [46, 47]
- *(Input Selection)* How many inputs are needed to drive the nodes of a power system to fully control its dynamics? [48]
- *(Leader Selection)* Which UAVs do we need to choose in a multi-UAV system as leaders for the system to complete a surveillance task despite communication noise? [49, 50]

The effort to answer such questions has resulted in numerous papers on topics such as actuator placement for controllability [7, 51]; actuator selection and scheduling for bounded control effort [18, 52, 53, 54]; resilient actuator placement against failures and attacks [55, 56]; and sensor selection for target tracking and optimal Kalman filtering [57, 58, 59, 60]. In all these papers the underlying optimization problems have been proven (i) either polynomially-time solvable [46, 47, 48] (ii) or NP-hard, in which case polynomial-time algorithms have been proposed for their approximate solution [7, 18, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60].

But in several applications in systems, optimization, and control, such as in power systems [61, 62], transportation networks [63], and neural circuits [64, 65], the following problem also arises:

> **Minimal Reachability Problem.** Given times t_0 and t_1 such that $t_1 > t_0$, vectors x_0 and x_1, and a linear dynamical system with state vector $x(t)$ such that $x(t_0) = x_0$, find the minimal number of system nodes we need to actuate so that the state transfer from $x(t_0) = x_0$ to $x(t_1) = x_1$ is feasible.

For example, the stability of power systems is ensured by placing a few generators such that the state transfers from a set of possible initial conditions to the zero state are feasible [62].

The minimal reachability problem relaxes the objectives of the applications in [7, 18, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60]. For example, in comparison to the actuator placement problem for controllability [7], the minimal reachability problem aims to place a few actuators only to make a single transfer between two states feasible, whereas the

[1] This chapter is based on the paper by A. Jadbabaie, A. Olshevsky, G. J. Pappas, and V. Tzoumas [45].

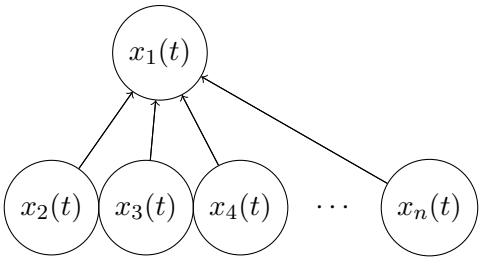


Figure 1: Graphical representation of the linear system $\dot{x}_1(t) = \sum_{j=2}^{n} x_j(t)$, $\dot{x}_i(t) = 0$, $i = 2, \ldots, n$; each node represents an entry of the system's state $(x_1(t), x_2(t), \ldots, x_n(t))$, where t represents time; the edges denote that the evolution in time of x_1 depends on $(x_2, x_3, \ldots, x_n)$.

minimal controllability problem aims to place a few actuators to make the transfer among any two states feasible [7, 51].

The fact that the minimal reachability problem relaxes the objectives of the papers [7, 18, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60] is an important distinction whenever we are interested in the feasibility of only a few state transfers by a small number of placed actuators. The reason is that under the objective of minimal reachability the number of placed actuators can be much smaller in comparison to the number of placed actuators under the objective of controllability. For example, in the system of Fig. 1 the number of placed actuators under the objective of minimal reachability from $(0, \ldots, 0)$ to $(1, \ldots, 0)$ is one, whereas the number of placed actuators under the objective of controllability grows linearly with the system's size.

The minimal reachability problem was introduced in [66], where it was found to be NP-hard. Similar versions of the reachability problem were studied in the context of power systems in [62] and [67]. For the polynomial-time solution of the reachability problems in [62, 66, 67], greedy approximation algorithms were proposed therein. The approximation performance of these algorithms was claimed by relying on the modularity result [68, Lemma 8.1], which states that the distance from a point to a subspace created by the span of a set of vectors is supermodular in the choice of the vectors.

In this chapter, we first show that the modularity result [68, Lemma 8.1] is incorrect. In particular, we show this via a counterexample to [68, Lemma 8.1], and as a result, we prove that the distance from a point to a subspace created by the span of a set of vectors is non-supermodular in the choice of the vectors. Then, we also prove the following strong intractability result for the minimal reachability problem, which is our main contribution in this chapter:

> **Contribution 1.** Assuming NP $\notin$ BPTIME($n^{\text{poly log n}}$), we show that for each $\delta > 0$, there is no polynomial-time algorithm that can distinguish between the two cases where:
>
> – the reachability problem has a solution with cardinality k;

– the reachability problem has no solution with cardinality $k2^{\Omega\left(\log^{1-\delta} n\right)}$, where n is the dimension of the system.

We note that the complexity hypothesis NP $\notin$ BPTIME($n^{\text{poly log n}}$) means there is no randomized algorithm which, after running for $O(n^{(\log n)^c})$ time for some constant c, outputs correct solutions to problems in NP with probability 2/3; see [69] for more details.

Notably, Contribution 1 remains true even if we allow the algorithm to search for an approximate solution that is relaxed as follows: instead of choosing the actuators to make the state transfer from the initial state x_0 to a given final state x_1 possible, some other state $\widehat{x}_1$ that satisfies $\|x_1 - \widehat{x}_1\|_2^2 \leq \epsilon$ should be reachable from x_0. This is a substantial relaxation of the reachability problem's objective, and yet, we show that the intractability result of Contribution 1 still holds.

The rest of this chapter is organized as follows. In Section 2.2, we introduce formally the minimal reachability problem. In Section 2.3, we provide a counterexample to [68, Lemma 8.1]. In Section 2.4, we present Contribution 1; in Section 2.5, we prove it. Section 2.6 concludes the chapter.

2.2. Minimal Reachability Problem

In this section we formalize the minimal reachability problem. We start by introducing the systems considered in this chapter and the notions of system node and of actuated node set.

System 1. *We consider continuous-time linear systems of the form*

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad t \geq t_0, \tag{2.1}$$

where t_0 is a given starting time, $x(t) \in \mathbb{R}^n$ is the system's state at time t, and $u(t) \in \mathbb{R}^m$ is the system's input vector. ◀

In this chapter we want to actuate the minimal number of the system's nodes in eq. (2.1) to make a desired state-transfer feasible (and not to achieve necessarily the system's controllability). We formalize this control objective using the following two definitions.

Definition 1 (System node)**.** *Given a system as in eq. (2.1), where $x(t) \in \mathbb{R}^n$, let $x_1(t)$, $x_2(t), \ldots, x_n(t) \in \mathbb{R}$ such that $x(t) = (x_1(t), x_2(t), \ldots, x_n(t))$. We refer to each $x_i(t)$ as a* system node. ◀

Definition 2 (Actuated node set)**.** *Given a system as in eq. (2.1), where $x(t) \in \mathbb{R}^n$, we say that the set $\mathcal{S} \in \{1, 2, \ldots, n\}$ is an* actuated node set *if for all times t the input $u(t)$ affects only the system nodes $x_i(t)$ where $i \in \mathcal{S}$. Formally, the set $\mathcal{S} \in \{1, 2, \ldots, n\}$ is an* actuated node set *if the system dynamics are given by*

$$\dot{x}(t) = Ax(t) + \mathbb{I}(\mathcal{S})Bu(t), \qquad t \geq t_0, \tag{2.2}$$

where $\mathbb{I}(\mathcal{S})$ is a $n \times n$ diagonal matrix such that if $i \in \mathcal{S}$, the i-th entry of $\mathbb{I}(\mathcal{S})$'s diagonal is 1, otherwise it is 0. ◀

The definition of $\mathbb{I}(\mathcal{S})$ in eq. (2.2) implies that the input $u(t)$ affects only those system nodes $x_i(t)$ where $i \in \mathcal{S}$. In more detail,

- if $i \in \mathcal{S}$, the system node $x_i(t)$ is affected by $u(t)$, since for $i \in \mathcal{S}$ the i-th row of $\mathbb{I}(\mathcal{S})B$ is the i-th row of B;
- if $i \notin \mathcal{S}$, the system node $x_i(t)$ cannot be affected by $u(t)$, since for $i \notin \mathcal{S}$ the i-th row of $\mathbb{I}(\mathcal{S})B$ is zero.

Overall, the set $\mathcal{S}$ determines via the matrix $\mathbb{I}(\mathcal{S})B$ which rows of B will be set to zero and which will remain the same.

Problem 1 (Minimal Reachability)**.** *Given*

- *times* t_0 *and* t_1 *such that* $t_1 > t_0$*,*
- *vectors* $x_0,\ x_1 \in \mathbb{R}^n$*; and*
- *a system* $\dot{x}(t) = Ax(t)+Bu(t),\ t \geq t_0$*, as in eq.* (2.1)*, with initial condition* $x(t_0) = x_0$*,*

find an actuated node set with minimal cardinality such that there exists an input $u(t)$ *defined over the time interval* (t_0, t_1) *that achieves* $x(t_1) = x_1$*. Formally, using the notation* $|\mathcal{S}|$ *to denote the cardinality of a set* $\mathcal{S}$*:*

$$\underset{\mathcal{S}\subseteq\{1,2,\ldots,n\}}{\text{minimize}} \quad |\mathcal{S}|$$

such that there exist $u : (t_0, t_1) \mapsto \mathbb{R}^m,\ x : (t_0, t_1) \mapsto \mathbb{R}^n$ *with*

$$\begin{aligned} \dot{x}(t) = Ax(t) + \mathbb{I}(\mathcal{S})Bu(t), \qquad t \geq t_0, \\ x(t_0) = x_0, \quad x(t_1) = x_1. \end{aligned}$$

A special case of particular interest is when B is the identity matrix. Then, minimal reachability asks for the fewest system nodes that need to be directly actuated by an input $u(t)$ so that at time t_1 the state x_1 is reachable from the system's initial condition $x(t_0) = x_0$.

2.3. Non-supermodularity of distance from point to subspace

In this section, we provide a counterexample to the supermodularity result [68, Lemma 8.1]. We begin with some notation. In particular, given a matrix $M \in \mathbb{R}^{n\times n}$, a vector $v \in \mathbb{R}^n$, and a set $\mathcal{S} \subset \{1, \ldots, n\}$, let $M(\mathcal{S})$ denote the matrix by throwing away columns of M not in $\mathcal{S}$. In addition, for any set $\mathcal{S} \subset \{1, \ldots, n\}$, let the set function

$$f(\mathcal{S}) = \text{dist}^2(v, \text{Range}(M(\mathcal{S}))),$$

where $\text{dist}(y, X)$ is the distance from a point to a subspace; formally,

$$\text{dist}(y, X) = \min_{x\in X} ||y - x||_2.$$

We show that there exist v and M such that the function:

$$f : \{1, 2, \ldots, n\} \mapsto \text{dist}^2(v, \text{Range}(M(\mathcal{S}))),$$

is non-supermodular. We start with the definitions of monotone and supermodular set functions.

Notation. For any set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ on a ground set $\mathcal{V}$, and any element $x \in \mathcal{V}$, $f(x)$ denotes $f(\{x\})$. ◀

Definition 3 (Monotonicity)**.** *Consider any finite set $\mathcal{V}$. The set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is non-decreasing if and only if for any $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{V}$, we have $f(\mathcal{A}) \leq f(\mathcal{A}')$.* ◀

In words, a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is non-decreasing if and only if adding elements in any set $\mathcal{A} \subseteq \mathcal{V}$ cannot decrease the value of $f(\mathcal{A})$.

Definition 4 (Supermodularity [70, Proposition 2.1])**.** *Consider any finite set $\mathcal{V}$. The set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is supermodular if and only if for any $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{V}$ and $x \in \mathcal{V}$,*

$$f(\mathcal{A}) - f(\mathcal{A} \cup \{x\}) \geq f(\mathcal{A}') - f(\mathcal{A}' \cup \{x\}).$$ ◀

In words, a function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is supermodular if and only if it satisfies the following diminishing returns property: for any $x \in \mathcal{V}$, the decrease $f(\mathcal{A}) - f(\mathcal{A} \cup \{x\})$ diminishes as $\mathcal{A}$ grows; equivalently, for any $\mathcal{A} \subseteq \mathcal{V}$ and $x \in \mathcal{V}$, $f(\mathcal{A}) - f(\mathcal{A} \cup \{x\})$ is non-increasing.

Example 1. *We show that for*

$$v = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}, \qquad M = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$f : \{1, 2, \ldots, n\} \mapsto \mathrm{dist}^2(v, \mathrm{Range}(M(\mathcal{S})))$ *is non-supermodular.*

Since v is orthogonal to the first and third columns of M,

$$\begin{aligned} f(\{1\}) = \mathrm{dist}^2(v, M(\{1\})) &= ||v||_2^2 \\ f(\{1,3\}) = \mathrm{dist}^2(v, M(\{1,3\})) &= ||v||_2^2 \end{aligned}$$

Therefore,

$$f(\{1\}) - f(\{1,3\}) = 0.$$

At the same time, the span of the first two columns of M is the subspace $\{x \in \mathbb{R}^3 : x_3 = 0\}$. Thus,

$$f(\{1,2\}) = \mathrm{dist}^2(v, M(\{1,2\})) = 1.$$

Moreover, since the three columns of A are linearly independent,

$$f(\{1,2,3\}) = \mathrm{dist}^2(v, M(\{1,2,3\})) = 0,$$

and as a result,

$$f(\{1,2\}) - f(\{1,2,3\}) = 1.$$

In sum,

$$f(\{1,2\}) - f(\{1,2,3\}) > f(\{1\}) - f(\{1,3\});$$

hence, for v and M as defined in this example, $f : \{1, 2, \ldots, n\} \mapsto \mathrm{dist}^2(v, \mathrm{Range}(M(\mathcal{S})))$ is

non-supermodular. ◀

We remark that the same argument as in Example 1 shows that the set function $g : \{1, 2, \ldots, n\} \mapsto \mathbb{R}$ such that $g(\mathcal{S}) = [\text{dist}(v, \text{Range}(M(\mathcal{S}))]^c$ is not supermodular for any $c > 0$.

2.4. Inapproximability of Minimal Reachability Problem

We show that, subject to a widely believed conjecture in complexity theory, there is no efficient algorithm that solves, even approximately, the minimal reachability Problem 1. Towards the statement of this result, we next introduce a definition of approximability and the definition of quasi-polynomial running time.

Definition 5 (Approximability)**.** *Consider the minimal reachability Problem 1, and let the set $\mathcal{S}^\star$ to denote one of its optimal solutions. We say that an algorithm renders Problem 1* $(\Delta_1(n), \Delta_2(n))$-approximable *if it returns a set $\mathcal{S}$ such that:*

- *there is a state $\widehat{x}_1$ such that $x(t_1) = \widehat{x}_1$ and $||\widehat{x}_1 - x_1||_2 < \Delta_1(n)$;*
- *the cardinality of $\mathcal{S}$ is at most $\Delta_2(n)|\mathcal{S}^\star|$.* ◀

In other words, the notion of $(\Delta_1(n), \Delta_2(n)$-approximability allows some slack both in the quality of the reachability requirement, and in the number of actuators utilized to achieve it.

Definition 6 (Quasi-polynomial running time)**.** *An algorithm is quasi-polynomial if it runs in $2^{O(\log n)^c}$ time, where c is a constant.* ◀

We note that any polynomial-time algorithm is a quasi-polynomial time algorithm since $n^k = 2^{k \log n}$. On the other hand, a quasi-polynomial algorithm is asymptotically faster than an exponential-time algorithm (i.e., one that runs in $O(2^{n^\epsilon})$, for some $\epsilon > 0$).

We present next our main result in this chapter.

Theorem 1 (Inapproximability)**.** *There is a collection of instances of Problem 1 where*

- *the system's initial condition is $x(t_0) = 0$;*
- *the final state x_1 is of the form $[1, 1, \ldots, 1, 0, 0, \ldots, 0]^\top$;*
- *the system's input matrix is $B = I$, where I is the identity matrix,*

such that for each $\delta \in (0, 1)$, there exists some function $\Delta(n) = 2^{\Omega(\log^{1-\delta} n)}$ so that, unless NP$\in$ BPTIME($\text{n}^{\text{poly log n}}$)*, there exists no quasi-polynomial algorithm for which Problem 1 is $(\Delta(n), 2^{\Omega(\log^{1-\delta} n)})$-approximable.*

Theorem 1 says that if NP $\notin$ BPTIME($\text{n}^{\text{poly log n}}$) there is no polynomial time algorithm (or quasi-polynomial time algorithm) that can choose which entries of the system's x state to actuate so that $x(t_1)$ is even approximately close to a desired state $x_1 = [1, 1, \ldots, 1, 0, 0, \ldots, 0]^\top$ at time t_1.

To make sense of Theorem 1, first observe that we can always actuate every entry of the system's state, i.e., we can choose $\mathcal{S} = \{1, 2, \ldots, n\}$. This means every system is $(0, n)$-approximable; let us rephrase this by saying that every system is $(0, 2^{\log n})$ approximate. Theorem 1 tells us that we cannot achieve $(0, 2^{\Omega(\log^{1-\delta} n)})$-approximability *for any* $\delta > 0$. In other words, improving the guarantee of the strategy that actuates every state by just a little bit, in the sense of replacing $\delta = 0$ with some $\delta > 0$, is not possible —subject to the complexity-theoretic hypothesis NP $\notin$ BPTIME($n^{\text{poly log n}}$). Furthermore, the theorem tells us it remains impossible even if we allow ourselves some error $\Delta(n)$ in the target state, i.e., even $(\Delta(n), 2^{\Omega(\log^{1-\delta} n)})$–approximability is ruled out.

Remark 1. *In [66, Theorem 3] it is claimed that for any $\epsilon > 0$ the minimal reachability Problem 1 is $\left(\epsilon, O\left(\log \frac{n}{\epsilon}\right)\right)$-approximable, which contradicts Theorem 1. However, the proof of this claim was based on [68, Lemma 8.1], which we proved incorrect in Section 2.3.* ◀

Remark 2. *The minimal controllability problem [7] seeks to place the fewest number of actuators to make the system controllable. Theorem 1 is arguably surprising, as it was shown in [7] that the sparsest set of actuators for controllability can be approximated to a multiplicative factor of $O(\log n)$ in polynomial time. By contrast, we showed in this chapter that an almost exponentially worse approximation ratio* cannot *be achieved for minimum reachability.* ◀

2.5. Proof of Inapproximability of Minimal Reachability

In this section, we provide a proof of our main result, namely Theorem 1. We use some standard notation throughout: $\mathbf{1}_k$ is the all-ones vector in $\mathbb{R}^k$, $\mathbf{0}_k$ is the zero vector in $\mathbb{R}^k$, and e_k is the k'th standard basis vector. We next give some standard definitions related to the reachability space of a linear system.

2.5.1. Reachability Space for continuous-time linear systems

Definition 7 (Reachability space)**.** *Consider a system $\dot{x}(t) = Ax(t) + Bu(t)$ as in eq.* (2.1) *whose size is n. The* Range$([B,\ AB,\ A^2B, \ldots,\ A^{n-1}B])$ *is called the* reachability space *of $\dot{x}(t) = Ax(t) + Bu(t)$.* ◀

The reason why Definition 7 is called the reachability space is explained in the following proposition.

Proposition 1 ([71, Proof of Theorem 6.1])**.** *Consider a system as in eq.* (2.1)*, with initial condition x_0. There exists a real input $u(t)$ defined over the time interval (t_0, t_1) such that the solution of $\dot{x} = Ax + Bu,\ \ x(t_0) = x_0$ satisfies $x(t_1) = x_1$ if and only if*

$$x_1 - e^{A(t_1 - t_0)} x_0 \in \text{Range}([B,\ AB,\ A^2B, \ldots,\ A^{n-1}B]).$$

The notion of reachability space allows us to redefine the minimal reachability Problem 1 as follows.

Corollary 1. *The minimal reachability Problem 1 is equivalent to*

$$\begin{aligned} &\underset{\mathcal{S}\subseteq\{1,2,\ldots,n\}}{\text{minimize}} \ |\mathcal{S}| \\ &\text{such that } \ x_1 - e^{A(t_1-t_0)}x_0 \in \\ &\qquad\qquad \text{Range}([\mathbb{I}(\mathcal{S})B,\ A\mathbb{I}(\mathcal{S})B,\ldots,\ A^{n-1}\mathbb{I}(\mathcal{S})B]). \end{aligned}$$

Overall, Problem 1 is equivalent to picking the fewest rows of the input matrix B such that $x_1 - e^{A(t_1-t_0)}x_0$ is in the linear span of the columns of:

$$[\mathbb{I}(\mathcal{S})B,\ A\mathbb{I}(\mathcal{S})B,\ A^2\mathbb{I}(\mathcal{S})B,\ldots,\ A^{n-1}\mathbb{I}(\mathcal{S})B].$$

2.5.2. Variable Selection Problem

We show the intractability of the minimum reachability by reducing it to the *variable selection* problem, defined next.

Problem 2 (Variable Selection)**.** *Let $U \in \mathbb{R}^{m\times l}$, $z \in \mathbb{R}^m$, and let Δ be a positive number. The variable selection problem is to pick $y \in \mathbb{R}^l$ that is an optimal solution to the following optimization problem.*

$$\begin{aligned} &\underset{y\in\mathbb{R}^l}{\text{minimize}} \quad \|y\|_0 \\ &\text{such that} \quad \|Uy - z\|_2 \leq \Delta, \end{aligned}$$

where $||y||_0$ refers to the number of non-zero entries of y.

The variable selection Problem 2 is found in [72] to be inapproximable:

Theorem 2 ([72, Proposition 6])**.** *Unless* NP$\in$ BPTIME($n^{\text{poly log n}}$)*, we have that for each $\delta \in (0,1)$ there exist*

- *a function $\Delta(l) : \mathbb{N} \to \mathbb{N}$ which is $2^{\Omega(\log^{1-\delta} l)}$;*
- *a function $q_1(l) : \mathbb{N} \to \mathbb{N}$ which is in $2^{\Omega(\log^{1-\delta} l)}$ and $O(l)$;*
- *a polynomial[2] $p_1(l)$ which is $O(l)$;*
- *a polynomial $m(l)$,*

such that, given an $m(l)\times l$ matrix U, no quasi-polynomial algorithm can distinguish between the following two cases:

1. *There exists $y \in \{0,1\}^l$ such that $Uy = \mathbf{1}_{m(l)}$ and $||y||_0 \leq p_1(l)$.*
2. *For any $y \in \mathbb{R}^l$ such that $||Uy - \mathbf{1}_{m(l)}||_2^2 \leq \Delta(l)$, we have that $||y||_0 \geq p_1(l)q_1(l)$.*

[2] In this context, a function with a fractional exponent is considered to be a polynomial, e.g., $l^{1/5}$ is considered to be a polynomial in l.

Informally, for the variable selection Problem 2 in Theorem 2, unless NP$\in$BPTIME($n^{\text{poly log n}}$), there is no quasi-polynomial algorithm that can distinguish between the case where there exists a solution to Problem 2 with a few non-zero entries, and the case where every approximate solution has almost every entry nonzero.

2.5.3. Sketch of Proof of Theorem 1

We begin by sketching the intuition behind the proof of Theorem 1. Our general approach is to find instances of Problem 1 that are as hard as inapproximable instances of the variable selection Problem 2. We begin by discussing a construction that does *not* work, and then explain how to fix it.

Given the matrix U coming from a variable selection Problem 2, we first attempt to construct an instance of the minimal reachability Problem 1 where

- the system's initial condition is $x(t_0) = 0$;
- the destination state x_1 at time t_1 is of the form $[\mathbf{1}, \mathbf{0}]^\top$ (the exact dimensions of $\mathbf{1}$ and $\mathbf{0}$ are to be determined);
- the system's input matrix is $B = I$;
- the system's matrix A is

$$A = \begin{pmatrix} 0 & U \\ 0 & 0 \end{pmatrix}, \tag{2.3}$$

  where the number of zeros is large so that $A^2 = 0$.

Whereas the variable selection problem involves finding the smallest set of columns of U so that a certain vector is in their span, for the minimum reachability problem, every time we add the k-th state to the set of actuated variables $\mathcal{S}$, the reachability span expands by adding the span of the set of columns of the controllability matrix that correspond to the vector e_k being added in $\mathbb{I}(\mathcal{S})$. In particular, for the above construction, because $A^2 = 0$, when the k-th state is added to the set of actuated variables, the span of the two columns e_k and Ue_k is added to the reachability space.

In other words, with the above construction we are basically constrained to make "moves" which add columns in pairs, and we are looking for the smallest number of such "moves" making a certain vector lie in the span of the columns. It should be clear that there is a strong parallel between this and variable selection (where the columns are added one at a time). However, because the columns are being added in pairs, this attempt to connect minimum reachability with variable selection does not quite work. To fix this idea, we want only the columns of U to contribute meaningfully to the addition of the span, with any vectors e_k we add along the way being redundant; this would reduce minimal reachability

to exactly variable selection. We accomplish this by further defining,

$$U' = \begin{pmatrix} U \\ U \\ \vdots \\ U \end{pmatrix},$$

where we stack U some large number of times (to be determined in the main proof of Theorem 1 at Section 2.5.4). We then set

$$A = \begin{pmatrix} 0 & U' \\ 0 & 0 \end{pmatrix}. \tag{2.4}$$

The idea is that because U is "stacked" many times, adding a column of U to a set of vectors expands the span much more than adding any vector e_k, so there is never an "incentive" to even consider the contributions of the vectors e_k to the reachability space.

We next make this argument precise. First, given a matrix $M \in \mathbb{R}^{l \times l}$, for $n \geq kp$ we define $\phi_{n,d}(M)$ to be the $n \times n$ matrix which stacks U in the top-right hand corner d times. For example,

$$M = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad \phi_{5,2}(M) = \begin{pmatrix} 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 3 & 4 \\ 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 3 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

i.e., $\phi_{5,2}(M)$ stacks M twice, and then pads it with enough zeros to make the resulting matrix 5×5. Observe that if $n \geq 2dl$, then $\phi_{n,d}(M)^2 = 0$. We adopt the notation that the last l columns of $\phi_{n,d}(M)$ are called the *non-identity* columns, while the first $n - l$ columns are called the *identity columns*.

2.5.4. Proof of Theorem 1

We turn to the proof of Theorem 1. We adopt the definitions in the previous sections.

Proof of Theorem 1: Let U be an $l \times l$ matrix and consider solving the minimum variable selection problem with $y = \mathbf{1}$; by Theorem 2 this cannot be computed in quasi-polynomial time unless NP $\notin$ BPTIME($\mathrm{n}^{\mathrm{poly}\log \mathrm{n}}$). Adopting the notation of Theorem 2, we set:

- $d = m(l)\lceil p_1(l)q_1(l)\rceil$;
- $n = 2\max(d, l)$;
- for simplicity, we use m and $m(l)$ interchangeably.

We consider an instance of the minimal reachability where:

- the system's initial condition is $x(t_0) = 0$;

- the destination state x_1 at time t_1 is $[\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top$;
- the system's input matrix is $B = I$, where I is the identity matrix;
- the system's matrix is $A = \phi_{n,d}(U)$.

Given the above instance for Problem 1, we next prove Theorem 1 in two steps.

First step of proof: Suppose that there exists a vector $y \in \{0,1\}^l$ with $Uy = \mathbf{1}_m$ and $||y||_0 \leq p_1(l)$. In that case, we claim there exists a set $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ with $|\mathcal{S}| \leq p_1(l)$ such that $[\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top$ reachable. Indeed, let S be a set of columns of U that have $\mathbf{1}_m$ in their span, and set $\mathcal{S} = \{k + n - l \mid k \in S\}$. Then $|\mathcal{S}| \leq p_1(l)$, and

$$\mathbf{1}_m = \sum_{k \in S} U_k,$$

where U_k denotes the k'th column of the matrix U; hence, we have

$$\begin{pmatrix} \mathbf{1}_d \\ \mathbf{0}_{n-d} \end{pmatrix} = \begin{pmatrix} \mathbf{1}_m \\ \mathbf{1}_m \\ \vdots \\ \mathbf{1}_m \\ \mathbf{0}_{n-d} \end{pmatrix} = \sum_{k \in S} \begin{pmatrix} U_k \\ U_k \\ \vdots \\ U_k \\ \mathbf{0}_{n-d} \end{pmatrix} = \sum_{k \in S} A_{k+n-l},$$

where the final step follows by definition of $\phi_{n,d}(\cdot)$. Now each of the vectors in the last term is a column of $A\mathbb{I}(\mathcal{S})$ with this choice of $\mathcal{S}$, so $[\mathbf{1}_d, \mathbf{0}_{n-d}]^T$ indeed lies in the range of the controllability matrix.

Second step of proof: Conversely, suppose that any z with $||Uz - \mathbf{1}||_2^2 \leq \Delta(l)$ has the property that $||z||_0 \geq p_1(l)q_1(l)$. We refer to this as assumption A1. We claim that in this case there is no $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ with cardinality strictly less than $p_1(l)q_1(l)$ that makes any y with $||y - [\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top||_2^2 \leq \Delta(l)$ reachable. To prove this, assume the contrary, i.e., assume there exists $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ with cardinality strictly less than $p_1(l)q_1(l)$ that makes *some* y with $||y - [\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top||_2^2 \leq \Delta(l)$ reachable. We call this assumption A2. We obtain a contradiction as follows:

- Break up $\mathcal{S}$ into identity columns and non-identity columns such that $\mathcal{S} = \mathcal{S}_{\text{id}} \cup \mathcal{S}_{\text{non-id}}$.
- By the pigeonhole principle, it follows that in the set $\{1, 2, \ldots, d\}$ there is some interval $\mathcal{E} = \{\kappa m + 1, \kappa m + 2, \ldots, \kappa m + m\}$, where κ is a non-negative integer, such that $\mathcal{S} \cap V = \emptyset$, because $|\mathcal{S}| < p_1(l)q_1(l)$ and $d \geq m\lceil p_1(l)q_1(l) \rceil$.
- In particular, there is no $k \in \mathcal{S}_{\text{id}}$ such that $k \in \mathcal{E}$, since in the previous bullet point we showed $\mathcal{S} \cap \mathcal{E} = \emptyset$, and therefore $\mathcal{S}_{\text{id}} \cap \mathcal{E} = \emptyset$.
- As a consequence of the assumption that there is $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ with cardinality strictly less than $p_1(l)q_1(l)$ that makes any y with $||y - [\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top||^2 \leq \Delta(l)$

reachable, we have that there is $y \in \text{Range}[\mathbb{I}(\mathcal{S}), A\mathbb{I}(\mathcal{S}), 0, 0, \ldots, 0]$ such that $||y - [\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top||_2^2 \leq \Delta(l)$. Define $y_\mathcal{E} \in \mathbb{R}^m$ by taking the rows of y corresponding to indices in $\mathcal{E}$. Then, $||y_\mathcal{E} - \mathbf{1}_m||_2^2 \leq \Delta(l)$. Moreover, $y_\mathcal{E}$ is in the span of the vectors obtained by taking the rows $\kappa m+1, \ldots, \kappa m+m$ of the columns of the reachability matrix $[\mathbb{I}(\mathcal{S}), A\mathbb{I}(\mathcal{S}), 0, 0, \ldots, 0]$. Since in the previous bullet point we concluded $\mathcal{S}_{\text{id}} \cap \mathcal{E} = \emptyset$, all such columns are either zero or equal to a column of U.

- Thus, we have that a vector $y_\mathcal{E} \in \mathbb{R}^m$ such that $||y_\mathcal{E} - \mathbf{1}_m||_2^2 \leq \Delta(l)$ and $y_\mathcal{E}$ is in the span of $|\mathcal{S}|$ columns of U. Moreover, assumption A2 tells us that $|\mathcal{S}| < p_1(l)q_1(l)$ while assumption A1 tells us the opposite.

To summarize, we showed the dichotomy of (1a) and (1b):

1a) "There exists a vector $y \in \{0,1\}^l$ with $Uy = \mathbf{1}_m$ and $||y||_0 \leq p_1(l)$."

1b) "Any y with $||Uy - \mathbf{1}||_2^2 \leq \Delta(l)$ has the property that $||y||_0 \geq p_1(l)q_1(l)$."

implies the dichotomy of (i-a) and (i-b):

i-a) "There exists a set $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ with $|\mathcal{S}| \leq p_1(l)$ such that $[\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top$ reachable."

i-b) "There is no $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ with cardinality strictly less than $p_1(l)q_1(l)$ that makes any y with $||y - [\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]^\top||_2^2 \leq \Delta(l)$ reachable."

in the sense that (1a) implies (i-a) (first step of the proof) and (1b) implies (i-b) (second step of the proof).

Theorem 2 showed that unless NP$\in$BPTIME($\text{n}^{\text{poly log n}}$), no quasi-polynomial time algorithm can distinguish between (1a) and (1b). This implies that, under the same assumption, no quasi-polynomial time algorithm can distinguish between (i-a) and (i-b). In particular, since for any $\delta \in (0,1)$, we can take $q_1(l) = 2^{\Omega\left(\log^{1-\delta} l\right)}$ in Theorem 2, this implies that the smallest number of inputs rendering $[\mathbf{1}_d^\top, \mathbf{0}_{n-d}^\top]$ reachable cannot be approximated within a multiplicative factor of $\phi(l)$ which grows slower than $2^{\Omega\left(\log^{1-\delta} l\right)}$.

Finally, we note that because the dimension of A is polynomial in l (since A is $n \times n$, where $n = 2\max(d, l)$ with $d = m(l)\lceil p_1(l)q_1(l)\rceil$), we have that $\phi(l) = 2^{\Omega\left(\log^{1-\delta} n\right)}$. ■

2.6. Concluding Remarks & Future Work

We focused on the minimal reachability Problem 1, which is a fundamental question in optimization and control with applications such as power systems and neural circuits. By exploiting the connection to the variable selection Problem 2, we proved that Problem 1 is hard to approximate. Future work will focus on properties for the system matrix A so that Problem 1 is approximable in polynomial time.

We conclude with an open problem. As we have discussed, the minimum reachability problem is $\left(0, 2^{\log n}\right)$-approximable by the algorithm which actuates every variable; but $\left(0, 2^{\Omega\left(\log^{1-\delta} n\right)}\right)$ is impossible for any positive δ. We wonder, therefore, whether the min-

imum number of actuators can be approximated to within a multiplicative factor of say, $\sqrt{n}$ in polynomial time, or, more generally, n^c for some $c \in (0,1)$. Indeed, observe that since $\sqrt{n} = 2^{(1/2)\log n}$, the function $\sqrt{n}$ does not belong to $2^{O\left(\log^{1-\delta} n\right)}$ for any $\delta > 0$. Thus, the present chapter does not rule out the possibility of approximating the minimum reachability problem up to a factor of $\sqrt{n}$, or more broadly, n^c for $c \in (0,1)$. We remark that such an approximation guarantee would have considerable repercussions in the context of effective control, as at the moment the best polynomial-time protocol for actuation to meet a reachability goal (in terms of worst-case approximation guarantee) is to actuate every variable.

CHAPTER 3 : Minimal Actuator Placement with Bounds on Control Effort

We address the problem of minimal actuator placement in linear systems so that the volume of the set of states reachable with one unit or less of input energy is lower bounded by a desired value. First, following the recent work of Olshevsky, we prove that this is NP-hard. Then, we provide an efficient algorithm which, for a given range of problem parameters, approximates up to a multiplicative factor of $O(\log n)$, n being the network size, any optimal actuator set that meets the same energy criteria; this is the best approximation factor one can achieve in polynomial time, in the worst case. Moreover, the algorithm uses a perturbed version of the involved control energy metric, which we prove to be supermodular. Next, we focus on the related problem of cardinality-constrained actuator placement for minimum control effort, where the optimal actuator set is selected to maximize the volume of the set of states reachable with one unit or less of input energy. While this is also an NP-hard problem, we use our proposed algorithm to efficiently approximate its solutions as well.[1]

3.1. Introduction

During the past decade, an increased interest in the analysis of large-scale systems has led to a variety of studies that range from the mapping of the human's brain functional connectivity to the understanding of the collective behavior of animals, and the evolutionary mechanisms of complex ecological systems [74, 75, 76, 77]. At the same time, control scientists develop methods for the regulation of such complex systems, with the notable examples in [78], for the control of biological systems; [79], for the regulation of brain and neural networks; [80], for robust information spread over social networks, and [81], for load management in smart grid.

On the other hand, the large size of these systems, as well as the need for low cost control, has made the identification of a small fraction of their states, to steer them around the entire space, an important problem [52, 82, 83, 84]. This is a task of formidable complexity; indeed, it is shown in [82] that finding a small number of actuators, so that a linear system is controllable, is NP-hard. However, mere controllability is of little value if the required input energy for the desired transfers is exceedingly high, when, for example, the controllability matrix is close to singularity [85]. Therefore, by choosing input states to ensure controllability alone, one may not achieve a cost-effective control for the system.

In this chapter, we address this important requirement by providing efficient approximation algorithms to actuate a small fraction of a system's states so that a specified control energy performance over the entire state space is guaranteed. In particular, we first consider the selection of a minimal number of actuated states so that a pre-specified lower bound on the volume of the set of states reachable with one or less units of input energy is satisfied. Finding such a subset of states is a challenging task, since it involves the search for a small number of actuators that induce controllability, which constitutes a combinatorial problem that can be computationally intensive. Indeed, identifying a small number of actuated states for inducing controllability alone is NP-hard [82]. Therefore, we extend this computationally hard problem by introducing an energy performance requirement on the choice of the optimal

[1] This chapter is based on the paper by Tzoumas et al. [73].

actuator set, and we solve it with an efficient approximation algorithm.

Specifically, we first generalize the involved energy objective to an ϵ-close one, which remains well-defined even for actuator sets that render the system uncontrollable. Then, we make use of this metric and relax the implicit controllability constraint from the original actuator placement problem. Notwithstanding, we prove that for small values of ϵ all solutions of this auxiliary program still render the system controllable. This fact, along with the supermodularity of the generalized objective with respect to the choice of the actuator set, leads to an efficient algorithm which, for a given range of problem parameters, approximates up to a multiplicative factor of $O(\log n)$, where n is the size of the system, any optimal actuator set that meets the specified energy criterion. Moreover, this is the best approximation factor one can achieve in polynomial time, in the worst case. Hence, with this algorithm we address the open problem of minimal actuator placement subject to bounds on the control effort [52, 82, 84, 86, 87].

Relevant results are also found in [84], where the authors study the controllability of a system with respect to the smallest eigenvalue of the controllability Gramian, and they derive a lower bound on the number of actuators so that this eigenvalue is lower bounded by a fixed value. Nonetheless, they do not provide an algorithm to identify the actuators that achieve this value.

Next, we consider the problem of cardinality-constrained actuator placement for minimum control effort, where the optimal actuator set is selected so that the volume of the set of states that can be reached with one unit or less of input energy is maximized. The most related works to this problem are the [52] and [88], in which the authors assume a controllable system and consider the problem of choosing a few extra actuators in order to optimize some of the input energy metrics proposed in [19]. Their main contribution is in observing that these energy metrics are supermodular with respect to the choice of the extra actuated states. The assumption of a controllable system is necessary since these metrics depend on the inverse of the controllability Gramian, as they capture the control energy for steering the system around the entire state space. Nonetheless, it should be also clear that making a system controllable by first placing some actuators to ensure controllability alone, and then adding some extra ones to optimize a desired energy metric, introduces a sub-optimality that is carried over to the end result. In this chapter, we follow a parallel line of work to the minimal actuator placement problem, and provide an efficient algorithm that selects all the actuated states to maximize the volume of the set of states that can be reached with one unit or less of input energy without any assumptions on the controllability of the involved system.

A similar actuator placement problem is studied in [84] for stable systems. Nevertheless, its authors propose a heuristic actuator placement procedure that does not constrain the number of available actuators and does not optimize their control energy objective. Our proposed algorithm selects a cardinality-constrained actuator set that minimizes a control energy metric, even for unstable systems.

The remainder of this chapter is organized as follows. The formulation and model for the actuator placement problems are set forth in Section 3.2, where the corresponding integer

optimization programs are stated. In Sections 3.3 and 3.4 we discuss our main results, including the intractability of these problems, as well as the supermodularity of the involved control energy metrics with respect to the choice of the actuator sets. Then, we provide efficient approximation algorithms for their solution that guarantee a specified control energy performance over the entire state space. Section 3.5 concludes the chapter.

3.2. Problem Formulation

Notation. We denote the set of natural numbers $\{1, 2, \ldots\}$ as $\mathbb{N}$, the set of real numbers as $\mathbb{R}$, and we let $[n] \equiv \{1, 2, \ldots, n\}$ for all $n \in \mathbb{N}$. Also, given a set $\mathcal{X}$, we denote as $|\mathcal{X}|$ its cardinality. Matrices are represented by capital letters and vectors by lower-case letters. For a matrix A, A^T is its transpose and A_{ij} is its element located at the $i-$th row and $j-$th column. If A is positive semi-definite or positive definite, we write $A \succeq 0$ and $A \succ 0$, respectively. Moreover, for $i \in [n]$, we let $I^{(i)}$ be an $n \times n$ matrix with a single non-zero element: $I_{ii} = 1$, while $I_{jk} = 0$, for $j, k \neq i$. Furthermore, we denote as I the identity matrix, whose dimension is inferred from the context. Additionally, for $\delta \in \mathbb{R}^n$, we let $\text{diag}(\delta)$ denote an $n \times n$ diagonal matrix such that $\text{diag}(\delta)_{ii} = \delta_i$ for all $i \in [n]$. Finally, we set $\{0, 1\}^n$ to be the set of vectors in $\mathbb{R}^n$ whose elements are either zero or one.

3.2.1. Actuator Placement Model

Consider a linear system of n states, $x_1, x_2, \ldots, x_n$, whose evolution is described by

$$\dot{x}(t) = Ax(t) + Bu(t), t > t_0, \tag{3.1}$$

where $t_0 \in \mathbb{R}$ is fixed, $x \equiv \{x_1, x_2, \ldots, x_n\}$, $\dot{x}(t) \equiv dx/dt$, while u is the corresponding input vector. The matrices A and B are of appropriate dimension. We equivalently refer to (3.1) as a network of n nodes, $1, 2, \ldots, n$, which we associate with the states $x_1, x_2, \ldots, x_n$, respectively. Moreover, we denote their collection as $\mathcal{V} \equiv [n]$.

Henceforth, A is given while B is a *diagonal zero-one* matrix that we design so that (3.1) satisfies a specified control energy criterion over the entire state space.
Assumption 1. *$B = \text{diag}(\delta)$, where $\delta \in \{0, 1\}^n$.*

Specifically, if $\delta_i = 1$, state x_i may receive an input, while if $\delta_i = 0$, it receives none.
Definition 8 (Actuator Set, Actuator). *Given a $\delta \in \{0, 1\}^n$, let $\Delta \equiv \{i : i \in \mathcal{V}$ and $\delta_i = 1\}$; then, Δ is called an* actuator set *and each $i \in \Delta$ an* actuator.

3.2.2. Controllability and Related Energy Metrics

We consider the notion of controllability and relate it to the problems of this chapter, i.e., the minimal actuator placement for constrained control energy and the cardinality-constrained actuator placement for minimum control effort.

System (3.1) is controllable — equivalently, (A, B) is controllable — if for any finite $t_1 > t_0$ and any initial state $x_0 \equiv x(t_0)$ it can be steered to any other state $x_1 \equiv x(t_1)$ by some input $u(t)$ defined over $[t_0, t_1]$. Moreover, for general matrices A and B, the controllability

condition is equivalent to the matrix

$$W \equiv \int_{t_0}^{t_1} \mathrm{e}^{A(t-t_0)} B B^T \mathrm{e}^{A^T(t-t_0)} \, \mathrm{d}t, \tag{3.2}$$

being positive definite for any $t_1 > t_0$ [85]. Therefore, we refer to W as the *controllability matrix* of (3.1).

The controllability of a linear system is of interest because it is related to the solution of the following minimum-energy transfer problem

$$\begin{aligned} \underset{u(\cdot)}{\text{minimize}} \quad & \int_{t_0}^{t_1} u(t)^T u(t) \, \mathrm{d}t \\ \text{subject to} \quad & \\ & \dot{x}(t) = Ax(t) + Bu(t), t_0 < t \leq t_1, \\ & x(t_0) = 0, x(t_1) = x_1, \end{aligned} \tag{3.3}$$

where A and B are any matrices of appropriate dimension.

In particular, if for the given A and B (3.1) is controllable the resulting minimum control energy is given by

$$x_1^T W^{-1} x_1, \tag{3.4}$$

where $\tau = t_1 - t_0$ [19]. Thereby, the states that belong to the eigenspace of the smallest eigenvalues of (3.2) require higher energies of control input [85]. Extending this observation along all the directions of transfers in the state space, we infer that the closer W is to singularity the larger the expected input energy required for these transfers to be achieved [19]. For example, consider the case where W is singular, i.e., when there exists at least one direction along which system (3.1) cannot be steered [85]. Then, the corresponding minimum control energy along this direction is *infinity*.

This motivates the consideration of control energy metrics that quantify the steering energy along all the directions in the state space, as the $\log\det(W^{-1})$ [19]. Indeed, this metric is well-defined only for controllable systems — W must be invertible — and is directly related to (3.4). In more detail, $\sqrt{\det(W^{-1})}$ is inversely proportional to the volume of the set of states reachable with one or less units of input energy, i.e., the volume of $\{x : x^T W^{-1} x \leq 1\}$; as a result, when $\log\det(W^{-1})$ is minimized, the volume of $\{x : x^T W^{-1} x \leq 1\}$ is maximized. In this chapter, we aim to select a small number of actuators for system (3.1) so that $\log\det(W^{-1})$ either meets a specified upper bound or is minimized.

Per Assumption 1, further properties for the controllability matrix are due: For any actuator set Δ, let $W_\Delta \equiv W$; then,

$$W_\Delta = \sum_{i=1}^{n} \delta_i W_i, \tag{3.5}$$

where $W_i \equiv \int_{t_0}^{t_1} \mathrm{e}^{At} I^{(i)} \mathrm{e}^{A^T t} \, \mathrm{d}t$ for any $i \in [n]$. This follows from (3.2) and the fact that $BB^T = B = \sum_{i=1}^{n} \delta_i I^{(i)}$ for $B = \mathrm{diag}(\delta)$. Finally, for any $\Delta_1 \subseteq \Delta_2 \subseteq \mathcal{V}$, (3.5) and $W_1, W_2, \ldots, W_n \succeq 0$ imply $W_{\Delta_1} \preceq W_{\Delta_2}$.

3.2.3. Actuator Placement Problems

We consider the selection of a small number of actuators for system (3.1) so that $\log\det(W^{-1})$ either satisfies an upper bound or is minimized. The challenge is in doing so with as few actuators as possible. This is an important improvement over the existing literature where the goal of actuator placement problems has either been to ensure controllability alone [82] or the weaker property of structural controllability [89, 90]. Other relevant results consider the task of leader-selection [91, 92], where the leaders are the actuated states and are chosen so to minimize a mean-square convergence error of the remaining states.

Furthermore, the most relevant works to our study are the [52] and [88] since its authors consider the minimization of $\log\det(W^{-1})$; nevertheless, their results rely on a pre-existing actuator set that renders (3.1) controllable although this set is not selected for the minimization of this energy metric. One of our contributions is in achieving optimal actuator placement for minimum control effort without assuming controllability beforehand. Also, the authors of [84] adopt a similar framework for actuator placement but focus on deriving an upper bound for the smallest eigenvalue of W with respect to the number of actuators and a lower bound for the required number actuators so that this eigenvalue takes a specified value. In addition, they consider the maximization of $\mathrm{tr}(W)$; however, their techniques cannot be applied when minimizing the $\log\det(W^{-1})$, while the maximization of $\mathrm{tr}(W)$ may not ensure controllability [84].

We next provide the exact statements of our actuator placement problems, while their solution analysis follows in Sections 3.3 and 3.4. We first consider the problem

$$\begin{aligned} &\underset{\Delta \subseteq \mathcal{V}}{\text{minimize}} \quad |\Delta| \\ &\text{subject to} \\ &\qquad \log\det(W_\Delta^{-1}) \leq E, \end{aligned} \tag{I}$$

for some constant E. Its domain is $\{\Delta : \Delta \subseteq \mathcal{V} \text{ and } (A, B(\Delta)) \text{ is controllable}\}$ since the controllability matrix $W_{(\cdot)}$ must be invertible. Moreover, it is NP-hard, as we prove in Appendix 3.6.

Additionally, Problem (I) is feasible for certain values of E. In particular, for any Δ such that $(A, B(\Delta))$ is controllable, $0 \prec W_\Delta$, i.e., $\log\det(W_{\mathcal{V}}^{-1}) \leq \log\det(W_\Delta^{-1})$ since for any Δ (3.5) implies $W_\Delta \preceq W_{\mathcal{V}}$ [93]; thus, (I) is feasible for

$$E \geq \log\det(W_{\mathcal{V}}^{-1}). \tag{3.6}$$

Moreover, (I) is a generalized version of the minimal controllability problem of [82] so that its solution not only ensures controllability but also satisfies a guarantee in terms of a control energy metric; indeed, for $E \to \infty$ we recover the problem of [82].

We next consider the problem

$$\begin{aligned} \underset{\Delta\subseteq\mathcal{V}}{\text{minimize}} \quad & \log\det(W_\Delta^{-1}) \\ \text{subject to} \quad & \\ & |\Delta| \leq r, \end{aligned} \tag{II}$$

where the goal is to find at most r actuated states so that the volume of the set of states that can be reached with one unit or less of input energy is maximized. Its domain is $\{\Delta : \Delta \subseteq \mathcal{V}, |\Delta| \leq r \text{ and } (A, B(\Delta)) \text{ is controllable}\}$. Moreover, due to the NP-hardness of Problem (I), Problem (II) is also NP-hard (cf. Appendix 3.6).

Because (I) and (II) are NP-hard, we need to identify efficient approximation algorithms for their general solution; this is the subject of Sections 3.3 and 3.4. In particular, in Section 3.3 we consider Problem (I) and provide for it a best approximation algorithm, for a given range of problem parameters. To this end, we first define an auxiliary program, which ignores the controllability constraint of (I), and, nevertheless, admits an efficient approximation algorithm whose solutions not only satisfy an energy bound that is ϵ-close to the original one but also render system (3.1) controllable. Then, in Section 3.4 we turn our attention to (II), and following a parallel line of thought as for (I), we efficiently solve this problem as well.

Since the approximation algorithm for the aforementioned auxiliary program for (I) is based on results for supermodular functions, we present below a brief overview of the relevant concepts. The reader may consult [16] for a survey on these results.

3.2.4. Supermodular Functions

We give the definition of a supermodular function, as well as, a relevant result that will be used in Section 3.3 to construct an approximation algorithm for Problem (I). The material of this section is drawn from [94].

Let $\mathcal{V}$ be a finite set and denote as $2^{\mathcal{V}}$ its power set.

Definition 9 (Submodularity and supermodularity)**.** *A function $h : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* submodular *if for any sets Δ and Δ', with $\Delta \subseteq \Delta' \subseteq \mathcal{V}$, and any $a \notin \Delta'$,*

$$h(\Delta \cup \{a\}) - h(\Delta) \geq h(\Delta' \cup \{a\}) - h(\Delta').$$

A function $h : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is supermodular *if $(-h)$ is submodular.*

An alternative definition of a submodular function is based on the notion of non-increasing set functions.

Definition 10 (Monotone Set Function)**.** *A function $h : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is a* non-increasing set function *if for any $\Delta \subseteq \Delta' \subseteq \mathcal{V}$, $h(\Delta) \geq h(\Delta')$. Moreover, h is a* non-decreasing set function *if $(-h)$ is a non-increasing set function.*

Therefore, a function $h : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is submodular if, for any $a \in \mathcal{V}$, the function $h_a : 2^{\mathcal{V}\setminus\{a\}} \mapsto \mathbb{R}$, defined as $h_a(\Delta) \equiv h(\Delta \cup \{a\}) - h(\Delta)$, is a non-increasing set function. This property is

also called the *diminishing returns property*.

Next, we present a fact from the supermodular functions minimization literature, that we use in Section 3.3 so as to construct an approximation algorithm for Problem (I). In particular, consider the following optimization program, which is of similar structure to (I), where $h : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is a non-decreasing, supermodular set function:

$$\begin{aligned} &\underset{\Delta \subseteq \mathcal{V}}{\text{minimize}} \quad |\Delta| \\ &\text{subject to} \\ &\qquad h(\Delta) \leq E. \end{aligned} \tag{$\mathcal{O}$}$$

The following greedy algorithm has been proposed for its approximate solution, for which, the subsequent fact is true.

Algorithm 2 Approximation Algorithm for the Problem ($\mathcal{O}$).

Input: h, E.
Output: Approximate solution to Problem ($\mathcal{O}$).
  $\Delta \leftarrow \emptyset$
  while $h(\Delta) > E$ **do**
    $a_i \leftarrow a' \in \arg\max_{a \in \mathcal{V} \setminus \Delta} \{h(\Delta) - h(\Delta \cup \{a\})\}$
    $\Delta \leftarrow \Delta \cup \{a_i\}$
  end while

Fact 1. *Denote as $\Delta^\star$ a solution to Problem ($\mathcal{O}$) and as $\Delta_0, \Delta_1, \ldots$ the sequence of sets picked by Algorithm 2. Moreover, let l be the smallest index such that $h(\Delta_l) \leq E$. Then,*

$$\frac{l}{|\Delta^\star|} \leq 1 + \log \frac{h(\mathcal{V}) - h(\emptyset)}{h(\mathcal{V}) - h(\Delta_{l-1})}.$$

In Section 3.3, we provide an efficient approximation algorithm for (I), by applying Fact 1 to an appropriately perturbed version of this problem, so that it involves a non-decreasing supermodular function, as in ($\mathcal{O}$). This also leads to our second main contribution, presented in Section 3.4: An efficient approximation algorithm for Problem (II), which selects all the actuators to maximize the volume of the set of states that can be reached with one unit or less of input energy, without assuming controllability beforehand. This is in contrast to the related works [52] and [88]: there, the authors consider a similar problem for choosing a few actuators to optimize $\log\det(W_{(\cdot)}^{-1})$; however, their results rely on the assumption of a pre-existing actuator set that renders (3.1) controllable, although this set is not selected towards the minimization of $\log\det(W_{(\cdot)}^{-1})$. Nevertheless, this assumption is necessary, since they then prove that the $\log\det(W_{(\cdot)}^{-1})$ is a supermodular function in the choice of the extra actuators. On the other hand, our algorithms select all the actuators towards the involved energy objective, since they rely on a ϵ-perturbed version of $\log\det(W_{(\cdot)}^{-1})$, that we prove to be supermodular without assuming controllability beforehand.

Overall, our results supplement the existing literature by considering Problems (I) and

(II) when the system is not initially controllable and by providing efficient approximation algorithms for their solution, along with worst-case performance guarantees.

3.3. Minimal Actuator Sets with Constrained Control Effort

We present an efficient approximation algorithm for Problem (I). To this end, we first generalize the involved energy metric to an ϵ-close one that remains well-defined even when the controllability matrix is not invertible. Next, we relax (I) by introducing a new program that makes use of this metric and circumvents the restrictive controllability constraint of (I). Moreover, we prove that for certain values of ϵ all solutions of this auxiliary problem render the system controllable. This fact, along with the supermodularity property of the generalized metric that we establish, leads to our proposed approximation algorithm. The discussion of its efficiency ends the analysis of (I).

3.3.1. An ϵ-close Auxiliary Problem

Consider the following approximation to (I)

$$\begin{aligned} &\underset{\Delta\subseteq\mathcal{V}}{\text{minimize}} \quad |\Delta| \\ &\text{subject to} \\ &\qquad \log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq \tilde{E}, \end{aligned} \tag{I$'$}$$

where $\tilde{W}_\Delta$ is equivalent to $W_\Delta/(2\lambda_{\max}(W_\mathcal{V}))$, $\lambda_{\max}(W_\mathcal{V})$ is the maximum eigenvalue of $W_\mathcal{V}$, $\tilde{E}$ is equal to $E + n\log(2\lambda_{\max}(W_\mathcal{V}))$, and ϵ is positive.

In contrast to (I), the domain of this problem consists of all subsets of $\mathcal{V}$ since $\tilde{W}_{(\cdot)} + \epsilon I$ is always invertible. The ϵ-closeness is evident since for any Δ such that $(A, B(\Delta))$ is controllable $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq \tilde{E}$ becomes $\log\det(W_\Delta^{-1}) \leq E$ as $\epsilon \to 0$. Due to the definition of $\tilde{W}_\Delta$, for all $\Delta \subseteq \mathcal{V}$, all eigenvalues of $\tilde{W}_\Delta$ are at most $1/2$ [93, Theorem 8.4.9]; this property will be useful in the proof of one of our main results, in particular, Proposition 1.

In the following paragraphs, we identify an approximation algorithm for solving Problem (I$'$), and correspondingly, the ϵ-close, NP-hard Problem (I).

3.3.2. Approximation Algorithm for Problem (I$'$)

We first prove that all solutions of (I$'$) for $0 < \epsilon \leq \min\{1/2, e^{-\tilde{E}}\}$ render the system controllable, notwithstanding that no controllability constraint is imposed by this program on the choice of the actuator sets. Moreover, we show that the involved ϵ-close energy metric is supermodular with respect to the choice of actuator sets and then we present our approximation algorithm, followed by a discussion of its efficiency which ends this subsection.

Proposition 1. *Consider a constant $\omega > 0$, ϵ such that $0 < \epsilon < \min\{1/2, e^{-\omega}\}$, and any $\Delta \subseteq \mathcal{V}$: If $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq \omega$, then $(A, B(\Delta))$ is controllable.*

Proof: Assume that $(A, B(\Delta))$ is not controllable and let k be the corresponding number of

non-zero eigenvalues of W_Δ which we denote as $\lambda_1, \lambda_2, \ldots, \lambda_k$; therefore, $k \leq n-1$. Then,

$$\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} = \sum_{i=1}^{k} \log \frac{1}{\frac{\lambda_i}{2\lambda_{\max}(W_\mathcal{V})} + \epsilon} + (n-k)\log\frac{1}{\epsilon} > \log\frac{1}{\epsilon} > \omega,$$

since $\frac{\lambda_i}{2\lambda_{\max}(W_\mathcal{V})} + \epsilon < 1$ (because $\frac{\lambda_i}{2\lambda_{\max}(W_\mathcal{V})} \leq 1/2$ and $\epsilon < 1/2$), and $\epsilon < e^{-\omega}$. Therefore, we have a contradiction. ■

Note that ω is chosen independently of the parameters of system (3.1). Therefore, the absence of the controllability constraint in Problem (I′) for $0 < \epsilon \leq \min\{1/2, e^{-\tilde{E}}\}$ is fictitious; nonetheless, it obviates the necessity of considering only actuator sets that render the system controllable.

The next proposition is also essential and suggests an efficient approximation algorithm for solving (I′).

Proposition 2 (Supermodularity)**.** *The function* $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} : \Delta \subseteq \mathcal{V} \mapsto \mathbb{R}$ *is supermodular and non-increasing set with respect to the choice of* Δ*.*

Proof: To prove that the $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1}$ is non-increasing, recall from (3.5) that for any $\Delta_1 \subseteq \Delta_2 \subseteq [n]$, $\tilde{W}_{\Delta_1} \preceq \tilde{W}_{\Delta_2}$. Therefore, from [93, Theorem 8.4.9], $\log\det(\tilde{W}_{\Delta_2} + \epsilon I)^{-1} \preceq \log\det(\tilde{W}_{\Delta_1} + \epsilon I)^{-1}$, and as a result, $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1}$ is non-increasing.

Next, to prove that $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1}$ is a supermodular set function, recall from Section 3.2.4 that it suffices to prove that $\log\det(\tilde{W}_\Delta + \epsilon I)$ is a submodular one. In particular, recall that a function $h : 2^{[n]} \mapsto \mathbb{R}$ is submodular if and only if, for any $a \in [n]$, the function $h_a : 2^{[n]\setminus\{a\}} \mapsto \mathbb{R}$, where $h_a(\Delta) \equiv h(\Delta \cup \{a\}) - h(\Delta)$, is a non-increasing set function. Therefore, to prove that $h(\Delta) = \log\det(\tilde{W}_\Delta + \epsilon I)$ is submodular, we may prove that the $h_a(\Delta)$ is a non-increasing set function. To this end, we follow the proof of Theorem 6 in [52]: first, observe that

$$\begin{aligned} h_a(\Delta) &= \log\det(\tilde{W}_{\Delta\cup\{a\}} + \epsilon I) - \log\det(\tilde{W}_\Delta + \epsilon I) \\ &= \log\det(\tilde{W}_\Delta + \tilde{W}_a + \epsilon I) - \log\det(\tilde{W}_\Delta + \epsilon I). \end{aligned}$$

Now, for any $\Delta_1 \subseteq \Delta_2 \subseteq [n]$ and $z \in [0,1]$, define $\Omega(z) \equiv \epsilon I + \tilde{W}_{\Delta_1} + z(\tilde{W}_{\Delta_2} - \tilde{W}_{\Delta_1})$ and $\bar{h}(z) \equiv \log\det(\Omega(z) + \tilde{W}_a) - \log\det(\Omega(z))$; it is $\bar{h}(0) = h_a(\Delta_1)$ and $\bar{h}(1) = h_a(\Delta_2)$. Moreover, since $d\log\det(\Omega(z)))/dz = \mathrm{tr}\left(\Omega(z)^{-1} d\Omega(z)/dz\right)$ (cf. equation (43) in [95]),

$$\frac{d\bar{h}(z)}{dz} = \mathrm{tr}[((\Omega(z) + \tilde{W}_a)^{-1} - \Omega(z)^{-1})O_{21}],$$

where $O_{21} \equiv \tilde{W}_{\Delta_2} - \tilde{W}_{\Delta_1}$. From [93, Proposition 8.5.5], $(\Omega(z) + \tilde{W}_a)^{-1} \preceq \Omega(z)^{-1}$, because $\Omega(z) \succ 0$ for all $z \in [0,1]$, since $\epsilon I \succ 0$, $\tilde{W}_{\Delta_1} \succeq 0$, and $\tilde{W}_{\Delta_2} \succeq \tilde{W}_{\Delta_1}$. Thereby, from [93, Corollary 8.3.6], all eigenvalues of $((\Omega(z) + \tilde{W}_a)^{-1} - \Omega(z)^{-1})O_{21}$ are non-positive. As a

result, $d\bar{h}(z)/dz \leq 0$, and

$$h_a(\Delta_2) = \bar{h}(1) = \bar{h}(0) + \int_0^1 \frac{d\bar{h}(z)}{dz} dz \leq \bar{h}(0) = h_a(\Delta_1).$$

Therefore, $h_a(\Delta)$ is a non-increasing set function, and the proof is complete. ■

Therefore, the hardness of the ϵ-close Problem (I) is in agreement with that of the class of minimum set-covering problems subject to submodular constraints. Inspired by this literature [16, 94, 96], we have the following efficient approximation algorithm for Problem (I′), and as we show by the end of this section, for Problem (I) as well.

Algorithm 3 Approximation Algorithm for the Problem (I′).

Input: Bound $\tilde{E}$, parameter $\epsilon \leq \min\{1/2, e^{-\tilde{E}}\}$, matrices $W_1, W_2, \ldots, W_n$.
Output: Actuator set Δ.
  $\Delta \leftarrow \emptyset$
  while $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} > \tilde{E}$ **do**
    $a_i \leftarrow a' \in \arg\max_{a \in \mathcal{V}\setminus\Delta}\{\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} - \log\det(\tilde{W}_{\Delta\cup\{a\}} + \epsilon I)^{-1}\}$
    $\Delta \leftarrow \Delta \cup \{a_i\}$
  end while

Regarding the quality of Algorithm 3 the following is true.
Theorem 1 (A Submodular Set Coverage Optimization)**.** *Denote as $\Delta^\star$ a solution to Problem* (I′) *and as Δ the selected set by Algorithm 3. Then,*

$$(A, B(\Delta)) \text{ is controllable}, \tag{3.7}$$

$$\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq \tilde{E}, \tag{3.8}$$

$$\frac{|\Delta|}{|\Delta^\star|} \leq 1 + \log \frac{n\log(\epsilon^{-1}) - \log\det(\tilde{W}_\mathcal{V} + \epsilon I)^{-1}}{\tilde{E} - \log\det(\tilde{W}_\mathcal{V} + \epsilon I)^{-1}} \equiv F, \tag{3.9}$$

$$F = O(\log n + \log\log(\epsilon^{-1}) + \log \frac{1}{\tilde{E} - \log\det(\tilde{W}_\mathcal{V}^{-1})}). \tag{3.10}$$

Finally, the computational complexity of Algorithm 3 is $O(n^5)$.

Proof: We first prove (3.8), (3.9) and (3.10), and then, (3.7). We end the proof by clarifying the computational complexity of Algorithm 3.

First, let $\Delta_0, \Delta_1, \ldots$ be the sequence of sets selected by Algorithm 3 and l the smallest index such that $\log\det(\tilde{W}_{\Delta_l} + \epsilon I)^{-1} \leq E$. Therefore, Δ_l is the set that Algorithm 3 returns, and this proves (3.8).

Moreover, from [94], since for any $\Delta \subseteq \mathcal{V}$, $h(\Delta) \equiv -\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} + n\log(\epsilon^{-1})$ is a non-negative, non-decreasing, and submodular function (cf. Proposition 2), it is guaranteed

for Algorithm 3 that (cf. Fact 1)

$$\begin{aligned}\frac{l}{|\Delta^\star|} &\leq 1+\log\frac{h(\mathcal{V})-h(\emptyset)}{h(\mathcal{V})-h(\Delta_{l-1})}\\ &= 1+\\ &\quad \log\frac{n\log(\epsilon^{-1})-\log\det(\tilde{W}_{\mathcal{V}}+\epsilon I)^{-1}}{\log\det(\tilde{W}_{\Delta_{l-1}}+\epsilon I)^{-1}-\log\det(\tilde{W}_{\mathcal{V}}+\epsilon I)^{-1}}.\end{aligned}$$

Now, l is the first time that $\log\det(\tilde{W}_{\Delta_l}+\epsilon I)^{-1}\leq\tilde{E}$, and a result $\log\det(\tilde{W}_{\Delta_{l-1}}+\epsilon I)^{-1}>\tilde{E}$. This implies (3.9).

Moreover, observe that $0<\log\det(\tilde{W}_{\mathcal{V}}+\epsilon I)^{-1}<\log\det(\tilde{W}_{\mathcal{V}}^{-1})$ so that from (3.9) we get $F\leq 1+\log[n\log(\epsilon^{-1})/(\tilde{E}-\log\det(\tilde{W}_{\mathcal{V}}^{-1}))]$, which in turn implies (3.10).

On the other hand, since $0<\epsilon\leq\min\{1/2,e^{-\tilde{E}}\}$ and $\log\det(\tilde{W}_{\Delta_l}+\epsilon I)^{-1}\leq\tilde{E}$, Proposition 1 is in effect, i.e., (3.7) holds true.

Finally, with respect to the computational complexity of Algorithm 3, note that the `while` loop is repeated for at most n times. Moreover, the complexity to compute the determinant an $n\times n$ matrix, using Gauss-Jordan elimination decomposition, is $O(n^3)$. Additionally, at most n matrices must be inverted so that the "$\arg\max_{a\in\mathcal{V}\setminus\Delta}\{\log\det(\tilde{W}_{\Delta}+\epsilon I)^{-1}-\log\det(\tilde{W}_{\Delta\cup\{a\}}+\epsilon I)^{-1}\}$" can be computed. Furthermore, $O(n)$ time is required to find a maximum element between n available. Therefore, the computational complexity of Algorithm 3 is $O(n^5)$. ■

Therefore, Algorithm 3 returns a set of actuators that meets the corresponding control energy bound of Problem (I′) while it renders system (3.1) controllable. Moreover, the cardinality of this set is up to a multiplicative factor of F from the minimum cardinality actuator sets that meet the same control energy bound.

The dependence of F on n,ϵ and E was expected from a design perspective: Increasing the network size n or improving the accuracy by decreasing ϵ, as well as demanding a better energy guarantee by decreasing E should all push the cardinality of the selected actuator set upwards. Also, $\log\log(\epsilon^{-1})$ is the design cost for circumventing the difficult to satisfy controllability constraint of (I) [82], i.e., for assuming no pre-existing actuators that renders (3.1) controllable and choosing all the actuators towards the satisfaction of an energy performance criterion.

From a computational perspective, the computation of the determinant is the only intensive procedure of Algorithm 3, requiring $O(n^3)$ time, if we use the Gauss-Jordan elimination decomposition. On the other hand, to apply this algorithm on large-scale systems, we can speed up this procedure using the Coppersmith-Winograd algorithm [97], which requires $O(n^{2.376})$ time. Alternatively, we can use numerical methods, which efficiently compute an approximate the determinant of a matrix even if its size is of several thousands [98]. Moreover, we can speed up Algorithm 3 using a method proposed in [99], which avoids the computation of $\log\det(\tilde{W}_{\Delta}+\epsilon I)^{-1}-\log\det(\tilde{W}_{\Delta\cup\{a\}}+\epsilon I)^{-1}$ for unnecessary choices of a,

towards the computation of the $\arg\max_{a\in\mathcal{V}\setminus\Delta}\{\log\det(\tilde{W}_\Delta+\epsilon I)^{-1}-\log\det(\tilde{W}_{\Delta\cup\{a\}}+\epsilon I)^{-1}\}$, by taking advantage of the supermodularity of $\log\det(\tilde{W}_{(\cdot)}+\epsilon I)^{-1}$.

Finally, for large values of n, the computation of $W_1, W_2, \ldots, W_n$ is demanding as well. On the other hand, in the case of stable systems, as many physical, e.g., biological, networks are, the corresponding controllability Gramians can be used instead, which for a stable system can be calculated from the Lyapunov equations $AG_i + G_iA^T = -I^{(i)}$, for $i = 1, 2, \ldots, n$, respectively, and are given in closed-form by

$$G_i = \int_{t_0}^{\infty} \mathrm{e}^{A(t-t_0)} I^{(i)} \mathrm{e}^{A^T(t-t_0)}\, \mathrm{d}t. \tag{3.11}$$

Using these Gramians for the evaluation of W in (3.4) corresponds to the minimum state transfer energy with no time constraints. The advantage of this approach is that (3.11) can be solved efficiently using numerical methods, even when the system's size n has a value of several thousands [100].

In Section 3.3.3 we finalize our treatment of Problem (I) by employing Algorithm 3 to approximate its solutions.

3.3.3. Approximation Algorithm for Problem (I)

We present an efficient approximation algorithm for Problem (I) that is based on Algorithm 3. Let Δ be the actuator set returned by Algorithm 3, so that $(A, B(\Delta))$ is controllable and $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq \tilde{E}$. For any $c > 0$, there exists sufficiently small $\epsilon(c)$ such that:

$$\log\det(\tilde{W}_\Delta + \epsilon(c) I)^{-1} \geq \log\det(\tilde{W}_\Delta^{-1}) - c\tilde{E}. \tag{3.12}$$

Moreover, $\log\det(\tilde{W}_\Delta+\epsilon(c)I)^{-1} \leq \tilde{E}$, and therefore we get from (3.12) that $\log\det(\tilde{W}_\Delta^{-1}) \leq (1+c)\tilde{E}$, or

$$\log\det(W_\Delta^{-1}) \leq E + c\tilde{E}. \tag{3.13}$$

Hence, we refer to c as *approximation error*.

On the other hand, $\epsilon(c)$ is not known a priori. Hence, we need to search for a sufficiently small ϵ so that (3.13) holds true. One way to achieve this since ϵ is lower and upper bounded by 0 and $\min\{1/2, e^{-\tilde{E}}\}$, respectively, is to perform a search using bisection. We implement this procedure in Algorithm 4, where we denote as [Algorithm 3]$(\tilde{E}, \epsilon)$ the set that Algorithm 3 returns for given $\tilde{E}$ and ϵ.

In the worst case, when we first enter the inner `while` loop, the `if` condition is not satisfied, and as a result ϵ is set to a lower value. This process continues until the `if` condition is satisfied for the first time, given that a_0 is sufficiently small for the specified c, from which point and on this `while` loop converges up to the accuracy level a_0 to the largest value $\bar{\epsilon}$ of ϵ such that $\log\det(\tilde{W}_\Delta^{-1}) - \log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq c\tilde{E}$; specifically, $|\epsilon - \bar{\epsilon}| \leq a_0/2$, due to the mechanics of the bisection method. On the other hand, if a_0 is not sufficiently small, the

Algorithm 4 Approximation Algorithm for the Problem (I).

Input: Bound E, approximation error c, bisection's initial accuracy level a_0, matrices $W_1, W_2, \ldots, W_n$.

Output: Actuator set Δ.

```
a ← a_0, flag ← 0, l ← 0, u ← min{1/2, e^{-Ẽ}}, ε ← (l + u)/2
while flag ≠ 1 do
  while u − l > a do
  Δ ← [Algorithm 3](Ẽ, ε)
    if log det(W̃_Δ^{-1}) − log det(W̃_Δ + εI)^{-1} > cẼ then
      u ← ε
    else
      l ← ε
    end if
  ε ← (l + u)/2
  end while
  if log det(W̃_Δ^{-1}) − log det(W̃_Δ + εI)^{-1} > cẼ then
    u ← ε, ε ← (l + u)/2
  end if
Δ ← [Algorithm 3](Ẽ, ε)
  if log det(W̃_Δ^{-1}) − log det(W̃_Δ + εI)^{-1} ≤ cẼ then
    flag ← 1
  else
    a ← a/2
  end if
end while
```

value of a decreases within the last `if` statement of the algorithm, the variable flag remains zero and the outer loop is executed again, until the convergence within the inner `while` is feasible. Then, the `if` statement that follows the inner `while` loop ensures that ϵ is set below $\bar{\epsilon}$, so that $\log\det(\tilde{W}_\Delta^{-1}) - \log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq c\tilde{E}$. Finally, the last `if` statement sets the flag to 1 and the algorithm terminates. The efficiency of this algorithm for Problem (I) is summarized below.

Theorem 2. (Approximation Efficiency and Computational Complexity of Algorithm 4 for Problem (I)**)** *Denote as* $\Delta^\star$ *a solution to Problem* (I) *and as* Δ *the selected set by Algorithm 4. Then,*

$$(A, B(\Delta)) \text{ is controllable},$$

$$\log\det(W_\Delta^{-1}) \leq E + c\tilde{E}, \tag{3.14}$$

$$\frac{|\Delta|}{|\Delta^\star|} \leq F, \tag{3.15}$$

$$F = O(\log n + \max\{\log\log(n/(c\tilde{E})), \log\tilde{E}\} + \log\frac{1}{\tilde{E} - \log\det(\tilde{W}_\mathcal{V}^{-1})}). \tag{3.16}$$

Finally, let a be the bisection's accuracy level that Algorithm 4 terminates with. Then, if $a = a_0$, the computational complexity of Algorithm 4 is $O(n^5 \log_2(1/a_0))$, else it is:

$$O(n^5 \log_2(1/a) \log_2(a_0/a)).$$

Proof: We only prove statements (3.14), (3.15) and (3.16), while the first follows from Theorem 1. We end the proof by clarifying the computational complexity of Algorithm 4.

First, when Algorithm 4 exits the `while` loop, and after the following `if` statement,

$$\log\det(\tilde{W}_\Delta^{-1}) - \log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq c\tilde{E},$$

and since $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \leq \tilde{E}$, this implies (3.14).

To show (3.15), consider any solution $\Delta^\star$ to Problem (I) and any solution $\Delta^\bullet$ to Problem (I′). Then, $|\Delta^\star| \geq |\Delta^\bullet|$; to see this, note that for any $\Delta^\star$, $\log\det(\tilde{W}_{\Delta^\star} + \epsilon I)^{-1} < \log\det(\tilde{W}_{\Delta^\star}^{-1}) \leq \tilde{E}$ since $\epsilon > 0$, i.e., $\Delta^\star$ is a candidate solution to Problem (I′) because it satisfies all of its constraints. Therefore, $|\Delta^\star| \geq |\Delta^\bullet|$, and as a result $|\Delta|/|\Delta^\star| \leq |\Delta|/|\Delta^\bullet| \leq F$ per (3.9).

Next, note that (3.14) holds true when, e.g., ϵ is equal to $c\tilde{E}/(2n)$. Therefore, since also $\epsilon \leq e^{-\tilde{E}}$, $\log\log\epsilon^{-1} = O(\max\{\log\log(n/(c\tilde{E})), \log\tilde{E}\})$ and this proves (3.16).

Finally, with respect to the computational complexity of Algorithm 4, note that the inner `while` loop is repeated for at most $\log_2(1/(2a))$ times (since $\epsilon \leq 1/2$), in the worst case. Moreover, the time complexity of the procedures within this loop is of order $O(n^5)$, due to Algorithm 3. Finally, if $a = a_0$, the outer `while` loop runs for one time, and otherwise, for $\log_2(a_0/a)$ times. Therefore, the computational complexity of Algorithm 4 is $O(n^5 \log_2(1/a_0))$, or $O(n^5 \log_2(1/a) \log_2(a_0/a))$, respectively. ■

From a computational perspective, we can speed up Algorithm 4 using the methods we discussed in the end of Section 3.3.2. Moreover, for a wide class of systems, e.g., when $a = O(n^{n^{c_1}})$, where c_1 is a positive constant, independent of n, this algorithm runs in polynomial time, due to the logarithmic dependence on a.

From an approximation efficiency perspective we have that $F = O(\log(n))$, whenever $E = O(n^{c_1})$, $\lambda_{\max}(W_\mathcal{V}) = O(n^{n^{c_2}})$ and $1/(\tilde{E} - \log\det(\tilde{W}_\mathcal{V}^{-1})) = O(n^{c_3})$, where c_1, c_2 and c_3 are positive constants and independent of n. In other words, the cardinality of the actuator set that Algorithm 4 returns is up to a multiplicative factor of $O(\log n)$ from the minimum cardinality actuator sets that meet the same energy bound. Indeed, this is the best achievable bound in polynomial time for the set covering problem in the worst case [13], while (I) is a generalization of it [82]. Thus, Algorithm 4 is a best-approximation of (I) for this class of systems.

3.4. Minimum Energy Control by a Cardinality-Constrained Actuator Set

We present an approximation algorithm for Problem (II) following a parallel line of thought as in Section 3.3: First, we circumvent the restrictive controllability constraint of (II) using

the ϵ-close generalized energy metric defined in Section 3.3. Then, we propose an efficient approximation algorithm for its solution that makes use of Algorithm 4; this algorithm returns an actuator set that always renders (3.1) controllable while it guarantees a value for (II) that is provably close to its optimal one. We end the analysis of (II) by explicating further the efficiency of this procedure.

3.4.1. An ϵ-close Auxiliary Problem

For $\epsilon > 0$ consider the following approximation to (II)

$$\begin{aligned} \underset{\Delta \subseteq \mathcal{V}}{\text{minimize}} \quad & \log\det(\tilde{W}_\Delta + \epsilon I)^{-1} \\ \text{subject to} \quad & \\ & |\Delta| \leq r. \end{aligned} \tag{II$'$}$$

In contrast to (II), the domain of this problem consists of all subsets of $\mathcal{V}$ since $\tilde{W}_{(\cdot)} + \epsilon I$ is always invertible. Moreover, its objective is ϵ-close to that of Problem (II).

In the following paragraphs, we identify an efficient approximation algorithm for solving Problem (II$'$), and correspondingly, the ϵ-close, NP-hard Problem (II). We note that the hardness of the latter is in accordance with that of the general class of supermodular function minimization problems, as per Proposition 2 the objective $\log\det(\tilde{W}_\Delta + \epsilon I)^{-1}$ is supermodular. The approximation algorithms used in that literature however [16, 94, 96], fail to provide an efficient solution algorithm for (II$'$) — for completeness, we discuss this direction in the Appendix 3.6.1. In the next subsection we propose an efficient approximation algorithm for (II) that makes use of Algorithm 4.

3.4.2. Approximation Algorithm for Problem (II)

We provide an efficient approximation algorithm for Problem (II) that is based on Algorithm 4. In particular, since (II) finds an actuator set that minimizes $\log\det(W_{(\cdot)}^{-1})$, and any solution to (I) satisfies $\log\det(W_{(\cdot)}^{-1}) \leq E$, one may repeatedly execute Algorithm 4 for decreasing values of E as long as the returned actuators are at most r and E satisfies the feasibility constraint $E \geq \log\det(W_{\mathcal{V}}^{-1})$ (cf. Section 3.2.3). Therefore, for solving (II) we propose a bisection-type execution of Algorithm 4 with respect to E.

To this end, we also need an upper bound for the value of (II): Let $\Delta_{\mathcal{C}}$ be a small actuator set that renders system (3.1) controllable; it is efficiently found using Algorithm 4 for large E or the procedure proposed in [82]. Then, for any $r \geq |\Delta_{\mathcal{C}}|$, $\log\det(\tilde{W}_{\Delta_{\mathcal{C}}}^{-1})$ upper bounds the value of (II) since $\log\det(\tilde{W}_{(\cdot)}^{-1})$ is monotone.

Thus, having a lower and upper bound for the value of (II), we implement Algorithm 5 for approximating the solutions of (II); we consider only the non-trivial case where $r < n$ and denote the set that Algorithm 4 returns as [Algorithm 4]$(\tilde{E}, c, a_0)$ for given $\tilde{E}$, c and a_0.

In the worst case, when we first enter the `while` loop, the `if` condition is not satisfied, and as a result $\tilde{E}$ is set to a greater value. This process continues until the `if` condition is satisfied

Algorithm 5 Approximation algorithm for Problem (II).

Input: Set $\Delta_{\mathcal{C}}$, maximum number of actuators r such that $r \geq |\Delta_{\mathcal{C}}|$, approximation error c for Algorithm 4, bisection's accuracy level a_0 for Algorithm 4, bisection's accuracy level a_0' for current algorithm, matrices $W_1, W_2, \ldots, W_n$.

Output: Actuator set Δ.

$\Delta \leftarrow \emptyset$, $l \leftarrow \log\det(\tilde{W}_{\mathcal{V}}^{-1})$, $u \leftarrow \mathrm{tr}(W_{\Delta_{\mathcal{C}}}^{-1})$, $\tilde{E} \leftarrow (l+u)/2$, $\epsilon \leftarrow \min\{1/2, e^{-\tilde{E}}\}$
while $u - l > a_0'$ **do**
$\Delta \leftarrow$ [Algorithm 4]$(\tilde{E}, c, a_0)$
  if $|\Delta| > r$ **then**
    $l \leftarrow \tilde{E}$, $\tilde{E} \leftarrow (l+u)/2$
  else
    $u \leftarrow \tilde{E}$, $\tilde{E} \leftarrow (l+u)/2$
  end if
$\epsilon \leftarrow 1/\tilde{E}$
end while
if $|\Delta| > r$ **then**
  $l \leftarrow \tilde{E}$, $\tilde{E} \leftarrow (l+u)/2$
end if
$\Delta \leftarrow$ [Algorithm 4]$(\tilde{E}, c, a_0)$

for the first time from which point and on the algorithm converges up to the accuracy level a_0 to the smallest value $\underline{\tilde{E}}$ of $\tilde{E}$ such that $|\Delta| \leq r$; specifically, $|\tilde{E} - \underline{\tilde{E}}| \leq a_0'/2$ due to the mechanics of the bisection method, where $\underline{\tilde{E}} \equiv \min\{\tilde{E} : |[\text{Algorithm 4}](\tilde{E}, c, a_0)| \leq r\}$. Hereby $\underline{\tilde{E}}$ is the least bound $\tilde{E}$ for which Algorithm 4 returns an actuator set of cardinality at most r for the specified c and a_0 — $\underline{\tilde{E}}$ may be larger than the value of (II) due to worst-case approximability of the involved problems (cf. Theorem 2). Then, Algorithm 5 exits the `while` loop and the last `if` statement ensures that $\tilde{E}$ is set below $\underline{\tilde{E}}$ so that $|\Delta| \leq r$. Moreover, per Theorem 2 this set renders (3.1) controllable and guarantees that $\log\det(\tilde{W}_{\Delta}^{-1}) \leq E + c\tilde{E}$. Finally, with respect to the computational complexity of Algorithm 5, note that the `while` loop is repeated for at most $\log_2\left[(\log\det(\tilde{W}_{\Delta_{\mathcal{C}}}^{-1}) - \log\det(\tilde{W}_{\mathcal{V}}^{-1}))/a_0'\right]$ times. Moreover, the time complexity of the procedures within this loop are, in the worst case, of the same order as that of Algorithm 4 when it is executed for $\tilde{E}$ equal to $\underline{\tilde{E}}$. Regarding Theorem 2, denote this time complexity as $C(\underline{\tilde{E}}, c, a_0)$. Therefore, the computational complexity of Algorithm 4 is $O\left(C(\underline{\tilde{E}}, c, a_0)\log_2\left[(\log\det(\tilde{W}_{\Delta_{\mathcal{C}}}^{-1}) - \log\det(\tilde{W}_{\mathcal{V}}^{-1}))/a'\right]\right)$.

We summarize the above in the next corollary, which also ends the analysis of Problem (II).

Corollary 1. (Approximation Efficiency and Computational Complexity of Algorithm 5 for Problem (II)) *Denote as* Δ *the selected set by Algorithm 5. Then,*

$$
\begin{aligned}
&(A, B(\Delta)) \text{ is controllable},\\
&\log\det(W_{\Delta}^{-1}) \leq E + c\tilde{E},\\
&|\tilde{E} - \underline{\tilde{E}}| \leq a'/2,
\end{aligned}
$$

where $\underline{\tilde{E}} = min\{\tilde{E} : |[Algorithm\ 4](\tilde{E}, c, a)| \leq r\}$ *is the least bound* $\tilde{E}$ *that Algorithm 4 satisfies with an actuator set of cardinality at most* r *for the specified* c *and* a*. Finally, the computational complexity of Algorithm 5 is*

$$O\left(C(\underline{\tilde{E}}, c, a_0)\log_2\left(\frac{\log\det(\tilde{W}_{\Delta_\mathcal{C}}^{-1}) - \log\det(\tilde{W}_{\mathcal{V}}^{-1})}{a'}\right)\right),$$

where $C(\underline{\tilde{E}}, c, a_0)$ *denotes the computational complexity of Algorithm 4, with respect to Theorem 2, when it is executed for* $\tilde{E}$ *equal to* $\underline{\tilde{E}}$*.*

From a computational perspective, we can speed up Algorithm 5 using the methods we discussed in the end of Section 3.3.2. Moreover, for a wide class of systems, e.g., when $a = O(n^{n^{c_1}})$, where c_1 is a positive constant, independent of n, and similarly for a' and $\log\det(\tilde{W}_{\Delta_\mathcal{C}}^{-1})$, this algorithm runs in polynomial time, due to the logarithmic dependence on a, a' and $\log\det(\tilde{W}_{\Delta_\mathcal{C}}^{-1})$, respectively.

3.5. Concluding Remarks & Future Work

We addressed two actuator placement problems in linear systems: First, the problem of minimal actuator placement so that the volume of the set of states reachable with one or less units of input energy is lower bounded by a desired value, and then the problem of cardinality-constrained actuator placement for minimum control effort, where the optimal actuator set is selected so that the volume of the set of states that can be reached with one unit or less of input energy is maximized. Both problems were shown to be NP-hard, while for the first one we provided a best approximation algorithm for a given range of the problem parameters. Next, we proposed an efficient approximation algorithm for the solution of the second problem as well. Our future work is focused on exploring the effect that the underlying network topology of the involved system has on these actuator placement problems, as well as investigating distributed implementations of their corresponding algorithms.

3.6. Appendix: Computational Complexity

We prove that Problem I is NP-hard, providing an instance that reduces to the NP-hard controllability problem introduced in [82]. In particular, it is shown in [82] that deciding if (3.1) is controllable by a zero-one diagonal matrix B with $r+1$ non-zero entries reduces to the r-hitting set problem, as we define it below, which is NP-hard [101].

Definition 11 (r-hitting set problem)**.** *Given a finite set* $\mathcal{M}$ *and a collection* $\mathcal{C}$ *of non-empty subsets of* $\mathcal{M}$*, find an* $\mathcal{M}' \subseteq \mathcal{M}$ *of cardinality at most* r *that has a non-empty intersection with each set in* $\mathcal{C}$*.*

Without loss of generality, we assume that every element of $\mathcal{M}$ appears in at least one set in $\mathcal{C}$ and all sets in $\mathcal{C}$ are non-empty. Moreover in Definition 11, we let $|\mathcal{C}| = p$ and $\mathcal{M} = \{1, 2, \ldots, m\}$, and define $C \in \mathbb{R}^{p\times m}$ such that $C_{ij} = 1$ if the i-th set contains the element j and zero otherwise.

We show that Problem (I) for A as described below and with $E = n(2n)^{2n^2+12n+2} - n$ is equivalent to the NP-hard controllability problem introduced in [82]. Therefore, since E

can be described in polynomial time, as $log(E) = O(n^3)$, we conclude that Problem (I) is NP-hard.

In particular, as in [82], let $n = m + p + 1$ and $A = V^{-1}DV$, where $D \equiv \text{diag}(1, 2, \ldots, n)$ and[2]

$$V = \begin{bmatrix} 2I_{m\times m} & 0_{m\times p} & e_{m\times 1} \\ C & (m+1)I_{p\times p} & 0_{p\times 1} \\ 0_{1\times m} & 0_{1\times p} & 1 \end{bmatrix}. \tag{3.17}$$

It is shown in [82] that deciding if A is controllable by a zero-one diagonal matrix B with $r + 1$ non-zero entries is NP-hard.

Now, observe that all the entries of V are integers either zero or at most $m + 1$. Moreover, with respect to the entries of V^{-1}, it is shown in [82] that:

- For $i = 1, 2, \ldots, m$: It has a $1/2$ in the (i, i)-th place and a $-1/2$ in the (i, n)-th place, and zeros elsewhere.
- For $i = m+1, m+2, \ldots, m+p$: It has a $1/(m+1)$ in the (i, i)-th place, a $-1/(2(m+1))$ in the (i, j)-th place where $j \in C_i$ (C_i is the corresponding set of the collection $\mathcal{C}$), and $|C_i|/(2(m + 1))$ in the (i, n)-th place; every other entry of the i-th row is zero.
- Finally, the last row of V^{-1} is $[0, 0, \ldots, 0, 1]$.

Therefore, $2(m + 1)V^{-1}$ has all its entries as integers that are either zero or at most n^2, in absolute value.

Consider the controllability matrix associated with this system, given a zero-one diagonal B that makes it controllable, and denote it as W_B. Then,

$$\begin{aligned} W_B &= \int_{t_0}^{t_1} \mathrm{e}^{A(t-t_0)} BB^T \mathrm{e}^{A^T(t-t_0)}\, \mathrm{d}t \\ &= V^{-1} \int_{t_0}^{t_1} \mathrm{e}^{D(t-t_0)} VBV^T \mathrm{e}^{D^T(t-t_0)}\, \mathrm{d}t V^{-T}. \end{aligned}$$

Let $t_1 - t_0 = ln(n)$. Then, $(2n)! \int_0^{t_1-t_0} \mathrm{e}^{Dt} VBV^T \mathrm{e}^{D^T t}\, \mathrm{d}t$ evaluates to a matrix that has entries of the form $c_0 + c_1 n + c_2 n^2 + \ldots + c_n n^n$, where $c_0, c_1, \ldots, c_n$ are non-negative integers and all less than $(2n)! \leq (2n)^{2n}$. Thereby,

$$W'_B \equiv 4(m + 1)^2 (2n)! V^{-1} \int_0^{t_1-t_0} \mathrm{e}^{Dt} VBV^T \mathrm{e}^{D^T t}\, \mathrm{d}t V^{-T},$$

has entries of the form $c'_0 + c'_1 n + c'_2 n^2 + \ldots + c'_n n^n$, where $c'_0, c'_1, \ldots, c'_n$ are integers and all less than $(2n)^{2(n+3)}$ in absolute value due to the pre and post multiplications by $2(m+1)V^{-1}$ and $2(m + 1)V^{-T}$, respectively.

[2] V is invertible since it is strictly diagonally dominant.

We are interested on upper bounding $\log\det(W_B^{-1})$: since for $x > 0$, $\log(x) \leq x - 1$, $\log\det(W_B^{-1}) \leq tr(W_B^{-1}) - n$. In addition,

$$tr(W_B^{-1}) = 4(m+1)^2(2n)!tr(W_B'^{-1}) \leq (2n)^{2(n+1)}tr(W_B'^{-1}).$$

Therefore, we upper bound $tr(W_B'^{-1})$: Using Crammer's rule to compute $W_B'^{-1}$, due to the form of the entries of W_B', all of its elements, including the diagonal ones, if they approach infinity, they approach it with at most $n!n^n(2n)^{2n(n+3)} < (2n)^{2n(n+5)}$ speed, and as a result $tr(W_B'^{-1}) \leq n(2n)^{2n(n+5)}$. Hence, $tr(W_B^{-1}) \leq n(2n)^{2n(n+5)+2(n+1)} = n(2n)^{2n^2+12n+2}$, for any B that makes (3.1) controllable. Thus, if we set $E = n(2n)^{2n^2+12n+2} - n$ (which implies $log(E) = O(n^3)$ so that E can be described polynomially), Problem (I) is equivalent to the controllability problem of [82], which is NP-hard. ■

An immediate consequence of the above is the following one.
Corollary 2 (Computational Complexity of Problem (II))**.** *Problem* (II) *is NP-hard.*

3.6.1. The Greedy Algorithm used in the Supermodular Minimization Literature is Inefficient for solving Problem (II′)

Consider Algorithm 6 which is in accordance with the supermodular minimization literature [16, 94, 96].

Algorithm 6 Greedy algorithm for Problem (II′).

Input: Maximum number of actuators r, approximation parameter ϵ, number of steps that the algorithm will run l, matrices $W_1, W_2, \ldots, W_n$.
Output: Actuator set Δ_l
  $\Delta_0 \leftarrow \emptyset$, $i \leftarrow 0$
  while $i < l$ **do**
    $a_i \leftarrow \text{argmax}_{a \in \mathcal{V}\setminus\Delta}\{\log\det(W_{\Delta_i} + \epsilon I)^{-1} - \log\det(W_{\Delta_i \cup \{a\}} + \epsilon I)^{-1}\}$
    $\Delta_{i+1} \leftarrow \Delta_i \cup \{a_i\}, i \leftarrow i + 1$
  end while

The following is true for its performance.
Fact 2. *Let $v^\star$ denote the value of Problem* (II′)*. Then, Algorithm 6 guarantees that for any positive integer l,*

$$\log\det(W_{\Delta_l} + \epsilon I)^{-1} \leq (1 - e^{-l/r})v^\star + n\log(\epsilon^{-1})e^{-l/r}.$$

Proof: It follows from Theorem 9.3, Ch. III.3.9. of [96], since $-\log\det(W_{\Delta_l} + \epsilon I)^{-1} + n\log(\epsilon^{-1})$ is a non-negative, non-decreasing, and submodular function with respect to the choice of Δ (cf. Proposition 2). ■

Algorithm 6 suffers from an error term that is proportional to $n\log(\epsilon^{-1})$. Moreover, it is possible that Algorithm 6 returns an actuator set that does not render (3.1) controllable. Therefore, Algorithm 6 is inefficient for solving Problem (II′).

Part II

CONTRIBUTIONS TO SUBMODULAR MAXIMIZATION IN SENSING DESIGN

CHAPTER 4 : Sensor Placement for Optimal Kalman Filtering: Fundamental Limits, Submodularity, and Algorithms

In this chapter, we focus on sensor placement in linear dynamic estimation, where the objective is to place a small number of sensors in a system of interdependent states so to design an estimator with a desired estimation performance. In particular, we consider a linear time-variant system that is corrupted with process and measurement noise, and study how the selection of its sensors affects the estimation error of the corresponding Kalman filter over a finite observation interval. Our contributions are threefold: First, we prove that the minimum mean square error of the Kalman filter decreases only linearly as the number of sensors increases. That is, adding extra sensors so to reduce this estimation error is ineffective, a fundamental design limit. Similarly, we prove that the number of sensors grows linearly with the system's size for fixed minimum mean square error and number of output measurements over an observation interval; this is another fundamental limit, especially for systems where the system's size is large. Second, we prove that the log det of the error covariance of the Kalman filter, which captures the volume of the corresponding confidence ellipsoid, with respect to the system's initial condition and process noise is a supermodular and non-increasing set function in the choice of the sensor set. Therefore, it exhibits the diminishing returns property. Third, we provide an efficient approximation algorithm that selects a small number sensors so to optimize the Kalman filter with respect to this estimation error —the worst-case performance guarantees of this algorithm are provided as well.[1]

4.1. Introduction

In this chapter, we consider a linear time-variant system corrupted with process and measurement noise. Our first goal is to study how the placement of their sensors affects the minimum mean square error of their Kalman filter over a finite observation interval [103]. Moreover, we aim to select a small number of sensors so to minimize the volume of the corresponding confidence ellipsoid of this estimation error. Thereby, this study is an important distinction in the minimal sensor placement literature [7, 8, 84, 104, 105, 106, 107, 108, 109, 110, 111, 112], since the Kalman filter is the optimal linear estimator —in the minimum mean square sense— given a sensor set [113].

Our contributions are threefold:

Fundamental limits. First, we identify fundamental limits in the design of the Kalman filter with respect to its sensors. In particular, given any finite number of output measurements over an observation interval, we prove that the minimum mean square error of the Kalman filter decreases only linearly as the number of sensors increases. That is, adding extra sensors so to reduce this estimation error of the Kalman filter is ineffective, a fundamental design limit. Similarly, we prove that the number of sensors grows linearly with the system's size for fixed minimum mean square error; this is another fundamental limit, especially for systems where the system's size is large. Overall, our novel results quantify the trade-off between the number of sensors and that of output measurements so to achieve

[1] This chapter is based on the paper by Tzoumas et al. [102].

a specified value for the minimum mean square error.

These results are the first to characterize the effect of the sensor set on the minimum mean square error of the Kalman filter. In particular, in [84], the authors quantify only the trade-off between the total energy of the consecutive output measurements and the number of its selected sensors. Similarly, in [111], the authors consider only the maximum-likelihood estimator for the system's initial condition and only for a special class of stable linear time-invariant systems. Moreover, they consider systems that are corrupted merely with measurement noise, which is white and Gaussian. Finally, they also assume an infinite observation interval, that is, infinite number of consecutive output measurements. Nonetheless, we assume a finite observation interval and study the Kalman estimator both for the system's initial condition and for the system's state at the time of the last output measurement. In addition, we consider general linear time-variant systems that are corrupted with both process and measurement noise, of any distribution (with zero mean and finite variance). Overall, our results characterize the effect of the cardinality of the sensor set on the minimum mean square error of the Kalman filter, that is, the optimal linear estimator.

Submodularity. Second, we identify properties for the log det of the error covariance of the Kalman filter, which captures the volume of the corresponding confidence ellipsoid, with respect to the system's initial condition and process noise over a finite observation interval as a sensor set function —the design of an optimal Kalman filter with respect to the system's initial condition and process noise implies the design of an optimal Kalman filter with respect to the system's state. Specifically, we prove that it is a supermodular and non-increasing set function in the choice of the sensor set.

In contrast, in [114], the authors study sensor placement for monitoring static phenomena with only spatial correlations. To this end, they prove that the mutual information between the chosen and non-chosen locations is submodular. Notwithstanding, we consider dynamic phenomena with both spatial and temporal correlations, and as a result, we characterize as submodular a richer class of estimation performance metrics. Furthermore, in the sensor scheduling literature [20], the log det of the error covariance of the Kalman filter has been proven submodular but only for special cases of systems with zero process noise [115] and [5]. Nevertheless, we consider the presence of process noise, and prove our supermodularity result for the general case.[2]

Algorithms. Third, we consider the problem of sensor placement so to optimize the log det of the error covariance of the Kalman filter with respect to the system's initial condition and process noise over a finite observation interval —henceforth, we refer to this error as log det *error*, and to the latter problem as $\mathcal{P}_1$. Naturally, $\mathcal{P}_1$ is combinatorial, and in particular, it involves the minimization of a supermodular set function, that is, the minimum mean square error. Because the minimization of a general supermodular function is NP-hard [13],

[2] In [5], the authors prove with a counterexample in the context of sensor scheduling that the minimum mean square error of the Kalman filter with respect to the system's state is not in general a supermodular set function. We can extend this counterexample in the context of minimal sensor placement as well: the minimum mean square error of the Kalman with respect to the system's state is not in general a supermodular set function with respect to the choice of the sensor set.

we provide efficient approximation algorithms for their general solution, along with their worst-case performance guarantees. Specifically, we provide an efficient algorithm for $\mathcal{P}_1$ that returns a sensor set that satisfies the estimation guarantee of $\mathcal{P}_1$ and has cardinality up to a multiplicative factor from the minimum cardinality sensor sets that meet the same estimation bound. Moreover, this multiplicative factor depends only logarithmically on the problem's $\mathcal{P}_1$ parameters.[3]

In contrast, the related literature has focused either on the optimization of average estimation performance metrics, such as the log det of the error's covariance, or on heuristic algorithms that provide no worst-case performance guarantees. In particular, in [119], the authors minimize the log det of the error's covariance matrix of the Kalman estimator for the case where there is no process noise in the system's dynamics —in contrast, in our framework we assume both process and measurement noise. Moreover, to this end they use convex relaxation techniques that provide no performance guarantees. Furthermore, in [120] and [121], the authors design an H_2-optimal estimation gain with a small number of non-zero columns. To this end, they also use convex relaxation techniques that provide no performance guarantees. Finally, in [122], the author designs an output matrix with a desired norm so to minimize the minimum mean square error of the corresponding Kalman estimator; nonetheless, the author does not minimize the number of selected sensors. Overall, with this chapter we are the first to optimize the minimum mean square error of the Kalman filter using a small number of sensors and to provide worst-case performance guarantees.

The remainder of this chapter is organized as follows. In Section 4.2, we introduce our model, and our estimation and sensor placement framework, along with our sensor placement problems. In Section 4.3, we provide a series of design and performance limits and characterize the properties of the Kalman estimator with respect to its sensor set; in Section 4.4, we prove that the log det estimation error of the Kalman filter with respect to the system's initial condition and process noise is a supermodular and non-increasing set function in the choice of the sensor set; and in Section 4.5, we provide efficient approximation algorithms for selecting a small number of sensors so to design an optimal Kalman filter with respect to its log det error —the worst-case performance guarantees of these algorithms are provided as well. Finally, Section 4.6 concludes the chapter. Due to space limitations, the proofs of all of our results, as well as, the corresponding simulations, are omitted; they can be found in the full version of this chapter, located at our websites.

4.2. Problem Formulation

Notation. We denote the set of natural numbers $\{1, 2, \ldots\}$ as $\mathbb{N}$, the set of real numbers as $\mathbb{R}$, and the set $\{1, 2, \ldots, n\}$ as $[n]$, where $n \in \mathbb{N}$. Given a set $\mathcal{X}$, $|\mathcal{X}|$ is its cardinality. Matrices are represented by capital letters and vectors by lower-case letters. For a matrix A, $A^\top$ is its transpose and A_{ij} its element located at the $i-$th row and $j-$th column.

[3]Such algorithms, that involve the minimization of supermodular set functions, are also used in the machine learning [116], leader selection [8, 91, 92], sensor scheduling [5, 115], actuator placement [7, 106, 107, 110, 112, 117] and sensor placement in static environments [114, 118] literature. Their popularity is due to their simple implementation — they are greedy algorithms — and provable worst-case approximation factors, that are the best one can achieve in polynomial time for several classes of supermodular functions [13, 40].

$\|A\|_2 \equiv \sqrt{A^\top A}$ is its spectral norm, and $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ its minimum and maximum eigenvalues, respectively. Moreover, if A is positive semi-definite or positive definite, we write $A \succeq 0$ and $A \succ 0$, respectively. Furthermore, I is the identity matrix —its dimension is inferred from the context; similarly for the zero matrix 0. Finally, for a random variable $x \in \mathbb{R}^n$, $\mathbb{E}(x)$ is its expected value, and $\mathbb{C}(x) \equiv \mathbb{E}\left([x - \mathbb{E}(x)]\,[x - \mathbb{E}(x)]^\top\right)$ its covariance. The rest of our notation is introduced when needed.

4.2.1. Model and Estimation Framework

For $k \geq 0$, we consider the linear time-variant system

$$\begin{aligned} x_{k+1} &= A_k x_k + w_k, \\ y_k &= C_k x_k + v_k, \end{aligned} \tag{4.1}$$

where $x_k \in \mathbb{R}^n$ ($n \in \mathbb{N}$) is the state vector, $y_k \in \mathbb{R}^c$ ($c \in [n]$) the output vector, w_k the process noise and v_k the measurement noise —without loss of generality, the input vector is assumed zero. The initial condition is x_0.

Assumption 2. (For all $k \geq 0$, the initial condition, the process noise and the measurement noise are uncorrelated random variables) *x_0 is a random variable with covariance $\mathbb{C}(x_0) = \sigma^2 I$, where $\sigma \geq 0$. Moreover, for all $k \geq 0$, $\mathbb{C}(w_k) = \mathbb{C}(v_k) = \sigma^2 I$ as well. Finally, for all $k, k' \geq 0$ such that $k \neq k'$, x_0, w_k and v_k, as well as, w_k, $w_{k'}$, v_k and $v_{k'}$, are uncorrelated.*[4]

Moreover, for $k \geq 0$, consider the vector of measurements $\bar{y}_k$, the vector of process noises $\bar{w}_k$ and the vector of measurement noises $\bar{v}_k$, defined as follows: $\bar{y}_k \equiv (y_0^\top, y_1^\top, \ldots, y_k^\top)^\top$, $\bar{w}_k \equiv (w_0^\top, w_1^\top, \ldots, w_k^\top)^\top$, and $\bar{v}_k \equiv (v_0^\top, v_1^\top, \ldots, v_k^\top)^\top$, respectively; the vector $\bar{y}_k$ is known, while the $\bar{w}_k$ and $\bar{v}_k$ are not.

Definition 12 (Observation interval and its length)**.** *The interval $[0, k] \equiv \{0, 1, \ldots, k\}$ is called the* observation interval *of* (4.1)*. Moreover, $k + 1$ is its* length*.*

Evidently, the length of an observation interval $[0, k]$ equals the number of measurements $y_0, y_1, \ldots, y_k$.

In this chapter, given an observation interval $[0, k]$, we consider the minimum mean square linear estimators for $x_{k'}$, for any $k' \in [0, k]$ [103]. In particular, (4.1) implies

$$\bar{y}_k = \mathcal{O}_k z_{k-1} + \bar{v}_k, \tag{4.2}$$

where $\mathcal{O}_k$ is the $c(k+1) \times n(k+1)$ matrix $[L_0^\top C_0^\top, L_1^\top C_1^\top, \ldots, L_k^\top C_k^\top]^\top$, L_0 the $n \times n(k+1)$ matrix $[I, 0]$, L_i, for $i \geq 1$, the $n \times n(k+1)$ matrix $[A_{i-1} \cdots A_0, A_{i-1} \cdots A_1, \ldots, A_{i-1}, I, 0]$, and $z_{k-1} \equiv (x_0^\top, \bar{w}_{k-1}^\top)^\top$. As a result, the minimum mean square linear estimate of z_{k-1} is

[4] This assumption is common in the related literature [119], and it translates to a worst-case scenario for the problem we consider in this chapter.

the $\hat{z}_{k-1} \equiv \mathbb{E}(z_{k-1}) + \mathcal{O}_k^\top \left(\mathcal{O}_k \mathcal{O}_k^\top + I\right)^{-1} (\bar{y}_k - \mathcal{O}_k \mathbb{E}(z_{k-1}) - \mathbb{E}(\bar{v}_k))$; its error covariance is

$$\begin{aligned} \Sigma_{z_{k-1}} &\equiv \mathbb{E}\left((z_{k-1} - \hat{z}_{k-1})(z_{k-1} - \hat{z}_{k-1})^\top\right) \\ &= \sigma^2 \left(I - \mathcal{O}_k^\top \left(\mathcal{O}_k \mathcal{O}_k^\top + I\right)^{-1} \mathcal{O}_k\right) \end{aligned} \tag{4.3}$$

and its minimum mean square error

$$\begin{aligned} \text{mmse}(z_{k-1}) &\equiv \mathbb{E}\left((z_{k-1} - \hat{z}_{k-1})^\top (z_{k-1} - \hat{z}_{k-1})\right) \\ &= \text{tr}\left(\Sigma_{z_{k-1}}\right). \end{aligned} \tag{4.4}$$

As a result, the corresponding minimum mean square linear estimator of $x_{k'}$, for any $k' \in [0, k]$, is

$$\hat{x}_{k'} = L_{k'} \hat{z}_{k-1}, \tag{4.5}$$

(since $x_{k'} = L_{k'} z_{k-1}$), with minimum mean square error

$$\text{mmse}(x_{k'}) \equiv \text{tr}\left(\, L_{k'} \Sigma_{z_{k-1}} L_{k'}^\top\right). \tag{4.6}$$

In particular, the recursive implementation of (4.5) results to the Kalman filtering algorithm [123].

In this chapter, in addition to the minimum mean square error of $\hat{x}_{k'}$, we also consider per (4.5) the estimation error metric that is related to the η-confidence ellipsoid of $z_{k-1} - \hat{z}_{k-1}$ [119]. Specifically, this is the minimum volume ellipsoid that contains $z_{k-1} - \hat{z}_{k-1}$ with probability η, that is, the $\mathcal{E}_\epsilon \equiv \{z : z^\top \Sigma_{z_{k-1}} z \leq \epsilon\}$, where $\epsilon \equiv F^{-1}_{\chi^2_{n(k+1)}}(\eta)$ and $F_{\chi^2_{n(k+1)}}$ is the cumulative distribution function of a χ-squared random variable with $n(k+1)$ degrees of freedom [124]. Therefore, the volume of $\mathcal{E}_\epsilon$,

$$\text{vol}(\mathcal{E}_\epsilon) \equiv \frac{(\epsilon\pi)^{n(k+1)/2}}{\Gamma\left(n(k+1)/2 + 1\right)} \det\left(\Sigma^{1/2}_{z_{k-1}}\right), \tag{4.7}$$

where $\Gamma(\cdot)$ denotes the Gamma function [124], quantifies the estimation's error of $\hat{z}_{k-1}$, and as a result, for any $k' \in [0, k]$, of $\hat{x}_{k'}$ as well, since per (4.5) the optimal estimator for z_{k-1} defines the optimal estimator for $x_{k'}$.

Henceforth, we consider the logarithm of (4.7),

$$\log \text{vol}(\mathcal{E}_\epsilon) = \beta + 1/2 \log\det\left(\Sigma_{z_{k-1}}\right); \tag{4.8}$$

β is a constant that depends only on $n(k+1)$ and ϵ, in accordance to (4.7), and as a result, we refer to the $\log\det\left(\Sigma_{z_{k-1}}\right)$ as the $\log\det$ estimation error of the Kalman filter of (4.1):
Definition 13 ($\log\det$ estimation error of the Kalman filter)**.** *Given an observation interval* $[0, k]$*, the* $\log\det\left(\Sigma_{z_{k-1}}\right)$ *is called the* $\log\det$ estimation error of the Kalman filter *of* (4.1)*.*

In the following paragraphs, we present our sensor placement framework, that leads to our

sensor placement problems.

4.2.2. Sensor Placement Framework

In this chapter, we study among others the effect of the selected sensors in (4.1) on $\text{mmse}(x_0)$ and $\text{mmse}(x_k)$. Therefore, this translates to the following conditions on C_k, for all $k \geq 0$, in accordance with the minimal sensor placement literature [7].

Assumption 3 (C is a full row-rank constant zero-one matrix)**.** *For all $k \geq 0$, $C_k = C \in R^{c \times n}$, where C is a zero-one constant matrix. Specifically, each row of C has one element equal to one, and each column at most one, such that C has rank c.*

In particular, when for some i, C_{ij} is one, the j-th state of x_k is measured; otherwise, it is not. Therefore, the number of non-zero elements of C coincides with the number of placed sensors in (4.1).

Definition 14 (Sensor set and sensor placement)**.** *Consider a C per Assumption 3 and define $\mathcal{S} \equiv \{i : i \in [n] \text{ and } C_{ji} = 1, \text{ for some } j \in [r]\}$; $\mathcal{S}$ is called a* sensor set *or a* sensor placement *and each of its elements a* sensor*.*

4.2.3. Sensor Placement Problems

We introduce three objectives, that we use to define the sensor placement problems we consider in this chapter.

Objective 1 (Fundamental limits in optimal sensor placement)**.** *Given an observation interval $[0, k]$, $i \in \{0, k\}$ and a desired* $\text{mmse}(x_i)$*, identify fundamental limits in the design of the sensor set.*

As an example of a fundamental limit, we prove that the number of sensors grows linearly with the system's size for fixed estimation error $\text{mmse}(x_i)$ —this is clearly a major limitation, especially when the system's size is large. This result, as well as, the rest of our contributions with respect to Objective 1, is presented in Section 4.3.

Objective 2 ($\log\det$ estimation error as a sensor set function)**.** *Given an observation interval $[0, k]$, identify properties of the* $\log\det\left(\Sigma_{z_{k-1}}\right)$ *as a sensor set function.*

We address this objective in Section 4.4, where we prove that $\log\det\left(\Sigma_{z_{k-1}}\right)$ is a supermodular and non-increasing set function with respect to the choice of the sensor set —the basic definitions of supermodular set functions are presented in that section as well.

Objective 3 (Algorithms for optimal sensor placement)**.** *Given an observation interval $[0, k]$, identify a sensor set $\mathcal{S}$ that solves the* minimal sensor placement problem:

$$\begin{aligned} \underset{\mathcal{S} \subseteq [n]}{\textit{minimize}} \quad & |\mathcal{S}| \\ \textit{subject to} \quad & \log\det\left(\Sigma_{z_{k-1}}\right) \leq R. \end{aligned} \tag{$\mathcal{P}_1$}$$

That is, with $\mathcal{P}_1$ we design an estimator that guarantees a specified error and uses a minimal number of sensors. The corresponding algorithm is provided in Section 4.5.

All of our contributions with respect to the Objectives 1, 2 and 3 are presented in the

following sections.

4.3. Fundamental Limits in Optimal Sensor Placement

In this section, we present our contributions with respect to Objective 1. In particular, given any finite observation interval, we prove that the minimum mean square error decreases only linearly as the number of sensors increases. That is, adding extra sensors so to reduce the minimum mean square estimation error of the Kalman filter is ineffective, a fundamental design limit. Similarly, we prove that the number of sensors grows linearly with the system's size for fixed minimum mean square error; this is another fundamental limit, especially for systems where the system's size is large. On the contrary, given a sensor set of fixed cardinality, we prove that the length of the observational interval increases only logarithmically with the system's size for fixed minimum mean square error. Overall, our novel results quantify the trade-off between the number of sensors and that of output measurements so to achieve a specified value for the minimum mean square error.

To this end, given $i \in \{0, k\}$, we first determine a lower and upper bound for $\text{mmse}(x_i)$.[5]

Theorem 3. (A lower and upper bound for the estimation error with respect to the number of sensors and the length of the observation interval) *Consider a sensor set $\mathcal{S}$, any finite observation interval $[0, k]$ and a non-zero σ. Moreover, let $\mu \equiv \max_{m\in[0,k]} \|A_m\|_2$ and assume $\mu \neq 1$. Given $i \in \{0, k\}$,*

$$\frac{n\sigma^2 l_i}{|\mathcal{S}| \left(1 - \mu^{2(k+1)}\right) / \left(1 - \mu^2\right) + 1} \leq \text{mmse}(x_i) \leq n\sigma^2 u_i, \tag{4.9}$$

where $l_0 = 1$, $u_0 = 1$, $l_k = \lambda_{\min}\left(L_k^\top L_k\right)$ and $u_k = (k+1)\lambda_{\max}\left(L_k^\top L_k\right)$.

The upper bound corresponds to the case where no sensors have been placed ($C = 0$). On the other hand, the lower bound corresponds to the case where $|\mathcal{S}|$ sensors have been placed.

As expected, the lower bound in (4.9) decreases as the number of sensors or the length of the observational interval increases; the increase of either should push the estimation error downwards. Overall, this lower bound quantifies fundamental limits in the design of the Kalman estimator: first, this bound decreases only inversely proportional to the number of sensors. Therefore, the estimation error of the optimal linear estimator decreases only linearly as the number of sensors increases. That is, adding extra sensors so to reduce the minimum mean square estimation error of the Kalman filter is ineffective, a fundamental design limit. Second, this bound increases linearly with the system's size. This is another fundamental limit, especially for systems where the system's size is large. Finally, for fixed and non-zero $\lambda_{\min}\left(L_k^\top L_k\right)$, these scaling extend to the $\text{mmse}(x_k)$ as well, for any finite k.

Corollary 3. (Trade-off among the number of sensors, estimation error and the length of the observation interval) *Consider any finite observation interval $[0, k]$, a non-zero σ, and for $i \in \{0, k\}$, that the desired value for $\text{mmse}(x_i)$ is α ($\alpha > 0$). Moreover, let $\mu \equiv \max_{m\in[0,k]} \|A_m\|_2$ and assume $\mu \neq 1$. Any sensor set $\mathcal{S}$ that achieves $\text{mmse}(x_i) = \alpha$*

[5] The extension of Theorem 3 to the case $\mu = 1$ is straightforward, yet notationally involved; as a result, we omit it.

satisfies:

$$|\mathcal{S}| \geq \left(n\sigma^2 l_i/\alpha - 1\right) \frac{1-\mu^2}{1-\mu^{2(k+1)}}. \tag{4.10}$$

where $l_0 = 1$ *and* $l_k = \lambda_{\min}\left(L_k^\top L_k\right)$.

The above corollary shows that the number of sensors increases as the minimum mean square error or the number of output measurements decreases. More importantly, it shows that the number of sensors increases linearly with the system's size for fixed minimum mean square error. This is again a fundamental design limit, especially when the system's size is large.[6]

4.4. Submodularity in Optimal Sensor Placement

In this section, we present our contributions with respect to Objective 2. In particular, we first derive a closed formula for $\log\det\left(\Sigma_{z_{k-1}}\right)$ and then prove that it is a supermodular and non-increasing set function in the choice of the sensor set.

We now give the definition of a supermodular set function, as well as, that of an non-decreasing set function —we follow [94] for this material.

Denote as $2^{[n]}$ the power set of $[n]$.
Definition 15 (Submodularity and supermodularity)**.** *A function* $h : 2^{[n]} \mapsto \mathbb{R}$ *is* submodular *if for any sets* $\mathcal{S}$ *and* $\mathcal{S}'$*, with* $\mathcal{S} \subseteq \mathcal{S}' \subseteq [n]$*, and any* $a \notin \mathcal{S}'$*,*

$$h(\mathcal{S} \cup \{a\}) - h(\mathcal{S}) \geq h(\mathcal{S}' \cup \{a\}) - h(\mathcal{S}').$$

A function $h : 2^{[n]} \mapsto \mathbb{R}$ *is* supermodular *if* $(-h)$ *is submodular.*

An alternative definition of a submodular function is based on the notion of non-increasing set functions.
Definition 16 (Monotone set function)**.** *A function* $h : 2^{[n]} \mapsto \mathbb{R}$ *is a* non-increasing set function *if for any* $\mathcal{S} \subseteq \mathcal{S}' \subseteq [n]$*,* $h(\mathcal{S}) \geq h(\mathcal{S}')$*. Moreover,* h *is a* non-decreasing set function *if* $(-h)$ *is a non-increasing set function.*

Therefore, a function $h : 2^{[n]} \mapsto \mathbb{R}$ is submodular if, for any $a \in [n]$, the function $h_a : 2^{[n]\setminus\{a\}} \mapsto \mathbb{R}$, defined as $h_a(\mathcal{S}) \equiv h(\mathcal{S} \cup \{a\}) - h(\mathcal{S})$, is a non-increasing set function. This property is also called the *diminishing returns property.*

The first major result of this section follows, where we let

$$O_k \equiv \mathcal{O}_k^\top \mathcal{O}_k,$$

given an observation interval $[0, k]$.
Proposition 3 (Closed formula for the log det error as a sensor set function)**.** *Given any*

[6] For fixed and non-zero $\lambda_{\min}\left(L_k^\top L_k\right)$, the comments of this paragraph extend to the mmse(x_k) as well, for any finite k —on the other hand, if $\lambda_{\min}\left(L_k^\top L_k\right)$ varies with the system's size, since $\lambda_{\min}\left(L_k^\top L_k\right) \leq 1$, the number of sensors can increase sub-linearly with the system's size for fixed mmse(x_k).

finite observation interval $[0, k]$ *and non-zero* σ*, irrespective of Assumption 3,*

$$\log\det\left(\Sigma_{z_{k-1}}\right) = 2n(k+1)\log\left(\sigma\right) - \log\det\left(O_k + I\right). \tag{4.11}$$

Therefore, the $\log\det\left(\Sigma_{z_{k-1}}\right)$ depends on the sensor set through O_k. Now, the main result of this section follows, where we make explicit the dependence of O_k on the sensor set $\mathcal{S}$.

Theorem 4. The $\log\det$ **error is a supermodular and non-increasing set function with respect to the choice of the sensor set** *Given any finite observation interval* $[0, k]$*, the*

$$\log\det\left(\Sigma_{z_{k-1}}, \mathcal{S}\right) = 2n(k+1)\log\left(\sigma\right) - \log\det\left(O_{k,\mathcal{S}} + I\right) : \mathcal{S} \in 2^{[n]} \mapsto \mathbb{R}$$

is a supermodular and non-increasing set function with respect to the choice of the sensor set $\mathcal{S}$*.*

The above theorem states that for any finite observation interval, the $\log\det$ error of the Kalman filter is a supermodular and non-increasing set function with respect to the choice of the sensor set for any finite k. Hence, it exhibits the diminishing returns property: its rate of reduction with respect to newly placed sensors decreases as the cardinality of the already placed sensors increases. On the one hand, this property implies another fundamental design limit, in accordance to that of Theorem 3: adding new sensors, after a first few, becomes ineffective for the reduction of the estimation error. On the other hand, it also implies that greedy approach for solving $\mathcal{P}_1$ is effective [13, 40]. Thereby, we next use the results from the literature on submodular function maximization [96] and provide an efficient algorithm for $\mathcal{P}_1$.

4.5. Algorithms for Optimal Sensor Placement

In this section, we present our contributions with respect to Objective 3: $\mathcal{P}_1$ is combinatorial, and in Section 4.4 we proved that it involves the minimization of the supermodular set function $\log\det$ error. In particular, because the minimization of a general supermodular function is NP-hard [13], in this section we provide efficient approximation algorithms for the general solution of $\mathcal{P}_1$, along with their worst-case performance guarantees.

Specifically, we provide an efficient algorithm for $\mathcal{P}_1$ that returns a sensor set that satisfies the estimation bound of $\mathcal{P}_1$ and has cardinality up to a multiplicative factor from the minimum cardinality sensor sets that meet the same estimation bound. More importantly, this multiplicative factor depends only logarithmically on the problem's $\mathcal{P}_1$ parameters.

To this end, we first present a fact from the supermodular functions minimization literature that we use so to construct an approximation algorithm for $\mathcal{P}_1$ —we follow [94] for this material. In particular, consider the following problem, which is of similar structure to $\mathcal{P}_1$, where $h : 2^{[n]} \mapsto \mathbb{R}$ is a supermodular and non-increasing set function:

$$\begin{aligned} \underset{\mathcal{S}\subseteq[n]}{\text{minimize}} \quad & |\mathcal{S}| \\ \text{subject to} \quad & h(\mathcal{S}) \leq R. \end{aligned} \tag{$\mathcal{P}$}$$

The following greedy algorithm has been proposed for its approximate solution, for which, the subsequent fact is true.

Algorithm 7 Approximation Algorithm for $\mathcal{P}$.

Input: h, R.
Output: Approximate solution for $\mathcal{P}$.
  $\mathcal{S} \leftarrow \emptyset$
  while $h(\mathcal{S}) > R$ **do**
    $a_i \leftarrow a' \in \arg\max_{a\in[n]\setminus\mathcal{S}} \left(h(\mathcal{S}) - h(\mathcal{S}\cup\{a\})\right)$
    $\mathcal{S} \leftarrow \mathcal{S}\cup\{a_i\}$
  end while

Fact 3. *Denote as $\mathcal{S}^\star$ a solution to $\mathcal{P}$ and as $\mathcal{S}_0, \mathcal{S}_1, \ldots$ the sequence of sets picked by Algorithm 7. Moreover, let l be the smallest index such that $h(\mathcal{S}_l) \leq R$. Then,*

$$\frac{l}{|\mathcal{S}^\star|} \leq 1 + \log \frac{h([n]) - h(\emptyset)}{h([n]) - h(\mathcal{S}_{l-1})}.$$

For several classes of submodular functions, this is the best approximation factor one can achieve in polynomial time [13]. Therefore, we use this result to provide the approximation Algorithm 8 for $\mathcal{P}_1$, where we make explicit the dependence of $\log\det\left(\Sigma_{z_{k-1}}\right)$ on the selected sensor set $\mathcal{S}$. Moreover, its performance is quantified with Theorem 5.

Algorithm 8 Approximation Algorithm for $\mathcal{P}_1$.

For $h(\mathcal{S}) = \log\det\left(\Sigma_{z_{k-1}}, \mathcal{S}\right)$, where $\mathcal{S}\subseteq[n]$, Algorithm 8 is the same as Algorithm 7.

Theorem 5 (A Submodular Set Coverage Optimization for $\mathcal{P}_1$)**.** *Denote a solution to $\mathcal{P}_1$ as $\mathcal{S}^\star$ and the selected set by Algorithm 8 as $\mathcal{S}$. Then,*

$$\log\det\left(\Sigma_{z_{k-1}}, \mathcal{S}\right) \leq R, \tag{4.12}$$

$$\begin{aligned} \frac{|\mathcal{S}|}{|\mathcal{S}^\star|} &\leq 1 + \log \frac{\log\det\left(\Sigma_{z_{k-1}}, \emptyset\right) - \log\det\left(\Sigma_{z_{k-1}}, [n]\right)}{R - \log\det\left(\Sigma_{z_{k-1}}, [n]\right)} \\ &\equiv F_i, \end{aligned} \tag{4.13}$$

where $\log\det\left(\Sigma_{z_{k-1}}, \emptyset\right) \leq n(k+1)\log(\sigma^2)$. Finally, the computational complexity of Algorithm 8 is $O(n^2(nk)^3)$.

Therefore, Algorithm 8 returns a sensor set that meets the estimation bound of $\mathcal{P}_1$. Moreover, the cardinality of this set is up to a multiplicative factor of F_i from the minimum cardinality sensor sets that meet the same estimation bound —that is, F_i is a worst-case approximation guarantee for Algorithm 8. Additionally, F_i depends only logarithmically on

the problem's $\mathcal{P}_1$ parameters. Finally, the dependence of F_i on n, R and σ^2 is expected from a design perspective: increasing the network size n, requesting a better estimation guarantee by decreasing R, or incurring a noise of greater variance, should all push the cardinality of the selected sensor set upwards.

4.6. Concluding Remarks & Future Work

We considered a linear time-variant system and studied the properties of its Kalman estimator given an observation interval and a sensor set. Our contributions were threefold. First, in Section 4.3 we presented several design limits. For example, we proved that the number of sensors grows linearly with the system's size for fixed minimum mean square error; this is a fundamental limit, especially for systems where the system's size is large. Second, in Section 4.4 we proved that the log det error is a supermodular and non-increasing set function with respect to the choice of the sensor set. Third, in Section 4.5, we used this result to provide an efficient approximation algorithm for the solution of $\mathcal{P}_1$, along with its worst-case performance guarantees. Our future work is focused on extending the results of this chapter to the problem of sensor scheduling.

4.7. Appendix: Proof of Results

• **Theorem 3**

Proof: We first prove the lower bound in (4.9): observe first that $\text{mmse}(x_0) \geq \text{mmse}(x_0)^{w.=0}$, where $\text{mmse}(x_0)^{w.=0}$ is the minimum mean square error of x_0 when the process noise w_k in (4.1) is zero for all $k \geq 0$. To express $\text{mmse}(x_0)^{w.=0}$ in a closed form similar to (4.11), note that in this case (4.2) becomes $\bar{y}_k = \tilde{\mathcal{O}}_k x_0 + \bar{v}_k$, where $\tilde{\mathcal{O}}_k \equiv \left[C_0^\top, \Phi_1^\top C_1^\top, \ldots, \Phi_k^\top C_k^\top\right]^\top$ and $\Phi_m \equiv A_{m-1} \cdots A_0$, for $m > 0$, and $\Phi_m \equiv I$, for $m = 0$. Thereby, from Corollary E.3.5 of [123], the minimum mean square linear estimate of x_0, denoted as $\hat{x}_{k_0}^{w.=0}$, has error covariance

$$\begin{aligned}\Sigma_{k_0}^{w.=0} &\equiv \mathbb{E}\left((x_0 - \hat{x}_{k_0}^{w.=0})(x_0 - \hat{x}_{k_0}^{w.=0})^\top\right) \\ &= \sigma^2\left(I - \tilde{\mathcal{O}}_k^\top\left(\tilde{\mathcal{O}}_k\tilde{\mathcal{O}}_k^\top + I\right)^{-1}\tilde{\mathcal{O}}_k\right), \end{aligned} \tag{4.14}$$

and minimum mean square error

$$\begin{aligned}\text{mmse}(x_0)^{w.=0} &\equiv \text{tr}\left(\Sigma_{k_0}^{w.=0}\right) \\ &= \sigma^2\text{tr}\left[\left(\tilde{\mathcal{O}}_k^\top\tilde{\mathcal{O}}_k + I\right)^{-1}\right] \qquad (4.15) \\ &\equiv \sigma^2\text{tr}\left[\left(\tilde{O}_k + I\right)^{-1}\right], \qquad (4.16)\end{aligned}$$

where we deduce (4.15) from (4.14) using the Woodbury matrix identity (Corollary 2.8.8 of [93]), and (4.16) from (4.15) using the notation $\tilde{O}_k \equiv \tilde{\mathcal{O}}_k^\top\tilde{\mathcal{O}}_k$. In particular, $\tilde{O}_k$ is the observability matrix $\tilde{O}_k = \sum_{m=0}^{k-1}\Phi_m^\top C_k^\top C_k \Phi_m$ of (4.1) ([85]).

Hence, $\text{mmse}(x_0) \geq \sigma^2 \text{tr}\left[\left(\tilde{O}_k + I\right)^{-1}\right]$, and since the arithmetic mean of a finite set of positive numbers is at least as large as their harmonic mean, using (4.16),

$$\text{mmse}(x_0) \geq \frac{n^2\sigma^2}{\text{tr}\left(\tilde{O}_k + I\right)} \geq \frac{n^2\sigma^2}{\text{tr}\left(\tilde{O}_k\right) + n}.$$

Now, for $i \in [n]$, let $I^{(i)}$ be the $n \times n$ matrix where I_{ii} is one, while I_{jk} is zero, for all $(j,k) \neq (i,i)$. Then, $\text{tr}(\tilde{O}_k) = \text{tr}\left(\sum_{m=0}^{k} \Phi_m^\top C^\top C \Phi_m\right) = \sum_{i=1}^{n} s_i \text{tr}\left(\sum_{m=0}^{k} \Phi_m^\top I^{(i)} \Phi_m\right)$; now,

$$\text{tr}\left(\sum_{m=0}^{k} \Phi_m^\top I^{(i)} \Phi_m\right) \leq n\lambda_{\max}\left(\sum_{m=0}^{k} \Phi_m^\top I^{(i)} \Phi_m\right) = n\|\sum_{m=0}^{k} \Phi_m^\top I^{(i)} \Phi_m\|_2 \leq n\sum_{m=0}^{k} \|\Phi_m\|_2^2,$$

because $\|I^{(i)}\|_2 = 1$, and from the definition of Φ_m and Proposition 9.6.1 of [93],

$$\sum_{m=0}^{k} \|\Phi_m\|_2^2 \leq \frac{1-\mu^{2(k+1)}}{1-\mu^2}.$$

Therefore, $\text{tr}(\tilde{O}_k) \leq \sum_{i=1}^{n} s_i n \frac{1-\mu^{2(k+1)}}{1-\mu^2} = n|\mathcal{S}|\frac{1-\mu^{2(k+1)}}{1-\mu^2}$, and as a result, the lower bound in (4.9) for $\text{mmse}(x_0)$ follows.

Next, we prove the upper bound in (4.9), using (4.19), which is proved in the proof of Proposition 3, and (4.6) for $k' = 0$: $O_k + I \succeq \sigma^2 I$, and as a result, from Proposition 8.5.5 of [93], $(O_k + I)^{-1} \preceq \sigma^{-2} I$. Hence, $\text{mmse}(x_0) \leq \text{tr}\left[L_0 \sigma^2 I L_0^\top\right] \leq n\sigma^2$.

Finally, to derive the lower and upper bounds for $\text{mmse}(x_k)$, observe that $\text{mmse}(x_0) \leq \text{mmse}(z_{k-1})$ and $\text{mmse}(z_{k-1}) \leq n(k+1)\sigma^2$ —the proof follows using similar steps as above. Then, from Theorem 1 of [125],

$$\lambda_{\min}\left(L_k^\top L_k\right)\text{mmse}(z_{k-1}) \leq \text{mmse}(x_k) \leq \lambda_{\max}\left(L_k^\top L_k\right)\text{mmse}(z_{k-1}).$$

The combination of these inequalities completes the proof. ■

• **Proposition 3**

Proof: From $\hat{x}_i = L_{i-1}\hat{z}_{k-1}$,

$$\text{mmse}(x_i) = \text{tr}(L_{i-1}\Sigma_{z_{k-1}}L_{i-1}^\top).$$

Also,

$$\Sigma_{z_{k-1}} = \sigma^2 \left(I - \mathcal{O}_k^\top \left(\mathcal{O}_k \mathcal{O}_k^\top + I \right)^{-1} \mathcal{O}_k \right) \tag{4.17}$$

$$= \sigma^2 \left(\mathcal{O}_k^\top \mathcal{O}_k + I \right)^{-1} \tag{4.18}$$

$$= \sigma^2 \left(O_k + I \right)^{-1}, \tag{4.19}$$

where we deduce (4.18) from (4.17) using the Woodbury matrix identity (Corollary 2.8.8 of [93]), and (4.19) from (4.18) using the fact that $O_k = \mathcal{O}_k^\top \mathcal{O}_k$. ■

• **Theorem 4**

Proof: To prove that the mmse(x_i) is non-increasing, observe that

$$O_{k,\mathcal{S}} = \sum_{m=1}^{n} s_m \sum_{j=0}^{k} M_j^\top I^{(m)} M_j = \sum_{m=1}^{n} s_m O_{k,\{m\}}, \tag{4.20}$$

where M_j is the $n \times nk$ matrix

$$M_j \equiv [L_{j-1}, 0].$$

Then, for any $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subseteq [n]$, (4.20) and that fact that $O_{k,\{1\}}, O_{k,\{2\}}, \ldots, O_{k,\{n\}} \succeq 0$ imply $O_{k,\mathcal{S}_1} \preceq O_{k,\mathcal{S}_2}$, and as a result, $O_{k,\mathcal{S}_1} + I \preceq O_{k,\mathcal{S}_2} + I$. Therefore, from Proposition 8.5.5 of [93],

$$\left(O_{k,\mathcal{S}_2} + I \right)^{-1} \preceq \left(O_{k,\mathcal{S}_1} + I \right)^{-1},$$

This implies

$$L_{i-1} \left(O_{k,\mathcal{S}_2} + I \right)^{-1} L_{i-1}^\top \preceq L_{i-1} \left(O_{k,\mathcal{S}_1} + I \right)^{-1} L_{i-1}^\top,$$

and as a result, mmse(x_i) is non-increasing.

Next, observe that

$$\mathrm{tr}\left[L_{i-1} \left(O_{k,\mathcal{S}} + I \right)^{-1} L_{i-1}^\top \right] = \mathrm{tr}\left[\left(O_{k,\mathcal{S}} + I \right)^{-1} L_{i-1}^\top L_{i-1} \right],$$

and consider the eigenvector decomposition of $L_{i-1}^\top L_{i-1}$, $\sum_{m=1}^{n} \lambda_m q_m q_m^\top$, where λ_m and q_m is the m-th eigenvalue and eigenvector of $L_{i-1}^\top L_{i-1}$, respectively. Thereby,

$$\begin{aligned} &\mathrm{tr}\left[L_{i-1} \left(O_{k,\mathcal{S}} + I \right)^{-1} L_{i-1}^\top \right] \\ &\qquad = \sum_{m=1}^{n} \lambda_m \mathrm{tr}\left[\left(O_{k,\mathcal{S}} + I \right)^{-1} q_m q_m^\top \right] \\ &\qquad = \sum_{m=1}^{n} \lambda_m q_m^\top \left(O_{k,\mathcal{S}} + I \right)^{-1} q_m. \end{aligned}$$

Since $L_{i-1}^{\top} L_{i-1} \succeq 0$, $\lambda_m \geq 0$, for all $m \in [n]$. Therefore, since the non-negative sum of supermodular set functions is a supermodular set function, it remains to prove that for any $q \in \mathbb{R}^n$, $q^{\top} \left(O_{k,\mathcal{S}} + I\right)^{-1} q$ is such. This follows from the proof of Proposition 2 of [110], and the proof is complete. ■

• **Theorem 5**

Proof: First, let $\mathcal{S}_0, \mathcal{S}_1, \ldots$ be the sequence of sets selected by Algorithm 8 and l the smallest index such that $\text{mmse}(x_i, \mathcal{S}_l) \leq R$. Therefore, $\mathcal{S}_l$ is the set that Algorithm 8 returns, and this proves (4.12).

Moreover, from Fact 3,

$$\begin{aligned} \frac{l}{|\mathcal{S}^{\star}|} &\leq 1 + \log \frac{h([n]) - h(\emptyset)}{h([n]) - h(\mathcal{S}_{l-1})} \\ &= 1 + \log \frac{\text{mmse}(x_i, \emptyset) - \text{mmse}(x_i, [n])}{\text{mmse}(x_i, \mathcal{S}_{l-1}) - \text{mmse}(x_i, [n])}. \end{aligned}$$

Now, l is the first time that $\text{mmse}(x_i, \mathcal{S}_l) \leq R$, and a result $\text{mmse}(x_i, \mathcal{S}_{l-1}) > R$. This implies (4.13).

Furthermore, for $i = 0$, $\text{mmse}(x_0, \emptyset) = n\sigma^2$. On the other hand, for $i = k$, first set for $m \geq j \geq 0$, $\Phi_{m,j} \equiv A_m A_{m-1} \cdots A_j$ and $\Phi_{m,m+1} \equiv I$; then,

$$\begin{aligned} \text{mmse}(x_k, \emptyset) &= \sigma^2 \text{tr}\left(L_{k-1} L_{k-1}^{\top}\right) \\ &= \sigma^2 \text{tr}\left(\sum_{m=0}^{k} \Phi_{k-1,m}^{\top} \Phi_{k-1,m}\right) \\ &\leq n\sigma^2 \lambda_{\max}\left(\sum_{m=0}^{k} \Phi_{k-1,m}^{\top} \Phi_{k-1,m}\right) \\ &= n\sigma^2 \| \sum_{m=0}^{k} \Phi_{k-1,m}^{\top} \Phi_{k-1,m} \|_2 \\ &\leq n\sigma^2 \sum_{m=0}^{k} \|\Phi_{k-1,m}\|_2^2 \leq n\sigma^2 \frac{1 - \mu^{2(k+1)}}{1 - \mu^2}. \end{aligned}$$

Finally, with respect to the computational complexity of Algorithm 8, note that the `while` loop is repeated for at most n times. Moreover, the complexity to invert an $nk \times nk$ matrix, using Gauss-Jordan elimination decomposition, is $O((nk)^3)$ (this is also the complexity to multiply two such matrices). Additionally, at most n matrices must be inverted so that the $\arg\max_{a \in [n] \setminus \mathcal{S}} \left(\text{mmse}(x_i, \mathcal{S}) - \text{mmse}(x_i, \mathcal{S} \cup \{a\})\right)$ can be computed. Furthermore, $O(n)$ time is required to find a maximum element between n available. Therefore, the computational complexity of Algorithm 8 is $O(n^2 (nk)^3)$. ■

CHAPTER 5 : Near-optimal sensor scheduling for batch state estimation: Complexity, algorithms, and limits

In this chapter, we focus on batch state estimation for linear systems. This problem is important in applications such as environmental field estimation, robotic navigation, and target tracking. Its difficulty lies on that limited operational resources among the sensors, e.g., shared communication bandwidth or battery power, constrain the number of sensors that can be active at each measurement step. As a result, sensor scheduling algorithms must be employed. Notwithstanding, current sensor scheduling algorithms for batch state estimation scale poorly with the system size and the time horizon. In addition, current sensor scheduling algorithms for Kalman filtering, although they scale better, provide no performance guarantees or approximation bounds for the minimization of the batch state estimation error. In this chapter, one of our main contributions is to provide an algorithm that enjoys both the estimation accuracy of the batch state scheduling algorithms and the low time complexity of the Kalman filtering scheduling algorithms. In particular: 1) our algorithm is near-optimal: it achieves a solution up to a multiplicative factor $1/2$ from the optimal solution, and this factor is close to the best approximation factor $1/e$ one can achieve in polynomial time for this problem; 2) our algorithm has (polynomial) time complexity that is not only lower than that of the current algorithms for batch state estimation; it is also lower than, or similar to, that of the current algorithms for Kalman filtering. We achieve these results by proving two properties for our batch state estimation error metric, which quantifies the square error of the minimum variance linear estimator of the batch state vector: a) it is supermodular in the choice of the sensors; b) it has a sparsity pattern (it involves matrices that are block tri-diagonal) that facilitates its evaluation at each sensor set.[1]

5.1. Introduction

Search and rescue [126], environmental field estimation [127], robotic navigation [128], and target tracking [129] are only a few of the challenging information gathering problems that employ the monitor capabilities of sensor networks [130]. In particular, all these problems face the following three main challenges:

- they involve systems whose evolution is largely unknown, corrupted with noisy inputs [129], and sensors with limited sensor capabilities, corrupted with measurement noise [103].
- they involve systems that change over time [127], and as a result, necessitate both spacial and temporal deployment of sensors in the environment. At the same time:
- they involve operational constraints, such as limited bandwidth and battery life, which limit the number of sensors that can be simultaneously used (i.e., be switched-on) in the information gathering process [131].

As a result of these challenges, researchers focused on the following question: "How do we select at each measurement step only a few sensors so to minimize the estimation error despite the above challenges?" The effort to answer this question resulted to the problem

[1] This chapter is based on the paper by Tzoumas et al. [58].

of sensor scheduling [131]: in particular, sensor scheduling offers a formal methodology to use at each measurement time only a few sensors and obtain an optimal trade-off between the estimation accuracy and the usage of the limited operational resource (e.g., the shared bandwidth). Clearly, sensor scheduling is a combinatorial problem of exponential complexity [130].

In this chapter, we focus on the following instance of this problem:

Problem 1 (Sensor Scheduling for Minimum Variance Batch State Estimation) *Consider a time-invariant linear system, whose state at time t_k is denoted as $x(t_k)$, a set of m sensors, and a fixed set of K measurement times $t_1, t_2, \ldots, t_K$. In addition, consider that at each t_k at most r_k sensors can be used, where $r_k \leq m$. At each t_k select a set of r_k sensors so to minimize the square estimation error of the minimum variance linear estimator of the batch state vector $(x(t_1), x(t_2), \ldots, x(t_K))$.*

There are two classes of sensor scheduling algorithms, that trade-off between the estimation accuracy of the batch state vector and their time complexity: these for Kalman filtering, and those for batch state estimation. In more detail:

Kalman filtering algorithms: These algorithms sacrifice estimation accuracy over reduced time complexity. The reason is that they are sequential algorithms: at each t_k, they select the sensors so to minimize the square estimation error of the minimum variance linear estimator of $x(t_k)$ (given the measurements up to t_k). Therefore, their objective is to minimize the sum of the square estimation errors of $x(t_k)$ across the measurement times t_k [132]. However, this sum is only an upper bound to the square estimation error of the batch state vector $(x(t_1), x(t_2), \ldots, x(t_K))$. Thus, the Kalman filtering algorithms lack on estimation accuracy with respect to the batch state estimation algorithms.

Batch state estimation algorithms: These algorithms sacrifice time complexity over estimation accuracy. The reason is that they perform global optimization, in accordance to Problem 1. Therefore, however, they lack on time complexity with respect to the Kalman filtering algorithms.

Notwithstanding, in several recent robotic applications, batch estimation algorithms have been proven competitive in their time complexity to their filtering counterparts [22, 133]. The reason is that sparsity patterns emerge in these applications, that reduce the time complexity of their batch estimation algorithms to an order similar to that of the filtering algorithms [134]. Thereby, the following question on Problem 1 arises:

Question 1. *"Is there an algorithm for Problem 1 that enjoys both the estimation accuracy of the batch state algorithms and the low time complexity of the Kalman filtering algorithms?"*

Literature review on sensor scheduling algorithms for batch state estimation. The most relevant paper on Problem 1 is [135], where an algorithm based on convex relaxation is provided. This algorithm scales poorly with the system's size and number of measurement times. In addition, it provides no approximation performance guarantees.

Literature review on sensor scheduling algorithms for Kalman filtering. Several papers in this category have focused on myopic algorithms [115]; such algorithms, however, often perform poorly [136]. Other papers have focused on algorithms that use: tree pruning [137], convex optimization [119], quadratic programming [138], or submodular function maximization [5, 139]. Nevertheless, these algorithms provide no performance guarantees on the batch state estimation error, or have time complexity that scales poorly with the system's size and number of measurement times [137] [138]. To reduce the time complexity of these algorithms, papers have also proposed periodic sensor schedules [132].

Contributions. We now present our contributions:

1) We prove that Problem 1 is NP-hard.

2) We provide an algorithm for Problem 1 (Algorithm 1) that answers Question 1 positively. The reasons are two:

   i) Algorithm 1 is near-optimal: it achieves a solution that is up to a multiplicative factor $1/2$ from the optimal solution. In addition, this multiplicative factor is close to the factor $1/e$ which we prove to be the best approximation factor one can achieve in polynomial time for Problem 1 in the worst-case.

   ii) Algorithm 1 has (polynomial) time complexity that is not only lower than that of the state of the art scheduling algorithms for batch state estimation; it is also lower than, or similar to, that of the state of the art scheduling algorithms for Kalman filtering. For example, it has similar complexity to the state of the art periodic scheduling algorithm in [132] (in particular: lower for K large enough), and lower than the complexity of the algorithm in [119].

   Overall, in response to Question 1, Algorithm 1 enjoys both the higher estimation accuracy of the batch state estimation approach (compared to the Kalman filtering approach, that only approximates the batch state estimation error with an upper bound) and the low time complexity of Kalman filtering approach.

3) We prove limits on the minimization of the square error of the minimum variance estimator of $(x(t_1),\, x(t_2), \ldots, x(t_K))$ with respect to the scheduled sensors. For example, we prove that the number r_k of used sensors at each measurement time must increase linearly with the system size for fixed estimation error and number of measurement times K; this is a fundamental limit, especially for large-scale systems.

Our technical contributions. We achieve our aforementioned contributions by proving the following two:

Supermodularity in Problem 1: We prove that our estimation metric, that quantifies the square error of the minimum variance estimator of $(x(t_1),\, x(t_2), \ldots, x(t_K))$, is a supermodular function in the choice of the used sensors. This result becomes important when we compare it to results on the multi-step Kalman filtering that show that the corresponding

estimation metric in this case is neither supermodular nor submodular [5, 139].[2]

In addition, this submodularity result cannot be reduced to the batch estimation problem in [114]. The main reasons are two: i) we consider sensors that can measure any linear combination of the element of $x(t_k)$, in contrast to [114], where each sensor measures directly only one element of $x(t_k)$. Nonetheless, the latter assumption is usually infeasible in dynamical systems [85]; ii) our error metric is relevant to estimation problems for dynamical systems and different to the submodular information gain considered in [114].

Sparsity in Problem 1: We identify a sparsity pattern in our error metric, that facilitates the latter's evaluation at each sensor set. In particular, we prove that the error covariance of the minimum variance linear estimator of the batch state vector is block tri-diagonal.

We organize the rest of the chapter as follows: In Section 5.2 we present formally Problem 1. In Section 5.3, we present in three subsections our main results: in Section 5.3.1, we prove that our sensor scheduling problem is NP-hard. In Section 5.3.2, we derive our near-optimal approximation algorithm. In Section 5.3.3, we prove limits on the minimization of the batch state estimation error with respect to the used sensors. Section 5.4 concludes the chapter with our future work.[3]

5.2. Problem Formulation

In the following paragraphs, we present our sensor scheduling problem for batch state estimation. To this end, we first build our system and measurement framework. Then, we define our sensor scheduling framework and, finally, present our sensor scheduling problem.

We start in more detail with the system model:

System Model. *We consider the linear time-invariant system:*

$$\dot{x}(t) = Ax(t) + Bu(t) + Fw(t), t \geq t_0, \tag{5.1}$$

where t_0 is the initial time, $x(t) \in \mathbb{R}^n$ ($n \in \mathbb{N}$) the state vector, $\dot{x}(t)$ the time derivative of $x(t)$, $u(t)$ the exogenous input, and $w(t)$ the process noise. The system matrices A, B and F are of appropriate dimensions. We consider that $u(t)$, A, B and F are known. Our main assumption on $w(t)$ is found in Assumption 4, that is presented after our measurement model.

Remark 1. *Our results extend to continuous and discrete time-variant systems, as explained*

[2] The observation of [5] is also important as it disproves previous results in the literature [140].

[3] *Standard notation is presented in this footnote:* We denote the set of natural numbers $\{1, 2, \ldots\}$ as $\mathbb{N}$, the set of real numbers as $\mathbb{R}$, and the set $\{1, 2, \ldots, n\}$ as $[n]$ ($n \in \mathbb{N}$). The empty set is denoted as $\emptyset$. Given a set $\mathcal{X}$, $|\mathcal{X}|$ is its cardinality. Matrices are represented by capital letters and vectors by lower-case letters. We write $A \in \mathcal{X}^{n_1 \times n_2}$ ($n_1, n_2 \in \mathbb{N}$) to denote a matrix of n_1 rows and n_2 columns whose elements take values in $\mathcal{X}$. Moreover, for a matrix A, $A^\top$ is its transpose, and $[A]_{ij}$ is its element at the i-th row and j-th column. In addition, $\|A\|_2 \equiv \sqrt{A^\top A}$ is its spectral norm, and $\det(A)$ its determinant. Furthermore, if A is positive semi-definite or positive definite, we write $A \succeq 0$ and $A \succ 0$, respectively. I is the identity matrix; its dimension is inferred from the context. Similarly for the zero matrix 0. Finally, for a random variable $x \in \mathbb{R}^n$, $\mathbb{E}(x)$ is its expected value, and $\mathbb{C}(x)$ its covariance.

in detail in Section 5.3 (Corollaries 4 and 5).

We introduce the measurement model:

Measurement Model. *We consider m sensors:*

$$z_i(t) = C_i x(t) + v_i(t), i \in [m], \tag{5.2}$$

where $z_i(t)$ is the measurement taken by sensor i at time t, $C_i \in \mathbb{R}^{d_i \times n}$ ($d_i \in \mathbb{N}$) is sensor's i measurement matrix, and $v_i(t)$ is its measurement noise.

We make the following assumption on $x(t_0)$, $w(t)$ and $v_i(t)$:
Assumption 4. *For all t, $t' \geq t_0$, $t \neq t'$, and all $i \in [m]$: $x(t_0)$, $w(t)$, $w(t')$, $v_i(t)$ and $v_i(t')$ are uncorrelated; in addition, $x(t_0)$, $w(t)$ and $v_i(t)$ have positive definite covariance.*

We now introduce the sensor scheduling model:

Sensor Scheduling Model. *The m sensors in (5.2) are used at K scheduled measurement times $\{t_1, t_2, \ldots, t_K\}$. Specifically, at each t_k only r_k of these m sensors are used ($r_k \leq m$), resulting in the batch measurement vector $y(t_k)$:*

$$y(t_k) = S(t_k) z(t_k), k \in [K], \tag{5.3}$$

where $z(t_k) \equiv (z_1^\top(t_k), z_2^\top(t_k), \ldots, z_m^\top(t_k))^\top$, and $S(t_k)$ is the sensor selection matrix: it is a block matrix, composed of matrices $[S(t_k)]_{ij}$ ($i \in [r_k]$, $j \in [m]$) such that $[S(t_k)]_{ij} = I$ if sensor j is used at t_k, and $[S(t_k)]_{ij} = 0$ otherwise. We consider that each sensor can be used at most once at each t_k, and as a result, for each i there is one j such that $[S(t_k)]_{ij} = I$ while for each j there is at most one i such that $[S(t_k)]_{ij} = I$.

We now present the sensor scheduling problem we study in this chapter. To this end, we use two notations:

Notation. First, we set $\mathcal{S}_k \equiv \{j : \text{there exists } i \in [r_k], [S(t_k)]_{ij} = 1\}$; that is, $\mathcal{S}_k$ is the set of indices that correspond to used sensors at t_k. Second, we set $\mathcal{S}_{1:K} \equiv (\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_K)$.

Problem 1 (Sensor Scheduling for Minimum Variance Batch State Estimation) *Given a set of measurement times $t_1, t_2, \ldots, t_K$, select at each t_k to use a subset of r_k sensors, out of the m sensors in (5.2), so to minimize the* log det *of the error covariance of the minimum variance linear estimator of $x_{1:K} \equiv (x(t_1), x(t_2), \ldots, x(t_K))$. In mathematical notation:*

$$\begin{aligned} &\underset{\mathcal{S}_k \subseteq [m], k \in [K]}{\textit{minimize}} && \log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})) \\ &\textit{subject to} && |\mathcal{S}_k| \leq r_k, k \in [K], \end{aligned}$$

where $\hat{x}_{1:K}$ is the minimum variance linear estimator of $x_{1:K}$, and $\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})$ its error covariance given $\mathcal{S}_{1:K}$.

Two remarks follow on the definition of Problem 1. In the first remark we explain why we

focus on $\hat{x}_{1:K}$, and in the second why we focus on $\log\det(\Sigma(\hat{x}_{1:K}))$.

Notation. For notational simplicity, we use $\Sigma(\hat{x}_{1:K})$ and $\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})$ interchangeably.
Remark 2. *We focus on the minimum variance linear estimator $\hat{x}_{1:K}$ because of its optimality: it minimizes among all linear estimators of $x_{1:K}$ the estimation error $\mathbb{E}(\|x_{1:K}-\hat{x}_{1:K}\|_2^2)$, where the expectation is taken with respect to $y(t_1), y(t_2), \ldots, y(t_K)$ [103]. Because $\hat{x}_{1:K}$ is also unbiased (that is, $\mathbb{E}(\hat{x}_{1:K}) = x_{1:K}$, where the expectation is taken with respect to $y(t_1), y(t_2), \ldots, y(t_K)$),* we equivalently say that $\hat{x}_{1:K}$ is the minimum variance estimator of $x_{1:K}$.

We compute the error covariance of $\hat{x}_{1:K}$ in Appendix 5.5.1.
Remark 3. *We focus on the estimation error metric $\log\det(\Sigma(\hat{x}_{1:K}))$ because when it is minimized the probability that the estimation error $\|x_{1:K}-\hat{x}_{1:K}\|_2^2$ is small is maximized. To quantify this statement, we note that this error metric is related to the η-confidence ellipsoid of $x_{1:K}-\hat{x}_{1:K}$ [119]: Specifically, the η-confidence ellipsoid is the minimum volume ellipsoid that contains $x_{1:K}-\hat{x}_{1:K}$ with probability η, that is, it is the $\mathcal{E}_\epsilon \equiv \{x : x^\top \Sigma(\hat{x}_{1:K})x \leq \epsilon\}$, where ϵ is the quantity $F^{-1}_{\chi^2_{n(k+1)}}(\eta)$, and $F_{\chi^2_{n(k+1)}}$ the cumulative distribution function of a χ-squared random variable with $n(k+1)$ degrees of freedom [124]. Thus, its volume*

$$vol(\mathcal{E}_\epsilon) \equiv \frac{(\epsilon\pi)^{n(k+1)/2}}{\Gamma\left(n(k+1)/2+1\right)} \det\left(\Sigma(\hat{x}_{1:K})^{1/2}\right), \tag{5.4}$$

where $\Gamma(\cdot)$ denotes the Gamma function [124], quantifies the estimation error of the optimal estimator $\hat{x}_{1:K}$. Therefore, by taking the logarithm of (5.4), we validate that when the $\log\det(\Sigma(\hat{x}_{1:K}))$ is minimized the probability that the estimation error $\|x_{1:K}-\hat{x}_{1:K}\|_2^2$ is small is maximized.

5.3. Main Results

Our main results are presented in three sections:

- In Section 5.3.1, we prove that Problem 1 is NP-hard.
- In Section 5.3.2, we derive a provably near-optimal approximation algorithm for Problem 1. In addition, we emphasize on its time complexity and compare it to that of existing sensor scheduling algorithms for two categories: batch state estimation, and Kalman filtering.
- In Section 5.3.3, we prove limits on the optimization of the estimation error $\mathbb{E}(\|x_{1:K}-\hat{x}_{1:K}\|_2^2)$ with respect to the scheduled sensors.

5.3.1. Computational Complexity of Sensor Scheduling for Batch State Estimation

In this section, we characterize the computational complexity of Problem 1. In particular, we prove:
Theorem 6. *The problem of sensor scheduling for minimum variance batch state estimation (Problem 1) is NP-hard.*

Algorithm 9 Approximation algorithm for Problem 1.

Input: Number of measurement times K, scheduling constraints $r_1, r_2, \ldots, r_K$, estimation error function $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})) : \mathcal{S}_k \subseteq [m], k \in [K] \mapsto \mathbb{R}$

Output: Sensor sets $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_K)$ that approximate the solution to Problem 1, as quantified in Theorem 7

$k \leftarrow 1$, $\mathcal{S}_{1:0} \leftarrow \emptyset$

while $k \leq K$ **do**

1. Apply Algorithm 10 to

$$\min_{S \subseteq [m]} \{\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:k-1}, \mathcal{S})) : |\mathcal{S}| \leq r_k\} \tag{5.5}$$

2. Denote as $\mathcal{S}_k$ the solution Algorithm 10 returns
3. $\mathcal{S}_{1:k} \leftarrow (\mathcal{S}_{1:k-1}, \mathcal{S}_k)$
4. $k \leftarrow k + 1$

end while

Proof: The proof is omitted due to space constraints. Notwithstanding, we note that the proof is complete by finding an instance of Problem 1 that is equivalent to the NP-hard minimal observability problem introduced in [7] [109]. ■

Due to Theorem 6, for the polynomial time solution of Problem 1 we need to appeal to approximation algorithms. To this end, in Section 5.3.2, we provide an efficient provably near-optimal approximation algorithm:

5.3.2. Algorithm for Sensor Scheduling for Minimum Variance Batch State Estimation

We propose Algorithm 9 for Problem 1 (Algorithm 9 uses Algorithm 10 as a subroutine); with the following theorem, we quantify its approximation performance and time complexity.

Theorem 7. *The theorem has two parts:*

1) Approximation performance of Algorithm 9: *Algorithm 9 returns sensors sets* $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_K$ *that:*

- *satisfy all the feasibility constraints of Problem 1:* $|\mathcal{S}_k| \leq r_k, k \in [K]$
- *achieve an error value* $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K}))$*, where* $\mathcal{S}_{1:K} \equiv (\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_K)$*, such that:*

$$\frac{\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})) - OPT}{MAX - OPT} \leq \frac{1}{2}, \tag{5.6}$$

where OPT *is the (optimal) value to Problem 1, and* MAX *is the maximum (worst) value to Problem 1* ($MAX \equiv \max_{\mathcal{S}'_{1:K}} \log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}'_{1:K}))$).

2) Time complexity of Algorithm 9: *Algorithm 9 has time complexity of order:*

$$O(n^{2.4} K \sum_{k=1}^{K} r_k^2).$$

Theorem 7 extends to continuous and discrete time-variant systems as follows:
Corollary 4. *Consider the time-variant version of* (5.1)*:*

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + F(t)w(t), t \geq t_0. \tag{5.7}$$

1) Part 1 of Theorem 7 holds.

2) Part 2 of Theorem 7 holds if the time complexity for computing each transition matrix $\Phi(t_{k+1}, t_k)$ *[85], where* $k \in [K-1]$*, is* $O(n^3)$*.*[4]

Corollary 5. *Consider the discrete time version of* (5.7)*:*

$$x[k+1] = A_k x[k] + B_k u[k] + F_k w[k], k \geq k_0. \tag{5.8}$$

Similarly, consider the discrete time counterparts of the sensor model (5.2)*, Assumption 4, and the sensor scheduling model* (5.3)*.*

1) Part 1 of Theorem 7 holds.

2) Part 2 of Theorem 7 holds if A_k *in* (5.8) *is full rank for all* $k \in [K]$*.*

We follow-up with several remarks on Theorem 7:
Remark 4. (Approximation quality of Algorithm 9) *Theorem 7 quantifies the worst-case performance of Algorithm 9 across all values of Problem 1's parameters. The reason is that the right-hand side of* (5.6) *is constant. In particular,* (5.6) *guarantees that for any instance of Problem 1, the distance of the approximate value* $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K}))$ *from OPT is at most* $1/2$ *the distance of the worst (maximum) value MAX from OPT. In addition, this approximation factor is near to the optimal approximation factor* $1/e \cong .38$ *one can achieve in the worst-case for Problem 1 in polynomial time [141]; the reason is twofold: first, as we comment in the next paragraph, we prove that Problem 1 involves the minimization of a non-increasing and supermodular function [96], and second, as we proved in Section 5.3.1, Problem 1 is in the worst-case equivalent to the minimal controllability problem introduced in [7], which cannot be approximated in polynomial time with a better factor than the* $1/e$ *[13].*
Remark 5. (Supermodularity of $\log\det(\Sigma(\hat{x}_{1:K}))$) *In the proof of Theorem 7 (Appendix 5.5.2), we show that* $\log\det(\Sigma(\hat{x}_{1:K}))$ *is a non-increasing and supermodular function with respect to the sequence of selected sensors. Specifically, the proof of* (5.6) *follows by combining these two results and results on the maximization of submodular functions over matroid constraints [12] —we present these three derivations in Appendices 5.5.2, 5.5.2, and 5.5.2, respectively.*

We continue with our third remark on Theorem 7:
Remark 6. (Time complexity of Algorithm 9) *Algorithm 9's time complexity is broken down into two parts: the first part is the number of evaluations of* $\log\det(\Sigma(\hat{x}_{1:K}))$ *required by the algorithm, and the second part is the time complexity of each such evaluation. In particular, Algorithm 9 requires at most* r_k^2 *evaluations of* $\log\det(\Sigma(\hat{x}_{1:K}))$ *at each* t_k*. Therefore, Algo-*

[4]The matrices $\Phi(t_{k+1}, t_k)$, where $k \in [K-1]$, are used in the computation of $\Sigma(\hat{x}_{1:K})$ (cf. proof of Theorem 7 in Appendix 5.5.2).

Algorithm 10 Single step greedy algorithm (subroutine in Algorithm 9).

Input: Current iteration k (corresponds to t_k), selected sensor sets $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{k-1})$ up to the current iteration, scheduling constraint r_k, estimation error function $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})) : \mathcal{S}_k \subseteq [m], k \in [K] \mapsto \mathbb{R}$

Output: Sensor set $\mathcal{S}_k$ that approximates the solution to Problem 1 at t_k

$\mathcal{S}^0 \leftarrow \emptyset$, $\mathcal{X}^0 \leftarrow [m]$, and $t \leftarrow 1$

Iteration t:

1. If $\mathcal{X}^{t-1} = \emptyset$, **return** $\mathcal{S}^{t-1}$
2. Select $i(t) \in \mathcal{X}^{t-1}$ for which $\rho_{i(t)}(\mathcal{S}^{t-1}) = \max_{i \in \mathcal{X}^{t-1}} \rho_i(\mathcal{S}^{t-1})$, with ties settled arbitrarily, where:

$$\rho_i(\mathcal{S}^{t-1}) \equiv \log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:k-1}, \mathcal{S}^{t-1})) - \log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:k-1}, \mathcal{S}^{t-1} \cup \{i\}))$$

   and $\mathcal{S}_{1:k-1} \equiv (\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{k-1})$

3.a. If $|\mathcal{S}^{t-1} \cup \{i(t)\}| > r_k$, $\mathcal{X}^{t-1} \leftarrow \mathcal{X}^{t-1} \setminus \{i(t)\}$, and go to Step 1

3.b. If $|\mathcal{S}^{t-1} \cup \{i(t)\}| \leq r_k$, $\mathcal{S}^t \leftarrow \mathcal{S}^{t-1} \cup \{i(t)\}$ and $\mathcal{X}^t \leftarrow \mathcal{X}^{t-1} \setminus \{i(t)\}$

4. $t \leftarrow t+1$ and continue

rithm 9 achieves a time complexity that is only linear in K with respect to the total number of evaluations of $\log\det(\Sigma(\hat{x}_{1:K}))$. *The reason is that* $\sum_{k=1}^{K} r_k^2 \leq \max_{k\in[K]}(r_k^2)K$. *In addition, for $w(t)$ zero mean and white Gaussian —as commonly assumed in the literature of sensor scheduling— the time complexity of each such evaluation is at most linear in K: the reason is that this $w(t)$ agrees with Assumption 4, in which case we prove that the time complexity of each evaluation of* $\log\det(\Sigma(\hat{x}_{1:K}))$ *is $O(n^{2.4}K)$ (linear in K).*[5]

Remark 7. (Sparsity of $\Sigma(\hat{x}_{1:K})$) *We state the three properties of* $\log\det(\Sigma(\hat{x}_{1:K}))$ *we prove to obtain the time complexity for Algorithm 9. The first two properties were mentioned in Remark 5: the monotonicity and supermodularity of* $\log\det(\Sigma(\hat{x}_{1:K}))$. *These two properties are responsible for that Algorithm 9 requires at most r_k^2 evaluations at each t_k. The third property, which follows, is responsible for the low time complexity for each evaluation of* $\log\det(\Sigma(\hat{x}_{1:K}))$*:*

- $\Sigma(\hat{x}_{1:K})$ *is the sum of two $nK \times nK$ sparse matrices: the first matrix is block diagonal, and the second one is block tri-diagonal. As a result, given that both of these matrices are known, each evaluation of* $\log\det(\Sigma(\hat{x}_{1:K}))$ *has time complexity $O(n^{2.4}K)$, linear in K (using the results in [142] —cf. Theorem 2 therein).*

We show in Appendix 5.5.2 that after we include at each evaluation step of $\log\det(\Sigma(\hat{x}_{1:K}))$ *the complexity to compute the two sparse matrices in $\Sigma(\hat{x}_{1:K})$, the total time complexity of Algorithm 9 is as given in Theorem 7.*

Our final remark on Theorem 7 follows:

Remark 8. (Comparison of Algorithm 9's time complexity to that of existing scheduling

[5] We can also speed up Algorithm 9 by implementing in Algorithm 10 the method of lazy evaluations [99]: this method avoids in Step 2 of Algorithm 10 the computation of $\rho_i(\mathcal{S}^{t-1})$ for unnecessary choices of i.

algorithms) *We do the comparison for two cases: batch state estimation, and Kalman filtering. In particular, we show that the time complexity of our algorithm is lower than that of existing sensor scheduling algorithms for batch state estimation, and of the similar order, or lower, of existing algorithms for Kalman filtering.*

Comparison with algorithms for batch state estimation. *In [135], Problem 1 is considered, and a semi-definite programming (SDP) algorithm is proposed; its time complexity is of the order* $O(\max_{k\in[K]}(r_k)K(nK)^{3.5} + (\max_{k\in[K]}(r_k^2)K^2(nK)^{2.5})$ *[143]. Clearly, this time complexity is higher than that of Algorithm 9, whose complexity is* $O(\max_{k\in[K]}(r_k)^2K^2n^{2.4})$. *In addition, the algorithm presented in [135] provides no worst-case approximation guarantees* (5.6)*, in contrast to Algorithm 9 that provides* (5.6).

Comparison with algorithms for Kalman filtering. *We do the comparison in two steps: first, we consider algorithms based on the maximization of submodular functions, and second, algorithms based on convex relaxation techniques or the alternating direction method of multipliers (ADMM):*

- Algorithms based on the maximization of submodular functions: *In [5], an algorithm is provided that is valid for a restricted class of linear systems: its time complexity is* $O(\max_{k\in[K]}(r_k)mn^2K + n^{2.4}K)$. *This time complexity is of similar order to that of Algorithm 9, whose complexity is of the order* $O(\max_{k\in[K]}(r_k)^2Kn^{2.4}K)$, *since* $\max_{k\in[K]}(r_k) < m$. *Specifically, we observe in Algorithm 9's time complexity the additional multiplicative factor* K *(linear in* K*); this difference emanates from that Algorithm 9 offers a near-optimal guarantee over the whole time horizon* $(t_1, t_2, \ldots, t_K)$ *whereas the algorithm in [5] offers a near-optimal guarantee only for the last time step* t_K. *In addition, Algorithm 9 holds for any linear continuous time-invariant system (no restrictions are necessary), in contrast to the algorithm in [5], and it holds for any discrete time-variant systems where* A_k *in* (5.8) *is full rank; the latter assumption is one of the four restrictive conditions in [5] (Theorem 13).*

- Algorithms based on convex relaxation techniques or ADMM: *In [119], the authors assume a single sensor (*$r_k = 1$ *across* t_k*), and their objective is to achieve a minimal estimation error by minimizing the number of times this sensor will be used over the horizon* $t_1, t_2, \ldots, t_K$. *The time complexity of the proposed algorithm is* $O(n^{2.5}K^2 + n^{3.5}K)$. *This time complexity is higher than that of Algorithm 9, whose complexity for* $r_k = 1$ *is of the order* $O(n^{2.4}K^2)$. *In [132], the authors employ ADMM techniques to solve a periodic sensor scheduling problem. They consider a zero mean and white Gaussian* $w(t)$. *The time complexity of the proposed algorithm is* $O((nK)^3 + (\max_{k\in[K]}(r_k)K)n^2K^2 + max(r_k)^2nK^3)$. *This time complexity is of similar order to that of Algorithm 9, whose complexity in this case is* $O(\max_{k\in[K]}(r_k^2)n^{2.4}K^2)$, *since* $\max_{k\in[K]}(r_k) \leq K$; *in particular, for* $K > n^{0.4}\max_{k\in[K]}(r_k)$, *Algorithm 9 has lower time complexity.*[6]

[6] More algorithms exist in the literature, that also use convex relaxation [144] or randomization techniques [20], and have similar time complexity to Algorithm 9. They achieve this complexity using additional approximation methods: e.g., they optimize instead an upper bound to the involved estimation error metric.

With the above remarks we conclude: Algorithm 9 enjoys both the estimation accuracy of the batch state scheduling algorithms and the low time complexity of the Kalman filtering scheduling algorithms, since:

- Algorithm 9 offers a near-optimal worst-case approximation guarantee for the batch state estimation error. This estimation error is only approximated by the Kalman filtering sensor scheduling algorithms: the reason is that they aim instead to minimize the sum of each of the estimation errors for $x(t_k)$ (across t_k). However, this sum only upper bounds the batch state estimation error.

- Algorithm 9 has time complexity lower than the state of the art batch estimation algorithms, and at the same time, lower than, or similar to, the time complexity of the corresponding Kalman filtering algorithms.

In addition: Algorithm 9's approximation guarantee holds for any linear system (continuous or discrete time). Moreover, Algorithm 9's time complexity guarantee holds for any continuous time system, and for discrete time systems where A_k in (5.8) is full rank across k.

The proof of Theorem 7 can be found in Appendix 5.5.2.

5.3.3. Limits on Sensor Scheduling for Minimum Variance Batch State Estimation

In this section, we derive two trade-offs between three important parameters of our sensor scheduling problem:[7]

- the number of measurements times $(t_1, t_2, \ldots, t_K)$

- the number r_k of sensors that can be used at each t_k

- the value of the estimation error $\mathbb{E}(\|x_{1:K} - \hat{x}_{1:K}\|_2^2)$.

The first of the two trade-offs is captured in the next theorem:

Theorem 8. *Let* $\sigma_w^{(-1)} \equiv \max_{i\in[nK]}[\mathbb{C}(x_{1:K})^{-1}]_{ii}$ *and* $\sigma_v^{(-1)} \equiv \|\mathbb{C}(v_{1:K})^{-1}\|_2$. *Also, let* $C_{1:K}$ *be the block diagonal matrix where each of its* K *diagonal elements is equal to* C, *where* C *is the matrix* $[C_1^\top, C_2^\top, \ldots, C_m^\top]^\top$. *For the variance of the error of the minimum variance estimator* $\hat{x}_{1:K}$*:*

$$\mathbb{E}(\|x_{1:K} - \hat{x}_{1:K}\|_2^2) \geq \frac{n}{\sigma_v^{(-1)} \max_{k\in[K]}(r_k)\|C_{1:K}\|_2^2 + \sigma_w^{(-1)}/K}. \tag{5.9}$$

The lower bound in (5.9) decreases as the number of used sensors for scheduling r_k increases or the number measurement times K increases, and increases as the system's size increases. Since these qualitative relationships were expected, the importance of this theorem lies on the quantification of these relationships (that also includes the dependence on the noise

[7] We recall from Section 5.2 that the objective of Problem 1 is related to $\mathbb{E}(\|x_{1:K} - \hat{x}_{1:K}\|_2^2)$ in that when $\log\det(\Sigma(\hat{x}_{1:K}))$ is minimized the probability that the estimation error $\|x_{1:K} - \hat{x}_{1:K}\|_2^2$ is small is maximized.

parameters $\sigma_w^{(-1)}$ and $\sigma_v^{(-1)}$): for example, (5.9) decreases only inversely proportional with the number of sensors for scheduling; that is, increasing the number r_k so to reduce the variance of the error of the minimum variance estimator is ineffective, a fundamental limit. In addition, this bound increases linearly with the system's size; this is another limit for large-scale systems.

Similar results are proved in [145] for the steady state error covariance of scalar systems in the case that the number of sensors goes to infinity. In more detail, the authors in [145] account for different types of multi-access schemes, as well as, for fading channels between the sensors and the fusion centre that combines the sensor measurements.

The next corollary presents our last trade-off:
Corollary 6. *Consider that the desired value for* $\mathbb{E}(\|x_{1:K}-\hat{x}_{1:K}\|_2^2)$ *is* α. *Any set of scheduled sensors at* $t_1, t_2, \ldots, t_K$ *that achieves this error satisfies:*

$$\max_{k\in[K]}(r_k) \geq \frac{n/\alpha - \sigma_w^{(-1)}/K}{\sigma_v^{(-1)}\|C_{1:K}\|_2^2}. \tag{5.10}$$

Eq. (5.10) implies that the number of sensors used for scheduling at each t_k increases as the error of the minimum variance estimator or the number of measurements times K decreases. More importantly, it quantifies that this number increases linearly with the system's size for fixed error variance. This is again a fundamental limit, meaningful for large-scale systems.

5.4. Concluding Remarks & Future Work

We work on extending the results of this chapter to largely unknown systems, under the presence of non-linear measurements. The first of these extensions allows systems whose evolution is captured by, e.g., Gaussian processes or random networks (the former example is a widely used assumption for motion models; cf. [133] and references therein). The second of these extensions allows complex measurement environments, such as camera-sensor environments, that can enable the application of our results in domains such as robotics and the automotive sector.

5.5. Appendix: Proof of Results

5.5.1. Closed formula for the error covariance of $\hat{x}_{1:K}$

We compute the error covariance of $\hat{x}_{1:K}$: Denote as $S_{1:K}$ the block diagonal matrix with diagonal elements the sensor selection matrices $S(t_1), S(t_2), \ldots, S(t_K)$. Moreover, denote as C the matrix $[C_1^\top, C_2^\top, \ldots, C_m^\top]^\top$. Finally, denote $y_{1:K} \equiv (y(t_1)^\top, y(t_2)^\top, \ldots, y(t_k)^\top)^\top$, $w_{1:K} \equiv (w(t_1)^\top, w(t_2)^\top, \ldots, w(t_k)^\top)^\top$, and $v_{1:K} \equiv (v(t_1)^\top, v(t_2)^\top, \ldots, v(t_k)^\top)^\top$, where $v(t_k) \equiv (v_1(t_k)^\top, v_2(t_k)^\top, \ldots, v_m(t_k)^\top)^\top$. Then, from (5.1), (5.2) and (5.3):

$$y_{1:K} \;=\; O_{1:K}x_{1:K} + S_{1:K}v_{1:K}, \tag{5.11}$$

where $O_{1:K}$ is the $\sum_{k=1}^{K} r_k \times nK$ block diagonal matrix with diagonal elements the matrices $S(t_1)C, S(t_2)C, \ldots, S(t_K)C$. $\hat{x}_{1:K}$ has the error covariance $\Sigma(\hat{x}_{1:K}) = \mathbb{E}((x_{1:K} - \hat{x}_{1:K})(x_{1:K} - \hat{x}_{1:K})^\top)$ [103]:

$$\Sigma(\hat{x}_{1:K}) = \mathbb{C}(x_{1:K}) - \mathbb{C}(x_{1:K}) O_{1:K}^\top \Xi O_{1:K} \mathbb{C}(x_{1:K}), \tag{5.12}$$

where $\Xi \equiv (O_{1:K}\mathbb{C}(x_{1:K})O_{1:K}^\top + S_{1:K}\mathbb{C}(v_{1:K})S_{1:K}^\top)^{-1}$.

We simplify (5.12) in the following lemma:

Lemma 1. *The error covariance of $\hat{x}_{1:K}$ has the equivalent form:*

$$\Sigma(\hat{x}_{1:K}) = \left(\sum_{k=1}^{K} \sum_{i=1}^{m} s_i(t_k) U^{(ki)} + \mathbb{C}(x_{1:K})^{-1} \right)^{-1}, \tag{5.13}$$

where $s_i(t_k)$ is a zero-one function, equal to 1 if and only if sensor i is used at t_k, and $U^{(ki)}$ is the block diagonal matrix $C_{1:K}^\top I^{(ki)} \mathbb{C}(v_{1:K})^{-1} I^{(ki)} C_{1:K}$; $C_{1:K}$ is the block diagonal matrix where each of its K diagonal elements is equal to C, and $I^{(ki)}$ is the block diagonal matrix with mK diagonal elements such that: the $((k-1)m+i)$-th element is the $d_i \times d_i$ identity matrix I, and the rest of the elements are equal to zero.

5.5.2. Proof of Theorem 7

We prove Theorem 7 in three steps: we first show that $\log\det(\Sigma(\hat{x}_{1:K}))$ is a non-increasing function in the choice of the sensors; we then show that $\log\det(\Sigma(\hat{x}_{1:K}))$ is a supermodular function in the choice of the sensors; finally, we prove Theorem 7 by combining the aforementioned two results and results on the maximization of submodular functions over matroid constraints [12].

Notation. We recall that any collection $(x_1, x_2, \ldots, x_k)$ is denoted as $x_{1:k}$ $(k \in \mathbb{N})$.

Monotonicity in Sensor Scheduling for Minimum Variance Batch State Estimation

We first provide two notations, and then the definition of non-increasing and non-decreasing set functions. Afterwards, we present the main result of this subsection.

Notation. Given K disjoint finite sets $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_K$ and $A_i, B_i \in \mathcal{X}_i$, we write $A_{1:K} \preceq B_{1:K}$ to denote that for all $i \in [K]$, $A_i \subseteq B_i$ (A_i is a subset of B_i). Moreover, we denote that $A_i \in \mathcal{X}_i$ for all $i \in [K]$ as $A_{1:K} \in \mathcal{X}_{1:K}$.

Definition 17. *Consider K disjoint finite sets $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_K$. A function $h : \mathcal{X}_{1:K} \mapsto \mathbb{R}$ is* non-decreasing *if and only if for all $A, B \in \mathcal{X}_{1:K}$ such that $A \preceq B$, $h(A) \leq h(B)$; $h : \mathcal{X}_{1:K} \mapsto \mathbb{R}$ is* non-increasing *if $-h$ is non-decreasing.*

The main result of this subsection follows:

Proposition 4. *For any finite $K \in \mathbb{N}$, consider K distinct copies of $[m]$, denoted as $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_K$. The estimation error metric $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})) : \mathcal{M}_{1:K} \mapsto \mathbb{R}$ is a non-increasing function in the choice of the sensors $\mathcal{S}_{1:K}$.*

We next show that $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K}))$ is a supermodular function with respect to the selected sensors $\mathcal{S}_{1:K}$.

Submodularity in Sensor Scheduling for Minimum Variance Batch State Estimation

We first provide a notation, and then the definition of submodular and supermodular set functions. Afterwards, we present the main result of this subsection.

Notation. Given K disjoint finite sets $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_K$ and $A_{1:K}, B_{1:K} \in \mathcal{X}_{1:K}$, we write $A_{1:K} \uplus B_{1:K}$ to denote that for all $i \in [K]$, $A_i \cup B_i$ (A_i union B_i).
Definition 18. *Consider K disjoint finite sets $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_K$. A function $h : \mathcal{X}_{1:K} \mapsto \mathbb{R}$ is* submodular *if and only if for all $A, B, C \in \mathcal{X}_{1:K}$ such that $A \preceq B$, $h(A \uplus C) - h(A) \geq h(B \uplus C) - h(B)$; $h : \mathcal{X}_{1:K} \mapsto \mathbb{R}$ is* supermodular *if $-h$ is submodular.*

According to Definition 18, set submodularity is a diminishing returns property: a function $h : \mathcal{X}_{1:K} \mapsto \mathbb{R}$ is set submodular if and only if for all $C \in \mathcal{X}_{1:K}$, the function $h_C : \mathcal{X}_{1:K} \mapsto \mathbb{R}$ defined for all $A \in \mathcal{X}_{1:K}$ as $h_C(A) \equiv h(A \uplus C) - h(A)$ is non-increasing.

The main result of this subsection follows:
Proposition 5. *For any finite $K \in \mathbb{N}$, consider K distinct copies of $[m]$, denoted as $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_K$; the estimation error metric $\log\det(\Sigma(\hat{x}_{1:K}|\mathcal{S}_{1:K})) : \mathcal{M}_{1:K} \mapsto \mathbb{R}$ is a set supermodular function in the choice of the sensors $\mathcal{S}_{1:K}$.*

Proposition 5 implies that as we increase at each t_k the number of sensors used, the marginal improvement we get on the estimation error of $x_{1:K}$ diminishes.

We are now ready for the proof of Theorem 7.

Proof of Theorem 7

We first provide the definition of a matroid, and then continue with the main proof:
Definition 19. *Consider a finite set $\mathcal{X}$ and a collection $\mathcal{C}$ of subsets of $\mathcal{X}$. $(\mathcal{X}, \mathcal{C})$ is:*

- *an* independent system *if and only if:*
  - *$\emptyset \in \mathcal{C}$, where $\emptyset$ denotes the empty set*
  - *for all $X' \subseteq X \subseteq \mathcal{X}$, if $X \in \mathcal{C}$, $X' \in \mathcal{C}$.*
- *a* matroid *if and only if in addition to the previous two properties:*
  - *for all $X', X \in \mathcal{C}$ where $|X'| < |X|$, there exists $x \notin X'$ and $x \in X$ such that $X' \cup \{x\} \in \mathcal{C}$.*

Proof:[of Part 1 of Theorem 7] We use the next result from the literature of maximization of submodular functions over matroid constraints:

Lemma 2 (Ref. [12]). *Consider K independence systems $\{(\mathcal{X}_k, \mathcal{C}_k)\}_{k\in[K]}$, each the intersection of at most P matroids, and a submodular and non-decreasing function $h : \mathcal{X}_{1:K} \mapsto \mathbb{R}$. There exist a polynomial time greedy algorithm that returns an (approximate) solution $\mathcal{S}_{1:K}$ to:*

$$\begin{array}{ll} \underset{\mathcal{S}_{1:K}\preceq\mathcal{X}_{1:K}}{maximize} & h(\mathcal{S}_{1:K}) \\ subject\ to & \mathcal{S}_k \cap \mathcal{X}_k \in \mathcal{C}_k, k \in [K], \end{array} \tag{5.14}$$

that satisfies:

$$\frac{h(\mathcal{O}) - h(\mathcal{S}_{1:K})}{h(\mathcal{O}) - h(\emptyset)} \leq \frac{P}{1+P}, \tag{5.15}$$

where $\mathcal{O}$ is an (optimal) solution to (5.14).

In particular, we prove:
Lemma 3. *Problem 1 is an instance of* (5.14) *with* $P = 1$.

This observation, along with Lemmas 2 and 3 complete the proof of (5.6), since the adaptation to Problem 1 of the greedy algorithm in [12] (Theorem 4.1) results to Algorithm 9.

■

Proof of Part 2 of Theorem 7: In Lemma 1 in Appendix 5.5.1 we prove that $\Sigma(\hat{x}_{1:K})$ is the sum of two matrices: the first matrix is a block diagonal matrix, and the second one is the inverse of the covariance of $x_{1:K}$, $\mathbb{C}(x_{1:K})$. The block diagonal matrix is computed in $O(n^{2.4}K)$ time. Moreover, by extending the result in [133] (Theorem 1), we get that $\mathbb{C}(x_{1:K})^{-1}$ is a block tri-diagonal matrix, that is described by the $(K-1)$ transition matrices $\Phi(t_{k+1}, t_k)$ [85], where $k \in [K-1]$, and K identity matrices. For continuous time systems, the time complexity to compute all the block elements in $\mathbb{C}(x_{1:K})^{-1}$ is $O(n^3K)$ [146]; for discrete time systems, it is $O(n^{2.4}K)$ [85]. This computation of $\mathbb{C}(x_{1:K})^{-1}$ is made only once. Finally, from Theorem 2 in [142], we can now compute the $\det(\Sigma(\hat{x}_{1:K}))$ in $O(n^{2.4}K)$ time, since $\Sigma(\hat{x}_{1:K})$ is block tri-diagonal. Therefore, the overall time complexity of Algorithm 9 is: $O(n^3K) + O(2n^{2.4}K\sum_{k=1}^{K} r_k^2) = O(n^{2.4}K\sum_{k=1}^{K} r_k^2)$ for K large, since $\mathbb{C}(x_{1:K})^{-1}$ is computed only once, and Algorithm 9 requests at most $\sum_{k=1}^{K} r_k^2$ evaluations of $\Sigma(\hat{x}_{1:K})$.

The proof is complete. ■

CHAPTER 6 : Selecting sensors in biological fractional-order systems

In this chapter, we focus on sensor selection, i.e., determine the minimum number of state variables that need to be measured, to monitor the evolution of the entire biological system, i.e., all the state variables, when modeled by discrete-time fractional-order systems (DTFOS) that are subject to modeling errors, process and measurement noise. These systems are particularly relevant when modeling of spatiotemporal dynamics of processes in which the impact of long-range memory cannot be properly modeled by multivariate auto-regressive integrative moving-average models. Therefore, DTFOS enable a unified state-space framework to model the evolution of several biological (e.g., stem cell growth and bacteria evolution) and physiological signals (e.g., electroencephalogram and electromyogram).

Therefore, in this chapter, we focus on the solution to four different (yet related) problems of sensor selection for DTFOS, that are motivated by constraints on the data acquisition that are enforced by the detrimental impact of the sensing mechanisms to the biological system, the cost of performing the measurements with the current sensing technology, or spatial constraints that limit the number of sensors that can be deployed. Towards determining the solution to these problems that we show to be NP-hard, we leverage the representation of the DTFOS to derive new objectives and conditions that, ultimately, enable us to efficiently approximate a solution to the different problems by exploiting the submodularity structure, which enables us to establish sub-optimality guarantees.[1]

6.1. Introduction

A multitude of complex systems exhibits long-range (non-local) properties, interactions and/or dependencies (e.g., power-law decays in the weights of linear combination of past data) used to describe the biological system evolution. Example of such systems includes Hamiltonian systems, where the memory (i.e., dependence on the past data) is the result of stickiness of trajectories in time to the islands of regular motion [147]. Alternatively, it has been rigorously confirmed that viscoelastic properties are typical for a wide variety of biological entities like stem cells, liver, pancreas, heart valve, brain, muscles [147, 148, 149, 149, 150, 151, 152, 153, 154, 155], suggesting that the long-range memory of these systems obey the power law distributions. These dynamical systems can be characterized by the well-established mathematical theory of fractional calculus [156], and the corresponding systems could be described by fractional differential equations [157, 158, 159, 160, 161]. However, it is until recently that fractional order system (FOS) starts to find its strong position in a wide spectrum of applications in different domains due to the availability of computing and data acquisition methods to evaluate its efficacy in terms of capturing the underlying system states evolution.

Specifically, in [157], by the adoption of non-Gaussian statistical approaches, the authors identify the co-existence of fast and slow dividing subpopulations, and quiescent cells, in stem cells from three species. The mathematical analysis also shows that, instead of developing independently, stem cells exhibit a time-dependent fractal behavior as they interact with each other through molecular and tactile signals. In [158], the existence of a statistical

[1] This chapter is based on the paper by Tzoumas et al. [21].

fractal behavior and inadequacy of modeling blood glucose dynamics via linear state space models, is proved by the multi-fractal spectrum computed from the blood glucose time series of four individuals. A fractional order system model is alternatively proposed and evaluated to have superior regarding predictive power and controller synthesis. In [159], a multi-dimensional FOS is considered to capture the muscular dynamics in the process of forearm movement. The motivation comes from the power-law correlation decay as opposed to the exponential law, which is fundamentally assumed by the popular autoregressive moving average model. After the retrieval of the FOS model from the observations, it is shown that the model output is superior to ARMA to capture the observed spatial and temporal long-range dependence. In [160], a more comprehensive set of physiological processes (i.e., neural, muscular and vascular processes) are considered to study the minimal sensor placement problem in the context of the multi-dimensional FOS. The experimental results suggest that the adoption of FOS and the control theory developed based on it can help improve the design of efficient and reliable cyber-physical systems in the biomedical domain. In [161], the authors propose a statistical non-extensive causal inference framework and construct the generalized master equation (GME) to characterize the dynamics of complex systems that exhibit power-law spatiotemporal dependencies. The solution of the GME suggests a FOS be considered to capture the dynamical behaviors of the systems. In addition to the application of fractional order calculus to differentiable dynamical systems, very recent efforts have also been very successful to extend local fractional calculus to non-differentiable, irregular sets like fractals or fractal networks [162, 163, 164, 165]. The fractality/multifractality of network, their characterization, computation, their influence on the dynamics of complex networked systems is attracting greater attention from a multi-disciplinary perspective. The possibility to extend the fractional to self-similar non-smooth objects is opening new frontiers in science. Non-linear analysis of data offers still unsolved analytical problems related not only to complex physics and abstract mathematical theories including fractals and fractional calculus [166].

Subsequently, because the current sensing technology is mainly digital, we focus on *discrete-time fractional-order systems* (DTFOS) [167], whose parameterization consists of a relatively small number of parameters, and the dynamics subject to modeling errors and external disturbances. Furthermore, in addition to modeling errors and external disturbances in the DTFOS dynamics, we also account for external disturbances in the sensing technology since our motivating technology, i.e., the EEG, uses sensing technology where noise commonly corrupts the collected data. Subsequently, in this chapter, we propose to explore and exploit the trade-offs between the selected sensors and the capability to assess the process state over time, which we refer to as estimation performance since the state is obtained up to a confidence level subject to disturbance and noise. In other words, the combined effect of the modeling errors and external disturbances in the spatiotemporal dynamical processes requires the proper deployment of sensing technology that guarantees the best estimation performance (that is, the least estimation error) given the modeling errors' and external disturbances' characteristics.

In the last years, we have witnessed a growing interest on the trade-off between the number of used sensors and the degree of observability of linear time-invariant (LTI) systems [7, 52, 53, 84, 105, 110, 111, 168], which are a particular case of DTFOS. In particular, this

trade-off has been explored under the assumption that either the exact LTI system model is available, in which case one needs to ensure observability [7, 52, 53, 84, 105, 110, 111, 168], or only the structure of the LTI system model is available, in which case one needs to ensure structural observability [109] or strong structural observability [169]. More recently, this interest as extended to deal with DTFOS, either when the models are exact [170], or in the context of structural observability [171]. Although ensuring observability is key towards the implementation of stable estimators, it does not explicitly explore the trade-offs between the chosen sensors and the quality of the state estimate and the model uncertainty, which is of utmost importance in biological settings, e.g., in EEG applications. These trade-off has been studied so far only for LTI systems, as we briefly review next. In [139, 172, 173], the authors explore the trade-offs for LTI systems in the context of Kalman estimation. Specifically, in [172], the authors quantify the trade-off between the number of sensors and the output measurements to achieve a specified value for the minimum mean square error (MMSE) of the Kalman estimator, whereas in [139] the authors consider to place small numbers of sensors to optimize the resultant MMSE, and in [173] the author designs an output matrix with a desired norm that minimizes the MMSE.

In this chapter, we extend the current literature to address the trade-off between the chosen sensors and the quality of the state estimate for the case of DTFOS with known parametric model and under possible uncertainties in the dynamics, as well as, noise in the measurements collected by the sensing technology. Specifically, we address the following problems: (*i*) determine the minimum number of sensors to ensure bounded process disturbance error within a prescribed threshold; (*ii*) determine the placement of a specified number of sensors to minimize the process disturbance error; (*iii*) determine the minimum number of sensors to ensure bounded state estimation error within a prescribed threshold; and (*iv*) determine the placement of a specified number of sensors to minimize the state estimation error. It is worth noticing that among these four problems, the first couple of problems enforces the validity of the model by quantifying the uncertainty of the system's evolution, whereas the remaining two aim to determine the most likely state of the process across a time-window.

The main contributions of this chapter can be cast in the following three domains:

Translational – it equips scientists (e.g., biologists and neuroscientists) and engineers alike with a unified framework to decide upon the sensor measurements to be considered to perform state estimation, i.e., to perform *sensor selection* to quantify uncertainty in the state and unknown disturbances and noises, in the context of fractional-order state-space representations capable of modeling spatiotemporal dynamics of processes in which the impact of long-range memory cannot be properly modeled by multivariate auto-regressive integrative moving-average models.

Theoretical – we propose to derive observability conditions that enable the quantification of the uncertainty of biological processes modeled by the proposed state-space representation, as well as identify the state variables that play a key role in monitoring the evolution of the dynamics while making the trade-off with the accuracy of the estimation. Specifically, we propose computationally efficient algorithms to provide sub-optimal solutions to the minimum number of variables that need to be measured (that is NP-hard), while establishing

guarantees on the optimality gap.

Application – recently there is a renewed interest in neuro-wearable devices largely boosted by initiatives sponsored by either the Facebook that aims to use wearable devices to write 100 words per minute, and the NeuraLink by Elon Musk that aims to develop implantable brain-computer interfaces. Subsequently, we propose to revisit the neuro-wearables that rely on electroencephalogram, and determine the sensor location that seems to be the most effective with respect to a pre-specified number of sensors. In particular, we argue that for a variety of tasks the location of sensors currently used in such wearable devices is sub-optimal with respect to the proposed objectives that aim to ensure the quality of estimated state, process and measurement noise. Consequently, we conclude that at the light of this framework, some of the neuro-wearables should be re-designed to enhance dynamic systems properties such as observability.

In summary, our main contributions are as follows: (*i*) we formalize the sensor placement problems in context of four different (yet related) problems pertaining to sensor placement to minimize the process disturbance error and state estimation error; (*ii*) we show that these problems are NP-hard; (*iii*) we present approximation schemes for their solution that have provably optimal approximation performance; and (*iv*) we illustrate the proposed approaches using EEG signal data associated with a variety of tasks.

The remainder of this chapter is organized as follows. In Section 6.2, we provide our setup and problem formulation. In Section 6.3, we present our main results. In Section 6.4, we illustrate how the main results can be applied in the context of real EEG signal data. Section 6.5 concludes the chapter.

6.2. Problem Statement

In this section, we introduce the problems addressed in the present chapter. First, we introduce the DTFOS model used in Section 6.2.1, while revisiting some of its properties, and the best linear estimator for it in Section 6.2.2. Then, in Section 6.2.3, we introduce the optimal sensor placement problem for DTFOS, which seeks to determine the minimum collection of sensors that ensure a pre-specified estimation performance, or the configuration of a given number of sensors that attain the best process disturbance and estimate quality.

6.2.1. DTFOS Model

We consider the linear DTFOS described by

$$\begin{aligned} \Delta x_{k+1} &= Ax_k + w_k, \\ y_k &= Cx_k + v_k, \quad k = 0, 1, \dots \end{aligned} \tag{6.1}$$

where $x_k = [x_k^1,\ x_k^2,\ \dots,\ x_k^n]^\top \in \mathbb{R}^n$ ($n \in \mathbb{N}$) is the state vector, $y_k \in \mathbb{R}^c$ the measured output vector, w_k the process disturbance and v_k the measurement noise, and x_0 the initial condition. Additionally, $\Delta \equiv \operatorname{diag}\left(\Delta_{k+1}^{\alpha_1}, \Delta_{k+1}^{\alpha_2}, \dots, \Delta_{k+1}^{\alpha_n}\right)$ is the diagonal matrix operator,

where $\Delta_{k+1}^{\alpha_i}$ is the discrete fractional-order difference operator such that

$$\Delta_{k+1}^{\alpha_i} x_{k+1}^i \equiv \sum_{j=0}^{k+1} (-1)^j \binom{\alpha_i}{j} x_{k-j+1}^i,$$

and $\binom{\alpha_i}{j} = \frac{\Gamma(\alpha_i+1)}{\Gamma(j+1)\Gamma(\alpha_i-j+1)}$, where $\alpha_i > 0$ is the *fractional-order exponent*, and $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$ is the Gamma function. In summary, the matrix A captures the spatial coupling (i.e., dependency) of the process, whereas α_i capture the temporal dependency of the process associated with x_i.

Also, we notice that it is possible to provide a closed-form solution to (6.1), following [174], and which can be described as follows.

Lemma 4. *For all $k \geq 1$, the solution to (6.1) is given by $x_k = G_k x_0 + \sum_{j=0}^{k-1} G_{k-1-j} w_j$, where*

$$G_k \equiv \begin{cases} I, & k = 0 \\ \sum_{j=0}^{k-1} A_j G_{k-1-j}, & k \geq 1, \end{cases}$$

where $A_0 = A$, and A_j is a diagonal matrix whose i-th entry is $(-1)^j \binom{\alpha_i}{j+1}$. ◇

In particular, Lemma 4 states that a linear DTFOS can be interpreted as a linear time-variant switching system, where transitions are known. Subsequently, we can develop a Kalman-like estimator for this process, which estimates' characterization is leveraged to study the trade-offs between performance of the estimator and a specified sensor placement.

6.2.2. Minimum Variance Linear Estimator

For any estimation horizon K (that is, k in (6.1) varies from 0 to K), we first present the minimum mean square linear estimator of $z_K \equiv (x_0^\top, w_0^\top, w_1^\top, \ldots, w_{K-1}^\top)^\top$. This estimator is particularly useful in biological systems to assess the validity of the model, since a quantification of uncertainty is obtained. To this end, we use the following common assumption.

Assumption 5. *Let the initial condition be unknown and modeled by a random variable whose expected value is $\bar{x}_0$ and its covariance is $\mathbb{C}(x_0) \succ 0$. In addition, let the process disturbance w_k and the measurement noise v_k to be described by zero-mean random variables, whose covariance is described respectively by $\mathbb{C}(w_k) \succ 0$ and $\mathbb{C}(v_k) \succ 0$, for all $k \geq 0$, where $\mathbb{C}(v_k)$ is a diagonal matrix; that is, the measurement noises between any two sensors that correspond to two rows of C are uncorrelated. Furthermore, for all $k, k' \geq 0$ with $k \neq k'$, let the x_0, w_k and v_k, as well as, the w_k, $w_{k'}$, v_k and $v_{k'}$ to be uncorrelated.*

◦

Moreover, we consider the following notations: let the vector of measurements $y_{0:K} \equiv (y_0^\top, y_1^\top, \ldots, y_K^\top)^\top$, the vector of process noises $w_{0:K-1} \equiv (w_0^\top, w_1^\top, \ldots, w_{K-1}^\top)^\top$ and the vector of measurement noises $v_{0:K} \equiv (v_0^\top, v_1^\top, \ldots, v_K^\top)^\top$. Notice that whereas the vector $y_{0:K}$ is known, the vectors $w_{0:K-1}$ and $v_{0:K}$ are not. Additionally, we refer to the interval $[0, K] \equiv \{0, 1, \ldots, K\}$ as the *estimation horizon* of (6.1), and its *length* is $K + 1$.

Next, given an estimation horizon $[0, K]$, to derive the minimum mean square linear estima-

tor of z_K, from (6.1) and Lemma 4, we have

$$y_{0:K} = \mathcal{O}_K z_K + v_{0:K}, \tag{6.2}$$

where $\mathcal{O}_K = [L_0^\top C^\top, L_1^\top C^\top, \ldots, L_K^\top C^\top]^\top$ with the $n \times n(K+1)$ matrix $L_i = [G_i, G_{i-1}, \ldots, G_0, \mathbf{0}]$, and $\mathbf{0}$ is the zero matrix with appropriate dimensions.

Thus, following similar steps to those performed for linear time-invariant systems [172], the minimum mean square linear estimate of z_K is given by

$$\begin{aligned}\hat{z}_K \equiv \mathbb{E}(z_K) + \mathbb{C}(z_K)\mathcal{O}_K^\top(\mathcal{O}_K\mathbb{C}(z_K)\mathcal{O}_K^\top + \mathbb{C}(v_{0:K}))^{-1} \\ (y_{0:K} - \mathcal{O}_K\mathbb{E}(z_K) - \mathbb{E}(v_{0:K})),\end{aligned}$$

where $\mathbb{E}(x)$ is the expected value of x, and $\mathbb{C}(x) \equiv \mathbb{E}([x - \mathbb{E}(x)][x - \mathbb{E}(x)]^\top)$ its covariance. Furthermore, the error covariance of $\hat{z}_K$ is given by

$$\begin{aligned}\Sigma_{\hat{z}_K} &\equiv \mathbb{E}((z_K - \hat{z}_K)(z_K - \hat{z}_K)^\top) \\ &= \mathbb{C}(z_K) - \mathbb{C}(z_K)\mathcal{O}_K^\top(\mathcal{O}_K\mathbb{C}(z_K)\mathcal{O}_K^\top + \mathbb{C}(v_{0:K}))^{-1} \\ &\qquad \mathcal{O}_K\mathbb{C}(z_K).\end{aligned} \tag{6.3}$$

In this chapter, we capture the estimation performance of $\hat{z}_K$ with the metric $\log\det(\Sigma_{\hat{z}_K})$, which is proportional to the conditional entropy of z_K given the measurements $y_{0:K}$, and as a result, captures how well z_K is explained by $y_{0:K}$ [175, Proposition 2]. In particular, the metric $\log\det(\Sigma_{\hat{z}_K})$ captures the probability that the estimation error $\|z_K - \hat{z}_K\|_2^2$ is small. To explain this, consider the η-confidence ellipsoid of $z_K - \hat{z}_K$ [119]: The η-confidence ellipsoid is the minimum volume ellipsoid that contains $z_K - \hat{z}_K$ with probability η. Specifically, it is encapsulated by $\mathcal{E}_\epsilon(\hat{z}_K) \equiv \{z : z^\top \Sigma_{\hat{z}_K} z \leq \epsilon\}$, where $\epsilon \equiv F^{-1}_{\chi^2_{n(K+1)}}(\eta)$ and $F_{\chi^2_{n(K+1)}}$ is the cumulative distribution function of a χ-squared random variable with $n(K+1)$ degrees of freedom [124]. Therefore, the volume of $\mathcal{E}_\epsilon(\hat{z}_K)$ that quantifies the estimation's error of $\hat{z}_K$ is given as follows:

$$\mathrm{vol}(\mathcal{E}_\epsilon(\hat{z}_K)) \equiv \frac{(\epsilon\pi)^{n(K+1)/2}}{\Gamma\left(n(K+1)/2+1\right)} \det(\Sigma_{\hat{z}_K}^{1/2}). \tag{6.4}$$

Henceforth, if we consider the logarithm of (6.4), we obtain

$$\log \mathrm{vol}(\mathcal{E}_\epsilon(\hat{z}_K)) = \beta + 1/2 \log\det(\Sigma_{\hat{z}_K}), \tag{6.5}$$

where β is a constant that depends only on $n(K+1)$ and ϵ, and, as a result, we refer to the $\log\det(\Sigma_{\hat{z}_K})$ as the $\log\det$ *initial state-uncertainty estimation error of the minimum variance linear estimator* of (6.1).

Alternatively, we might be interested in determine the minimum variance linear estimator of $x_{0:K} \equiv (x_0, x_1, \ldots, x_K)$, denoted by $\hat{x}_{0:K}$. To this end, the collection of measurements is given by $y_{0:K} = O_K x_{0:K} + v_{0:K}$, where O_K is the block diagonal matrix with diagonal elements $K+1$ copies of the matrix C. Subsequently, following similar steps to those in

[103], the state estimation $\hat{x}_{0:K}$ error covariance is given by

$$\Sigma_{\hat{x}_{0:K}} = \mathbb{C}(x_{0:K}) - \mathbb{C}(x_{0:K})O_K^\top(O_K\mathbb{C}(x_{0:K})O_K^\top + \mathbb{C}(v_{0:K}))^{-1}O_K\mathbb{C}(x_{0:K}). \tag{6.6}$$

Besides, by proceeding similarly to the reasoning above, we can define the log det *batch-state estimation error of the minimum variance linear estimator* of (6.1) as follows:

$$\log \mathrm{vol}(\mathcal{E}_\epsilon(\hat{x}_{0:K})) = \beta + 1/2 \log\det\left(\Sigma_{\hat{x}_{0:K}}\right). \tag{6.7}$$

6.2.3. Optimal Sensor Placement

Now, we introduce four different (yet, related) problems to assess the optimal sensor placement with respect to the log det of the initial state-uncertainty and batch-state estimation error of the minimum variance linear estimator of (6.1). Specifically, we propose for each to determine the placement of r sensors such that the overall estimation error is minimized, and determine a placement of sensors such that the estimation error satisfies a specified threshold.

Therefore, we propose to use the following sensor placement model: across the estimation horizon $[0, K]$, a *unique* subset of r sensors in (6.1) is placed and used, that corresponds to r of the c rows of C ($r \leq c$). In particular, for all $k \in [0, K]$ in (6.1),

$$y_k = SCx_k + v_k,\ k \in [0, K], \tag{6.8}$$

where S is the sensor placement matrix (constant across the estimation horizon $[0, K]$); that is, it is a zero-one matrix such that $S_{ij} = 1$ if sensor j is placed (which corresponds to the j-th row of C), and $S_{ij} = 0$ otherwise. We assume that a sensor can be placed at most once, and as a result, for each i there is one j such that $S_{ij} = 1$ while for each j there is at most one i such that $S_{ij} = 1$. Hence, given a sensor selection matrix S, the indices of the rows of C that correspond to used sensors is denoted by $\mathcal{S}$, i.e., $\mathcal{S} \equiv \{j : \text{ exists } i \text{ such that } S_{ij} = 1\}$.

Consequently, given the DTFOS in (6.1) and a finite estimation horizon $[0, K]$, we consider the following four problems:

Initial State-Uncertainty Estimation Error

(i) Provided a specified error threshold $R \in \mathbb{R}^+$, determine the *initial state-uncertainty minimal sensor placement problem* that is a solution to the following problem:

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{minimize}} \quad & |\mathcal{S}| \\ \text{subject to} \quad & \log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right) \leq R, \end{aligned} \tag{$\mathcal{P}_1$}$$

where $\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right)$ is the determinant of $\Sigma_{\hat{z}_K}$ in (6.3) when $\mathcal{O}_K$ is replaced by $\mathcal{O}_K(\mathcal{S})$, with explicit dependence on $\mathcal{S}$, and described by $\mathcal{O}_K(\mathcal{S}) = [L_0^\top C(\mathcal{S})^\top, L_1^\top C(\mathcal{S})^\top, \ldots, L_K^\top C(\mathcal{S})^\top]^\top$,

and $C(\mathcal{S})$ denotes the rows of C with indices in $\mathcal{S}$;

(*ii*) Provided a maximum number r of sensors to be placed, determine the *initial state-uncertainty cardinality-constrained sensor placement problem for minimum estimation error* that consists of a solution to the following problem:

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{minimize}} \quad & \log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right) \\ \text{subject to} \quad & |\mathcal{S}| \leq r. \end{aligned} \tag{$\mathcal{P}_2$}$$

Batch-State Estimation Error

(*iii*) Provided a specified error threshold $R \in \mathbb{R}^+$, determine the *minimal sensor placement problem* that is a solution to the following problem:

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{minimize}} \quad & |\mathcal{S}| \\ \text{subject to} \quad & \log\det\left(\Sigma_{\hat{x}_{0:K}}(\mathcal{S})\right) \leq R, \end{aligned} \tag{$\mathcal{P}_3$}$$

where $\det\left(\Sigma_{\hat{x}_{0:K}}(\mathcal{S})\right)$ is the determinant of $\Sigma_{\hat{x}_{0:K}}$ in (6.6) when O_K is replaced by $O_K(\mathcal{S})$, which is the block diagonal matrix with diagonal elements K copies of the matrix $C(\mathcal{S})$, and $C(\mathcal{S})$ denotes the rows of C with indices in $\mathcal{S}$; and

(*iv*) Provided a maximum number r of sensors to be placed, determine the *cardinality-constrained sensor placement problem for minimum estimation error* that consists of a solution to the following problem:

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{minimize}} \quad & \log\det\left(\Sigma_{\hat{x}_{0:K}}(\mathcal{S})\right) \\ \text{subject to} \quad & |\mathcal{S}| \leq r. \end{aligned} \tag{$\mathcal{P}_4$}$$

∘

Problems $(\mathcal{P}_1) - (\mathcal{P}_4)$ address different problems that focus on different practical considerations. Specifically, $(\mathcal{P}_1)$ aims to determine the minimum number of sensors to ensure bounded process disturbance error within a prescribed threshold, which enables the minimization of the estimation of the uncertainty that drives the system; thus, equipping us with an uncertainty quantification of the process evolution. In contrast, $(\mathcal{P}_2)$ addresses the problem of determining the placement of a specified number of sensors to minimize the process disturbance error, which captures the situations where one has a budget on the available sensing technology, and wants to deploy the sensors to maximize the performance of the process captured by minimizing the system's uncertainty.

Problem $(\mathcal{P}_3)$ focus on determining the minimum number of sensors to ensure bounded state estimation error within a prescribed threshold, which might be related with the satisfaction of some standard or accuracy required to have a sound estimate of the system's state. Finally, $(\mathcal{P}_4)$ targets the placement of a specified number of sensors to minimize the state estimation

error when the number of sensing mechanisms is limited and one aims to minimize the system's state estimate uncertainty.

Notwithstanding, as it will become clear in the upcoming sections, the underlying optimization structure is similar, which enables us to study them in a unified fashion. Specifically, to address these problems, we will show that both $\log\det(\Sigma_{\hat{z}_K})$ and $\log\det(\Sigma_{\hat{x}_{0:K}})$ are *supermodular* and *non-increasing* (formally defined in Section 6.3). As a consequence, approximation algorithms for these type of functions can be leveraged to provide approximate solutions to these problems with worst-case performance guarantees.

6.3. Sensor Placement for DTFOS

We present the main results of the present chapter. First, we show that $(\mathcal{P}_1)$-$(\mathcal{P}_4)$ are NP-hard (Theorem 9), which implies that optimal polynomial solutions to these problems are unlikely to exist. Next, we propose polynomial algorithms (Algorithm 11 and 12) to obtain an approximate solution to these problems, while ensuring worst-case performance guarantees (Theorem 11 and 12). In more detail, in Theorem 10, we show that the constraint and objective function in $(\mathcal{P}_1)/(\mathcal{P}_3)$ and $(\mathcal{P}_2)/(\mathcal{P}_4)$, respectively, are supermodular. Thereby, greedy algorithms can be provided to approximate the solution to these problems while ensuring a worst case scenario bounded optimality gap. Finally, in Theorem 13, we provide a discussion on the fundamental limits on the state-uncertainty estimation error and batch-state estimation, while exploring the trade-off with problems' parameters.

We start by showing the computational complexity of our problems in the next result.
Theorem 9. *The problems* $(\mathcal{P}_1)$-$(\mathcal{P}_4)$ *are NP-hard.* ◇

Subsequently, we need to devise a strategy that approximates the solutions to the proposed problems. Towards this goal, consider the following definitions.
Definition 20. *A function* $h : 2^{[c]} \mapsto \mathbb{R}$ *is* submodular*, where* $[c] = \{1, \ldots, c\}$*, if for any sets* $\mathcal{S}$ *and* $\mathcal{S}'$*, with* $\mathcal{S} \subseteq \mathcal{S}' \subseteq [c]$*, and any* $a \notin \mathcal{S}'$*,*

$$h(\mathcal{S} \cup \{a\}) - h(\mathcal{S}) \geq h(\mathcal{S}' \cup \{a\}) - h(\mathcal{S}').$$

A function $h : 2^{[c]} \mapsto \mathbb{R}$ *is* supermodular *if* $(-h)$ *is submodular.* ◇
Definition 21. *A function* $h : 2^{[c]} \mapsto \mathbb{R}$ *is a* non-increasing set function *if for any* $\mathcal{S} \subseteq \mathcal{S}' \subseteq [c]$ *it follows that* $h(\mathcal{S}) \geq h(\mathcal{S}')$*. Moreover,* h *is a* non-decreasing set function *if* $(-h)$ *is a non-increasing set function.* ◇

Furthermore, a function $h : 2^{[c]} \mapsto \mathbb{R}$ is submodular if, for any $a \in [c]$, the function $h_a : 2^{[c]\setminus\{a\}} \mapsto \mathbb{R}$ defined as $h_a(\mathcal{S}) \equiv h(\mathcal{S} \cup \{a\}) - h(\mathcal{S})$ is a non-increasing set function. This property is commonly referred to as the *diminishing returns property* [116].

Now, we show that the constraint and objective function of $(\mathcal{P}_1)/(\mathcal{P}_3)$ and $(\mathcal{P}_2)/(\mathcal{P}_4)$, respectively, are supermodular and non-increasing.
Theorem 10. *Let* c *be the number of rows of* C*, and* $s_i \in \{0, 1\}$ *be* 1 *if and only if* i*-th sensor (*i*-th row of* C*) is placed, and* $L_{0:K} \equiv [L_0^\top, L_1^\top, \ldots, L_K^\top]^\top$*. In addition, let* $M^{(i)} \equiv C_{0:K}^\top I^{(i)} \mathbb{C}(v_{0:K})^{-1} I^{(i)} C_{0:K}$*, where* $C_{0:K}$ *is the block diagonal matrix where each of its* $K+1$ *diagonal elements is equal to* C*, and* $I^{(i)}$ *is the diagonal matrix with* $c(K+1)$ *diagonal*

elements such that, for all $k \in [0, K]$, the $(kc + i)$-th element is 1, and the rest of the elements are equal to zero. Then, given any finite estimation horizon $[0, K]$, the following two equalities hold:

$$\log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right) = \\ -\log\det\left(\sum_{i=1}^{c} s_i L_{0:K}^{\top} M^{(i)} L_{0:K} + \mathbb{C}(z_K)^{-1}\right),$$

and

$$\log\det\left(\Sigma_{\hat{x}_{0:K}}(\mathcal{S})\right) = -\log\det\left(\sum_{i=1}^{c} s_i M^{(i)} + \mathbb{C}(x_{0:K})^{-1}\right).$$

Furthermore, both $\log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right)$ and $\log\det\left(\Sigma_{\hat{x}_{0:K}}(\mathcal{S})\right)$ are supermodular and non-increasing set functions with respect to the choice of the sensor set $\mathcal{S} \subset [c] = \{1, \ldots, c\}$. ◇

As consequence of Theorem 10, it follows that the functions exhibit the diminishing returns property, i.e., its rate of reduction with respect to newly placed sensors decreases as the cardinality of the already placed sensors increases. Therefore, some well known approximation schemes [13, 40] can be leveraged to obtain sub-optimal solutions to $(\mathcal{P}_1)$-$(\mathcal{P}_4)$ with optimality guarantees.

In Algorithm 11 and Algorithm 12, we present strategies to approximate the solutions to $(\mathcal{P}_1)/(\mathcal{P}_3)$ and $(\mathcal{P}_2)/(\mathcal{P}_4)$, respectively. Specifically, in Algorithm 11, we provide an efficient algorithm for $(\mathcal{P}_1)/(\mathcal{P}_3)$ that returns a sensor set that satisfies the prescribed threshold and has cardinality up to a multiplicative factor from the minimum cardinality sensor sets that meet the same estimation bound. More importantly, this multiplicative factor depends only logarithmically on the problems' parameters. These properties and the time complexity are described in the following result.

Algorithm 11 Approximation Algorithm for $(\mathcal{P}_1)/(\mathcal{P}_3)$

Input: $h_\alpha(\mathcal{S}) = \log\det\left(\Sigma_\alpha(\mathcal{S})\right)$, where $\alpha \in \{\hat{x}_{0:K}, \hat{z}_K\}$ for $k \in [0, K]$, and a threshold R on the total estimation error incurred by $h_\alpha(\mathcal{S})$.
Output: Approximate solution $\mathcal{S}_\alpha$ for $(\mathcal{P}_1)/(\mathcal{P}_3)$.

```
S_α ← ∅
while h_α(S_α) > R do
    a_i ← a' ∈ arg max_{a∈[c]\S_α} (h_α(S_α) − h_α(S_α ∪ {a}))
    S_α ← S_α ∪ {a_i}
end while
```

Theorem 11. *Let a solution to $(\mathcal{P}_1)/(\mathcal{P}_3)$ be denoted by $\mathcal{S}_\alpha^\star$, and the set obtained by Algorithm 11 be denoted by $\mathcal{S}_\alpha$. Moreover, denote the maximum diagonal element of $\mathbb{C}(x_0)$ and $\mathbb{C}(w_k)$, among all $k \in [0, K]$, as σ_0^2 and σ_w^2, respectively. Then,*

$$\log\det\left(\Sigma_\alpha(\mathcal{S}_\alpha)\right) \leq R, \tag{6.9}$$

and the optimality gap bounded as follows:

$$\frac{|\mathcal{S}_\alpha|}{|\mathcal{S}_\alpha^\star|} \leq 1 + \log\left\{\frac{\log\det\left(\Sigma_\alpha(\emptyset)\right) - \log\det\left(\Sigma_\alpha([c])\right)}{R - \log\det\left(\Sigma_\alpha([c])\right)}\right\} \equiv \eta \tag{6.10}$$

where $\log\det\left(\Sigma_{\hat{z}_K}(\emptyset)\right) \leq n(K+1)\log\max(\sigma_0^2, \sigma_w^2)$.

Furthermore, the time complexity of Algorithm 11 is $O(c^2(nK)^{2.4})$. ◇

Therefore, Algorithm 11 returns a sensor set that meets the estimation threshold of $(\mathcal{P}_1)/(\mathcal{P}_3)$. Moreover, the cardinality of this set is up to a multiplicative factor of η from the minimum cardinality sensor sets that meet the same estimation bound. In other words, η is a worst-case approximation guarantee for Algorithm 11. Besides, η depends only logarithmically on the problems' parameters. Additionally, the dependence of η on n, R and $\max(\sigma_0^2, \sigma_w^2)$ is expected from a design perspective. Specifically, by increasing the state space size n, requesting a better estimation guarantee by decreasing R, or incurring a noise of greater variance, should all push the cardinality of the selected sensor set upwards.

Next, in Algorithm 12, we provide an efficient algorithm for $(\mathcal{P}_2)/(\mathcal{P}_4)$ that returns a sensor set of cardinality r, where r is chosen by the designer. In the next result, we provide optimality guarantees of the solution obtained with Algorithm 12, as well as the computational complexity incurred by the algorithm.

Theorem 12. *Let a solution to* $(\mathcal{P}_2)/(\mathcal{P}_4)$ *be denoted by* $\mathcal{S}_\alpha^\star$*, and the set obtained by Algorithm 12 be denoted by* $\mathcal{S}_\alpha$*. Then,*

$$\frac{\log\det(\Sigma_\alpha(\mathcal{S}_\alpha)) - \log\det(\Sigma_\alpha(\emptyset))}{\log\det(\Sigma_\alpha(\mathcal{S}_\alpha^\star)) - \log\det(\Sigma_\alpha(\emptyset))} \geq 1 - \frac{1}{e}, \tag{6.11}$$

where the approximation factor $1 - 1/e$ *in* (6.11) *is the best one can achieve in polynomial time for this problem.*

Furthermore, the time complexity of Algorithm 12 is $O(cr(nK)^{2.4})$. ◇

Algorithm 12 Approximation Algorithm for $(\mathcal{P}_2)/(\mathcal{P}_4)$

Input: $h_\alpha(\mathcal{S}) = \log\det\left(\Sigma_\alpha(\mathcal{S})\right)$, where $\alpha \in \{\hat{x}_{0:K}, \hat{z}_K\}$ for $k \in [0, K]$, and a bound on the number r of sensors used to minimize $h_\alpha(\mathcal{S})$.
Output: Approximate solution $\mathcal{S}_\alpha$ for $(\mathcal{P}_2)/(\mathcal{P}_4)$.
  $\mathcal{S}_\alpha \leftarrow \emptyset$, $i \leftarrow 0$
  while $i < r$ **do**
    $a_i \leftarrow a' \in \arg\max_{a \in [c]\setminus\mathcal{S}_\alpha} \left(h_\alpha(\mathcal{S}_\alpha) - h_\alpha(\mathcal{S}_\alpha \cup \{a\})\right)$
    $\mathcal{S}_\alpha \leftarrow \mathcal{S}_\alpha \cup \{a_i\}$, $i \leftarrow i + 1$
  end while

Notice that from Theorem 12 it follows the approximation quality depends on n, r and $\max(\sigma_0^2, \sigma_w^2)$ as expected from a design perspective. Specifically, by increasing the state space size n, requesting a smaller sensor set by decreasing r, or incurring a noise of greater variance should all push the quality of the approximation level downwards.

Limits on Selecting Sensors in DTFOS

Next, we provide explicit bounds on the variance of the state-uncertainty estimation error, while exploring the trade-off with the following quantities: (*i*) the length of the estimation horizon $[0, K]$; (*ii*) the number of placed sensors r; and (*iii*) the characteristics of the noises w_k and v_k. In particular, the next result imposes limitations on the assessment of the results that cannot be overcome.

Theorem 13. *Let* $\sigma_0^{(-1)} \equiv \max_{i\in[n]}[\mathbb{C}(x_0)^{-1}]_{ii}$, $\sigma_w^{(-1)} \equiv \max_{i\in[n(K+1)]}[\mathbb{C}(w_{0:K})^{-1}]_{ii}$, *and* $\sigma_v^{(-1)} \equiv \|\mathbb{C}(v_{1:K})^{-1}\|_2$. *Also, denote by* $\bar{L}$ *the matrix* $L_{0:K}L_{0:K}^\top$. *Then, the following inequality holds for the variance of the error of the minimum variance estimator* $\hat{z}_K$*:*

$$tr(\Sigma_{\hat{z}_K}) \geq \frac{n(K+1)}{r\sigma_v^{(-1)}\|C\|_2^2\|\bar{L}\|_2 + \max\{\sigma_0^{(-1)}, \sigma_w^{(-1)}\}}. \tag{6.12}$$

◇

In other words, for constant $\|C\|_2^2$ and $\|\bar{L}\|_2$, (6.12) implies that the state-uncertainty estimation error used to assess the validity of the model (6.1) is bounded by a quantity that decreases as the number of placed sensors r increases, and increases as the system's state size or the horizon K increases. Subsequently, it implies that $\mathrm{tr}(\Sigma_{\hat{z}_K})$ can decrease only inversely proportional with the number r of placed sensors, and, as a result, increasing the number r to reduce the variance of the error of the minimum variance linear estimator is ineffective. Additionally, the bound in (6.12) increases linearly with the system's state size, which imposes additional fundamental limitations for large-scale DTFOS.

Lastly, we notice that similar arguments and fundamental bounds can be readily derived for the variance of the batch-state estimator error, i.e., $\mathrm{tr}(\Sigma_{\hat{x}_{0:K}})$, by following the same steps as in [58].

6.4. EEG Sensor Placement

In this section, we propose to study $(\mathcal{P}_1)$-$(\mathcal{P}_4)$ in a real-world application setting collected by the BCI2000 system with a sampling rate of 160Hz [176]. Specifically, we consider 64-channel EEG data set which records the brain activity of 10 subjects (S001-S010) when they are performing motor and imagery tasks [177]. Each subject sits in front of a screen where targets might appear at the right/left/top/bottom side of the screen. Upon noticing the target, each subject is asked to open and close the corresponding fists or feet as a function of where the target appears. Each individual performed 14 experimental runs consisting of one minute with eyes open, one minute with eyes closed, and three two-minute runs of 4 interacting tasks with the target: (*Task 1*) open and close left or right fist as the target appears on either left or right side of the screen; (*Task 2*) imagine opening and closing left or right fist as the target appears on either left or right side of the screen; (*Task 3*) open and close both fists or both feet as the target appears on either the top or the bottom of the screen; and (*Task 4*) imagine opening and closing both fists or both feet as the target appears on either the top or the bottom of the screen.

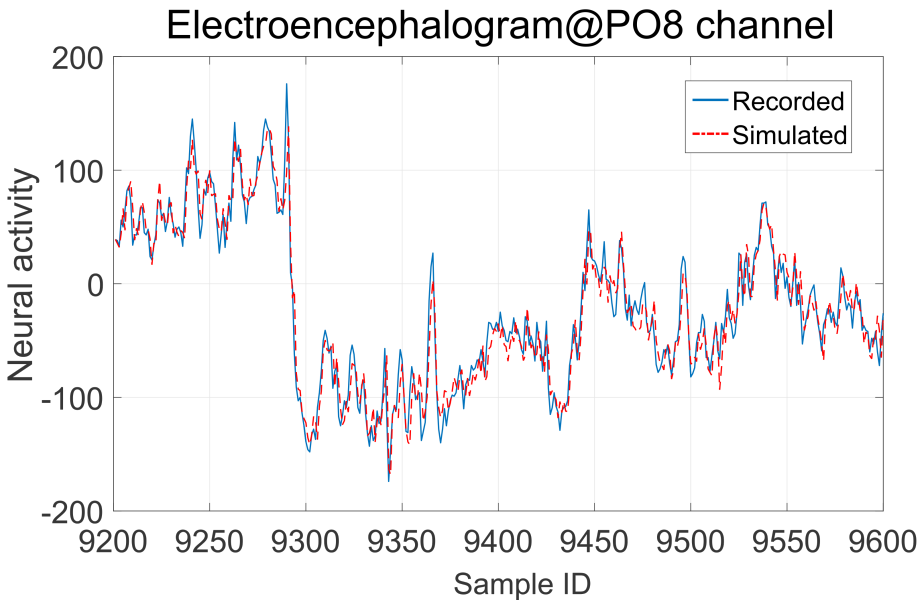


Figure 2: EEG data recorded and the simulated using DTFOS at the EEG channel PO_8.

First, we estimated the parameters of the DTFOS for the different tasks[2], which can be modeled by DTFOS as argued in [170]. To illustrate the modeling capabilities of the proposed DTFOS model, in Figure 2 we contrast the recorded data at location PO_8 against the one simulated using the DTFOS identified. It is worth mention that similar performances are achieved across different channels, subjects and tasks. Besides, the fractional order exponents range from 0.34 to 1.04 across different tasks, which provides evidence that these could not be properly modeled by linear time-invariant systems – see [170] for further details. Lastly, we considered that the initial state, disturbance, and measurement noise follow a normal distribution with zero mean and covariance described by the identity matrix (both with appropriate dimensions).

Initial State-Uncertainty Estimation Error

First, we considered a single subject (S002), and determined the different DTFOS systems associated with the four different tasks. We applied Algorithm 11, with $\alpha = \hat{z}_K$ and $K = 7$, to solve $(\mathcal{P}_1)$, and in Figure 3-(a) we plot the minimal number of sensors required as a function of required initial state-uncertainty $\log\det$ estimation error for the different tasks. The following observations are due: (i) given the same level of initial state-uncertainty estimation error required, the minimal number of sensors varies slightly when the subject is performing different tasks; and (ii) given a task, the initial state-uncertainty $\log\det$ error exhibits supermodular properties (see Theorem 10).

To address $(\mathcal{P}_2)$, i.e., to evaluate the achievable levels of the initial state-uncertainty estimation error given different sensor deployment budgets, we resorted to Algorithm 12 with

[2]The identification techniques used were introduced in [178], and the software implementation can be found at https://github.com/urashima9616/ DFOS_Sensor_Selection.

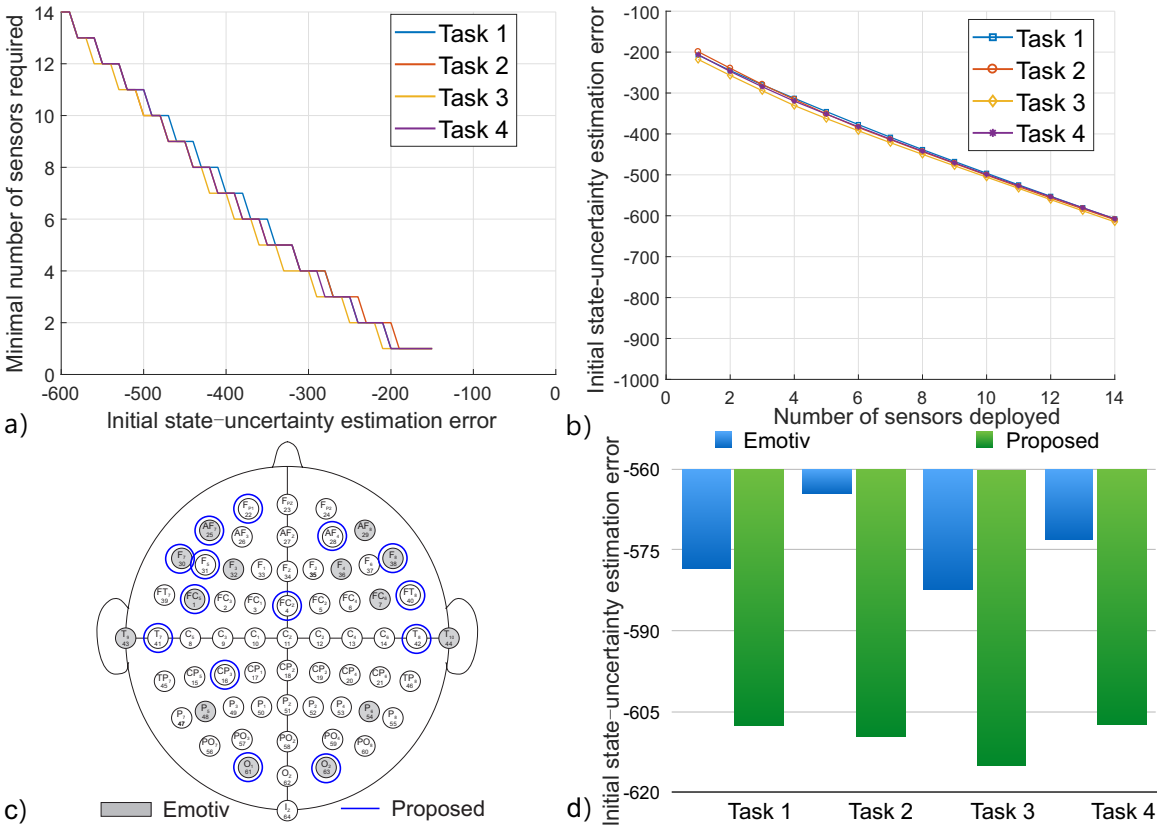


Figure 3: (a) Minimal sensor placement to achieve a prescribed initial state-uncertainty estimation errors. (b) initial state-uncertainty log det errors achieved given different sensor budgets. (c) The 64-channel geodesic sensor distribution for measurement of EEG, where the sensors in gray represent those of the Emotiv EPOC and the ones in red are those returned by Algorithm 12 when solving $(\mathcal{P}_2)$ (that relieved to be the same for all 4 tasks), given the identified DTFOS and a deployment budget of 14 sensors. (d) initial state-uncertainty log det estimation errors associated with the highlighted sensor placements in (c).

$\alpha = \hat{z}_K$ and $K = 7$. In Figure 3-(b), we present the summary of the results, namely, the log det errors given different cardinality-constrains under the 4 tasks. It can be observed that the information gain, i.e., the improvement on estimation errors, is diminishing as the number of sensors used increases – as predicted by Theorem 12.

Additionally, we considered the deployment of 14 sensors, which is the same number of sensors available in some of the current EEG wearable technology, e.g., the Emotiv EPOC [179]. In Figure 3-(c), we report the sensor deployment returned by Algorithm 12 when solving $(\mathcal{P}_2)$, which revealed to be the same across all 4 tasks (for the same individual). Specifically, we circled in blue the 14 sensors determined by our framework, whereas the Emotiv EPOC [179] sensors are colored in gray. From Figure 3-(c), we first notice that the sensor distribution pattern of Emotiv EPOC is symmetrical, whereas the Algorithm 12 places the sensors asymmetrically. Moreover, even though some of the locations are fairly close to each other (e.g., 41/43, 16/48, 40/38, 42/44, and 14/7), it turns out that only 5 out of 14 locations (i.e., 25, 29, 44, 48, 63) are identical, and Emotiv EPOC does not consider sensors

23, 4, and 18.

Subsequently, we assessed how the different sensor distributions, i.e., the proposed by our framework and the proposed by Emotiv EPOC, affect the estimation errors. In Figure 3-(d), we report the initial state-uncertainty estimation error across the different tasks. It is worth noticing that the sensors considered by our framework perform considerably better than the sensor distribution used by the Emotiv EPOC. Specifically, the $\log\det$ estimation errors attained by the proposed sensor placement are smaller compared to those of Emotiv EPOC. In fact, it presents considerable gains across the different tasks, and, in particular, in Tasks 2 and 4 that require the use of imagination instead of the motor skills. Lastly, it is important to notice that the same sensor placement performs almost equally well across different tasks. Therefore, these results support the fact that from the point-of-view of an initial state-uncertainty estimation error using a model-based approach, the sensors' locations of the commercial EEG devices should be re-designed so to ensure better estimation performance.

Batch-State Estimation Error

Now, we address the batch-state estimation problems proposed in $(\mathcal{P}_3)$-$(\mathcal{P}_4)$. Towards this goal, we consider the DTFOS corresponding to the four different tasks for the same subject (S002). In particular, we obtain the solution to $(\mathcal{P}_3)$ by relying on Algorithm 11 with $\alpha = \hat{x}_{0:K}$ and $K = 7$, whose solutions are found in Figure 4-(a) for several levels of batch-state estimation errors. From Figure 4-(a), we observe that for a specific level of batch-state estimation error the variation (across the different tasks) in the minimum number of sensors is minor. Moreover, we observe that the batch-state $\log\det$ estimation error exhibits a diminishing returns property – as per Theorem 10.

Next, we use Algorithm 12 with $\alpha = \hat{x}_{0:K}$ and $K = 7$ to tackle problem $(\mathcal{P}_4)$, i.e., to compute the achievable levels of the batch-state estimation error across several sensor placement budgets. The results are presented in Figure 4-(b), where we report the batch-state $\log\det$ estimation errors given different cardinality-constrains across the 4 tasks. Similarly to the previous figure, we notice that the gain, i.e., the improvement on the estimation error, is diminishing as the number of sensors used increases (see Theorem 12).

Furthermore, using Algorithm 12 to solve $(\mathcal{P}_4)$ with a budget of 14 sensors, we obtained the sensor placement illustrated in Figure 4-(c). In Figure 4-(c), we circle in red the 14 sensors' placement found by our framework, whereas we depict the Emotiv EPOC [179] sensors in gray. Notice that the sensor placement obtained turned out to be the same across all 4 tasks for the same individual and it is asymmetrical, in contrast with the one of the Emotiv EPOC. Notably, only 5 out of 14 locations (i.e., 25, 29, 44, 48, 63) in both cases are the same, even though some of the sensor locations are close to each other (e.g., 41/43, 16/48, 40/38, 42/44, and 14/7). Moreover, the Emotiv EPOC does not consider the sensors 23, 4, and 18. In Figure 4-(d), we compared the batch-state estimation errors (across the different tasks) of the sensor deployment returned by our framework against the sensors placement of the Emotiv EPOC. We make the following observations: first, the sensors considered by our framework perform considerably better than the sensor distribution used by the Emotiv EPOC, especially with respect to Tasks 2 and 4. Furthermore, it is worth mention that the same sensor placement returned by our framework performs almost equally well across the

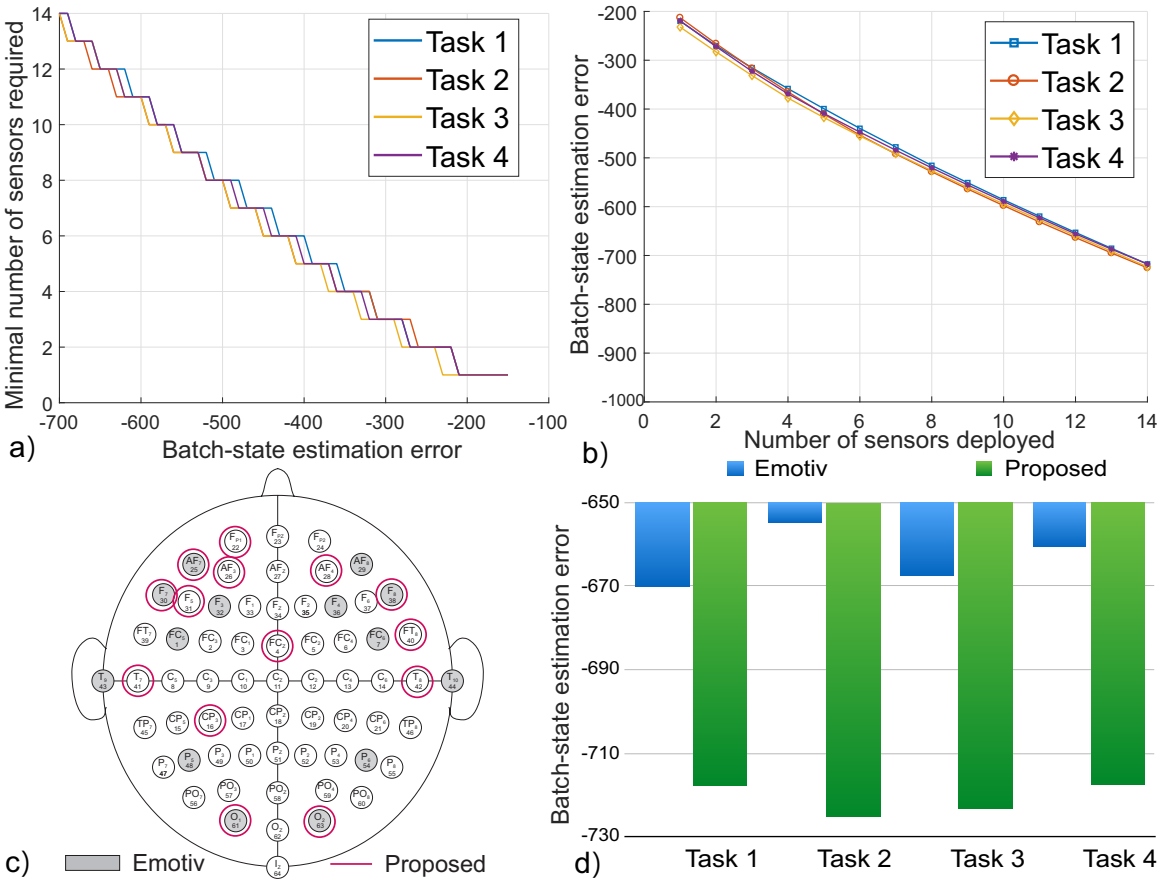


Figure 4: (a) Minimal sensor placement to achieve a prescribed batch-state estimation errors. (b) batch-state log det errors achieved given different sensor budgets. (c) The 64-channel geodesic sensor distribution for measurement of EEG, where the sensors in gray represent those of the Emotiv EPOC and the ones in red are those returned by Algorithm 12 when solving ($\mathcal{P}_4$) (that relieved to be the same for all 4 tasks), given the identified DTFOS and a deployment budget of 14 sensors. (d) batch-state log det estimation errors associated with the highlighted sensor placements in (c).

different tasks.

Assessment of Inter-subject variability

To assess how the inter-subject variability of brain dynamics affects the sensor selection under a fixed budget, we next consider a set of experiments where we solve $\mathcal{P}_2$ and $\mathcal{P}_4$ for 10 subjects across the four different tasks (Task 1-4). In particular, based on the identified DTFOS associated with the four tasks for a subject, we apply Algorithm 2 to obtain the placement of sensors given a deployment budget of 14 sensors by minimizing (i) initial state-uncertainty estimation error, and (ii) batch-state estimation error, respectively. In addition, we also identify the most voted 14 sensor locations based on the poll of sub-optimal solutions returned when solving $\mathcal{P}_2$ and $\mathcal{P}_4$ individually for all 10 subjects. The results are summarized in Figure 5-(a-d). We use heat maps to show the distribution of sensor locations returned by solving $\mathcal{P}_2$ and $\mathcal{P}_4$ individually in Figure 5-(a) and 5-(c), respectively.

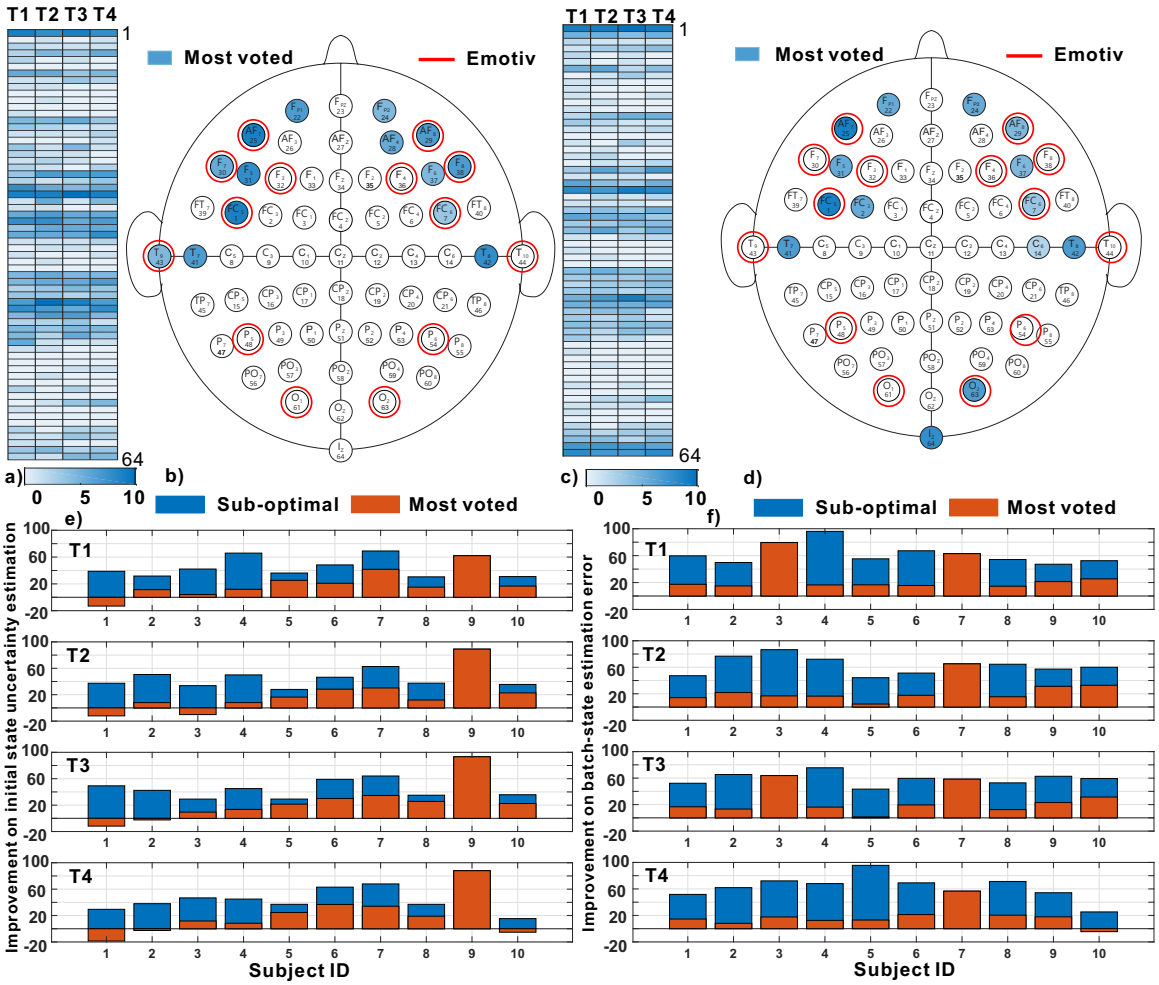


Figure 5: (a-b) The 64-channel geodesic sensor distribution over 10 subjects under Task 1-4 and the most voted deployment given a 14-sensor budget by minimizing (a) the initial state-uncertainty estimation error and (b) batch-state estimation error. (c-d) The improvement on (c) initial state-uncertainty estimation error and (d) batch-state estimation error when (i) the sub-optimal 14-sensor deployment returned by Algorithm 2 individually (blue bar) and (ii) the most voted 14-sensor deployment by 10 subjects (red bar) are considered.

We can make the following remarks: (i) there exists a noticeable degree of inter-subject variabilities in the sensor deployment. This can be evidenced by the fact that the chosen sensors span over $40, 41, 46, 46$ and $37, 43, 43, 46$ different locations across the 4 different tasks when solving $\mathcal{P}_2$ and $\mathcal{P}4$, respectively. This suggests that the underlying brain dynamics are subject to remarkable individual heterogeneities even in response to the same set of tasks. Subsequently, the best-possible sensor schemes have to be designed to be individual-specific.

At the same time, there is good percentage of agreement and lower loss of performance when planning homogeneous commercial solutions. Specifically, sensors $1, 25, 29, 41, 42$ are almost unanimously chosen by all 10 subjects, as result of solving both $\mathcal{P}_2$ and $\mathcal{P}_4$. This strongly hinges that (for the proposed tasks) there seems to be some fundamental underlying dynamics that enables the state estimation and this subset of sensors are responsible for

accessing them. To see this more clearly, we report the top voted 14 sensor locations as an average case and color them in descending order of consensus as a function of darkness in Figure 5-(b) and (d) – as comparison, we also report the sensor deployment proposed by Emotive EPOC with red circles in both geodesic maps. The following observations are due. First of all, notice that all 14 sensors are voted by at least 5 subjects while sensor $1, 25, 29, 41, 42$ are chosen by at least 8 subjects. Secondly, when considering the minimization of batch-state estimation errors instead of initial state-uncertainty estimation errors, the sensor deployment can be very different. For instance, sensor 63 and 64 are critically important when solving $\mathcal{P}_4$ (as 8 out of 10 subjects choose them) whereas their influence to the initial state-uncertainty estimation are out-weighted by other sensors. Third, similar with our previous case study, only $7/14$ and $6/14$ most voted sensor deployment by 10 subjects are identical to those proposed by Emotiv EPOC when solving $\mathcal{P}_2$ and $\mathcal{P}_4$, respectively. This potentially suggests the need of the redesign of Emotiv provided the estimation performance we setup in our study, since under our proposed approach the most voted deployment and sub-optimal individual deployment (as returned by the algorithms proposed in this chapter) achieve better performance. Specifically, we show the improvement on the $\log\det$ estimation error of both initial state-uncertainty and batch state over the one by Emotiv EPOC when the most voted deployment (red bar) and the sub-optimal individual deployment (blue bar) are employed – see Figure 5-(e) and (f), respectively. The positive improvement suggests that our proposed deployment is better than that of Emotiv EPOC. We notice that the sub-optimal deployment returned by solving $\mathcal{P}_2$ and $\mathcal{P}_4$ individually improves the estimation error significantly in all cases, which is aligned with the results of our previous case study on a single subject. Overall, based on the aforementioned observations one could conclude that the sensors' locations of the commercial EEG devices could be re-designed to enhance both their initial state and batch-state estimation error performance.

Discussion of the results

We first notice that the proposed sensor locations seem to cope better with scenarios where the neuro-activation is not as well understood as the motor-related tasks (Tasks 1 and 3), e.g., imagining actions associated with Tasks 2 and 4. Additionally, we emphasize that the time-window considered was small ($K = 7$), since aimed to attain real-time estimation. As a result, it is expected to obtain even better performance results if we increase the size window. Besides, the cumulative error increases faster in the Emotiv EPOC.

Although in our case study we relied on EEG data, it is expected that the proposed problems have distinct value in different biological settings – as explained in the introduction. Specifically, in problems where signals are well modeled by the proposed DTFOS, as it is the case of other physiological signals such as electromyograms (EMG) and electrocardiograms (ECG) [170].

6.5. Concluding Remarks & Future Work

We considered biologically motivated discrete-time linear fractional-order systems and studied the trade-off between the sensor placement and the properties that pertain to Kalman-like filter performance. Specifically, we formalized the sensor placement problems in context of four different (but related) problems pertaining to the sensor placement to minimize the

state-uncertainty and batch-state estimation error. We showed that these problems are NP-hard, and we presented polynomial approximation strategies for their solution that have sub-optimality guarantees.

Additionally, we explored the different problems in the context of real EEG data during a period of time where the individuals performed four different tasks. The results obtained support the capability of the proposed framework to deal with critical sensing deployment problems, and unveiled that the number and location of the sensors vary across tasks and subjects for the same experimental setup. Furthermore, we argue that these locations are not compatible with those used by state-of-the-art EEG wearables (e.g., Emotiv EPOC), which supports the need for further research and re-design of future EEG wearables that aim to attain a specified estimation performance for a given task.

Future research will consider the multi-scenario case, where the sensor placement has to consistently and reliably consider possible dynamics, e.g., multi-tasks simultaneously when EEG is considered. Additionally, we propose to validate the presented methodology when a large cohort of individuals and tasks is considered.

6.6. Appendix: Proof of the Results

Proof of Theorem 9: We prove that $(\mathcal{P}_1)$-$(\mathcal{P}_4)$ are NP-hard by focusing on the case where (i) the measurement matrix C is the identity matrix ($C = I$), (ii) the measurement noise v_k is Gaussian with zero mean and covariance the identity matrix ($\mathbb{C}(v_k) = I$), and (iii) $K = 0$, in which case $z_K = x_{0:K} = x_0$, and thus, $(\mathcal{P}_1)$-$(\mathcal{P}_4)$ are equivalent to the following problem:

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{minimize}} \quad & \log\det(\Sigma(\hat{x}_0)) \\ \text{subject to} \quad & |\mathcal{S}| \leq r. \end{aligned} \tag{6.13}$$

In more detail, we prove that the problem in (6.13) is NP-hard by proving that the following problem is equivalent to (6.13), and that it is NP-hard:

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{minimize}} \quad & \mathbb{H}(x_0|y_0(\mathcal{S})) \\ \text{subject to} \quad & |\mathcal{S}| \leq r, \end{aligned} \tag{6.14}$$

where $\mathbb{H}(x_0|\mathcal{S})$ is the entropy of x_0 given the measurements $y_0(\mathcal{S})$ collected by the selected sensors in $\mathcal{S}$ at time $k = 0$. Specifically, we prove that the problem in (6.14) is NP-hard by proving it is equivalent to the entropy maximization problem

$$\begin{aligned} \underset{\mathcal{S}\subseteq\{1,2,\ldots,c\}}{\text{maximize}} \quad & \mathbb{H}(y_0(\mathcal{S})) \\ \text{subject to} \quad & |\mathcal{S}| \leq r, \end{aligned} \tag{6.15}$$

which we prove to have an equivalent instance to the NP-hard instance of the entropy maximization problem in [180].

To prove that the problems in (6.13) and (6.14) are equivalent, we use [175, Proposition 2],

which implies for $K = 0$ that

$$\mathbb{H}(x_0|y_0(\mathcal{S})) = \frac{\log\det(\Sigma(\hat{x}_0))}{2} + \frac{n\log(2\pi e)}{2}, \tag{6.16}$$

where $\hat{x}_0$ is the minimum mean square estimator of x_0 given y_0 [123, Appendix E]. Therefore, (6.16) implies that minimizing the objective in (6.13) is equivalent to minimizing the objective in (6.14).

We next prove that the problems in (6.14) and (6.15) are equivalent. To this end, first observe that

$$\mathbb{H}(x_0|y_0(\mathcal{S})) = \mathbb{H}(x_0) + \mathbb{H}(y_0(\mathcal{S})|x_0) - \mathbb{H}(y_0(\mathcal{S})), \tag{6.17}$$

(we derive (6.17) using the conditional entropy chain rule [181]), as well as that:

- $\mathbb{H}(x_0)$ is constant with respect to $\mathcal{S}$.
- $\mathbb{H}(y_0(\mathcal{S})|x_0)$ is constant for $C = I$, $\mathbb{C}(v_0) = I$, and for fixed $|\mathcal{S}|$, (which is the case throughout this proof, since due to the monotonicity of the $\log\det$ and the entropy, in all problems in (6.13), (6.14), and (6.15), it is $|\mathcal{S}| = r$ for any $\mathcal{S}$ that solves (6.13), (6.14), and (6.15)), since if $y_0^{(i)}(\mathcal{S})$ ($x_0^{(i)}$, respectively) denotes the i-th element of $y_0(\mathcal{S})$ (x_0, respectively), then

$$\begin{aligned}
&\mathbb{H}(y_0(\mathcal{S})|x_0) \\
&\overset{a}{=} \sum_{i=1}^{|\mathcal{S}|} \mathbb{H}(y_0^{(i)}(\mathcal{S})|x_0, y_0^{(1)}(\mathcal{S}), \ldots, y_0^{(i-1)}(\mathcal{S})) \\
&\overset{b}{=} \sum_{i\in\mathcal{S}} \mathbb{H}(y_0^{(i)}(\mathcal{S})|x_0^{(i)}) \\
&\overset{c}{=} \frac{1}{2}\log(2\pi e)|\mathcal{S}|
\end{aligned}$$

  In particular, equalities (a), (b) and (c) hold for the following reasons, respectively: (a) holds due to the conditional entropy chain rule [181]; (b) holds since for all $j \neq i$, $y_0^{(i)}$ given $x_0^{(i)}$ is independent of $x_0^{(j)}$ and $y_0^{(j)}$; and (c) holds since $y_0^{(i)}$ given $x_0^{(i)}$ is Gaussian with variance 1, since we consider the case where $\mathbb{C}(v_0) = I$.

Due to (6.17) and the latter two observations, we conclude that (6.14) is equivalent to (6.15).

Given the equivalence of (6.14) and (6.15), we conclude the proof by finding an instance for the problem in (6.13) that is equivalent to an instance for the problem in (6.15) that is NP-hard. In particular, consider Σ to be any $n \times n$ matrix that makes the entropy maximization problem in [180, Theorem 1] NP-hard: Σ is a positive definite symmetric with all the diagonal entries equal to $3n$, and all the off-diagonal entries equal to 0 or 1. The problem (6.15) is NP-hard if we can find an instance for the problem in (6.13) where $y_0(\{1, 2, \ldots, c\})$ is a Gaussian random variable with covariance Σ. Indeed, let $\mathbb{C}(x_0)$ be any positive definite symmetric matrix with all diagonal entries equal to $3n-1$, and all off-diagonal entries equal to 0 or 1 (Σ_0 is positive define by construction, since it is both diagonally dominant, and

as a result invertible, and symmetric). For this selection of parameters, y_0 has covariance Σ; the reason is threefold: (i) $y_0 = x_0 + v_0$, (ii) x_0 and v_0 are Gaussian with covariances Σ_0 and I, respectively, and (iii) x_0 and v_0 are uncorrelated; as a result, y_0 is Gaussian with covariance $\Sigma_0 + I = \Sigma$. ■

Proof of Theorem 10: In the following paragraphs, we only present the proof for

$$\log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right),$$

since the proof for $\log\det\left(\Sigma_{\hat{x}_{0:K}}(\mathcal{S})\right)$ follows similar steps. In particular, we complete the proof for $\log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right)$ in three steps: (i) we prove its closed formula; (i) we show that it is non-increasing; and (iii) we prove that it is supermodular.

Given any finite estimation horizon $[0, K]$, we first prove that

$$\log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right) = \\ -\log\det\left(\sum_{i=1}^{c} s_i L_{0:K}^\top M^{(i)} L_{0:K} + \mathbb{C}(z_K)^{-1}\right). \tag{6.18}$$

In particular, as in the proof of Lemma 1 in [58], we let $S_{0:K}$ denote the block diagonal matrix with diagonal elements $K+1$ copies of the sensor placement matrix S in (6.8). Then, using the Woodbury matrix identity [93, Corollary 2.8.8] at (6.3), we obtain $\Sigma_{\hat{z}_K}(\mathcal{S}) = (\mathcal{O}_K^\top \Xi \mathcal{O}_K + \mathbb{C}(z_K)^{-1})^{-1}$, where $\Xi \equiv (S_{0:K}\mathbb{C}(v_{0:K})S_{0:K}^\top)^{-1}$, and due to the definitions of $S_{0:K}$, $C_{0:K}$ and $\mathcal{O}_K = S_{0:K}C_{0:K}L_{0:K}$. Moreover, due to the definition of $S_{0:K}$, i.e., it contains block of matrices that are only zero or identity matrices, and because $\mathbb{C}(v_{0:K})$ is block diagonal, it follows that $\Xi = S_{0:K}\mathbb{C}(v_{0:K})^{-1}S_{0:K}^\top$, which can be verified by direct calculation. Overall, we obtain

$$\Sigma_{\hat{z}_K}(\mathcal{S}) = (L_{0:K}^\top C_{0:K}^\top \Lambda \mathbb{C}(v_{0:K})^{-1} \Lambda C_{0:K} L_{0:K} + \mathbb{C}(z_K)^{-1})^{-1},$$

where $\Lambda \equiv S_{0:K}^\top S_{0:K}$. Now, since Λ and $\mathbb{C}(v_{0:K})^{-1}$ are block diagonal, and the blocks of Λ are either identity or zero matrices, $\mathbb{C}(v_{0:K})^{-1}\Lambda = \Lambda\mathbb{C}(v_{0:K})^{-1}$. Furthermore, the definition of $S_{0:K}$ implies that $\Lambda^2 = \Lambda$. Thus, it follows that

$$\Sigma_{\hat{z}_K}(\mathcal{S}) = (L_{0:K}^\top C_{0:K}^\top \Lambda \mathbb{C}(v_{0:K})^{-1} C_{0:K} L_{0:K} + \mathbb{C}(z_K)^{-1})^{-1}. \tag{6.19}$$

For the last step, observe first that $\Lambda = \sum_{i=1}^{c} s_i I^{(i)}$, so

$$L_{0:K}^\top C_{0:K}^\top \Lambda \mathbb{C}(v_{0:K})^{-1} C_{0:K} L_{0:K} \\ = \sum_{i=1}^{c} s_i L_{0:K}^\top C_{0:K}^\top I^{(i)} \mathbb{C}(v_{0:K})^{-1} C_{0:K} L_{0:K} \tag{6.20}$$

$$= \sum_{i=1}^{c} s_i L_{0:K}^\top M^{(i)} L_{0:K}, \tag{6.21}$$

where we derive (6.21) from (6.20) by using for $I^{(i)}$ the reverse steps to the ones we used for Λ to derive (6.19).

Next, to prove that $\log\det\left(\Sigma_{\hat{z}_K}(\mathcal{S})\right)$ is a non-increasing set function in the choice of the sensors $\mathcal{S}$, we follow similar steps to those in Theorem 2 in [58]. Specifically, consider $\mathcal{S} \subseteq \mathcal{S}'$, and observe that (6.18) and from [93, Theorem 8.4.9], $\Sigma_{\hat{z}_K}(\mathcal{S}') \preceq \Sigma_{\hat{z}_K}(\mathcal{S})$, since $L_{0:K}^{\top} M^{(i)} L_{0:K} \succeq 0$ and $\mathbb{C}(z_K) \succ 0$. As a result, $\log\det(\Sigma_{\hat{z}_K}(\mathcal{S}')) \leq \log\det(\Sigma_{\hat{z}_K}(\mathcal{S}))$.

Finally, to prove that $\log\det(\Sigma_{\hat{z}_K}(\mathcal{S}))$ is supermodular, we prove that $-\log\det(\Sigma_{\hat{z}_K}(\mathcal{S}))$ is submodular. In particular, recall that a function $h : 2^{[c]} \mapsto \mathbb{R}$ is submodular if and only if, for any $a \in [c]$, the function $h_a : 2^{[c]\setminus\{a\}} \mapsto \mathbb{R}$, where $h_a(\mathcal{S}) \equiv h(\mathcal{S} \cup \{a\}) - h(\mathcal{S})$, is a non-increasing set function. Therefore, to prove that $h(\mathcal{S}) = -\log\det(\Sigma_{\hat{z}_K}(\mathcal{S}))$ is submodular, we may prove that the $h_a(\mathcal{S})$ is a non-increasing set function. To this end, we denote $\sum_{i=1}^{c} s_i L_{0:K}^{\top} M^{(i)} L_{0:K}$ in (6.18) by $M(\mathcal{S})$, and follow similar steps to those in the proof of Theorem 6 in [52]. Specifically, we note that

$$\begin{aligned} h_a(\mathcal{S}) &= \log\det(M(\mathcal{S}\cup\{a\}) + \mathbb{C}(z_K)^{-1}) - \\ &\qquad\qquad \log\det(M(\mathcal{S}) + \mathbb{C}(z_K)^{-1}) \\ &= \log\det(M(\mathcal{S}) + M(\{a\}) + \mathbb{C}(z_K)^{-1}) - \\ &\qquad\qquad \log\det(M(\mathcal{S}) + \mathbb{C}(z_K)^{-1}). \end{aligned}$$

For $\mathcal{S} \subseteq \mathcal{S}'$ and $t \in [0,1]$, define $\Phi(t) \equiv \mathbb{C}(z_K)^{-1} + M(\mathcal{S}) + t(M(\mathcal{S}') - M(\mathcal{S}))$ and

$$g(t) \equiv \log\det\left(\Phi(t) + M(\{a\})\right) - \log\det\left(\Phi(t)\right).$$

Then, $g(0) = h_a(\mathcal{S})$ and $g(1) = h_a(\mathcal{S}')$. Moreover, since

$$\frac{d\log\det(\Phi(t))}{dt} = \mathrm{tr}\left(\Phi(t)^{-1}\frac{d\Phi(t)}{dt}\right)$$

(as in eq. (43) in [95]), it follows that

$$\dot{g}(t) = \mathrm{tr}\left[((\Phi(t) + M(\{a\}))^{-1} - \Phi(t)^{-1})F\right],$$

where $F \equiv M(\mathcal{S}') - M(\mathcal{S})$. Furthermore, from [93, Proposition 8.5.5], we have that

$$(\Phi(t) + M(\{a\}))^{-1} - \Phi(t)^{-1} \preceq 0,$$

where $\Phi(t)$ is invertible since $\mathbb{C}(z_K)^{-1} \succ 0$, $M(\mathcal{S}) \succeq 0$, and $M(\mathcal{S}') \succeq M(\mathcal{S}))$. Since also $F \succeq 0$, from [93, Corollary 8.3.6], it readily follows that

$$\lambda_{\max}[((\Phi(t) + M(\{a\}))^{-1} - \Phi(t)^{-1})F] \leq 0.$$

Thus, $\dot{g}(t) \leq 0$, and $h_a(\mathcal{S}') = g(1) = g(0) + \int_0^1 \dot{g}(t)dt \leq g(0) = h_a(\mathcal{S})$, i.e., h_a is non-increasing.

■

Proof of Theorem 11: The proof of the theorem in attained in three main steps: (*i*) we prove (6.9); (*ii*) we prove (6.10); and (*iii*) we prove the computational complexity of Algorithm 11.

To prove (6.9), let $\mathcal{S}_0, \mathcal{S}_1, \ldots$ be the sequence of sets selected by Algorithm 11 and l the smallest index such that $\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}_l\right) \leq R$. Therefore, $\mathcal{S}_l$ is the set that Algorithm 11 returns. To prove (6.9), we first observe that Theorem 10 implies $\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}\right))$ is a supermodular and non-increasing. Then, from [94], we have that

$$\frac{l}{|\mathcal{S}^\star|} \leq 1 + \log \frac{\log\det\left(\Sigma_{\hat{z}_K}, \emptyset\right) - \log\det\left(\Sigma_{\hat{z}_K}, [c]\right)}{\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}_{l-1}\right) - \log\det\left(\Sigma_{\hat{z}_K}, [c]\right)}.$$

Now, l is the first time that $\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}_l\right) \leq R$, and as a result $\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}_{l-1}\right) > R$, and, as a consequence, we have that (6.10) holds. Furthermore, $\log\det\left(\Sigma_{\hat{z}_K}, \emptyset\right) = \log\det\left(\mathbb{C}(z_K)\right)$, and from the geometric-arithmetic mean inequality, we obtain that

$$\begin{aligned}
\log\det\left(\mathbb{C}(z_K)\right) &\leq n(K+1)\log\frac{\mathrm{tr}(\mathbb{C}(z_K))}{n(K+1)} \\
&\leq n(K+1)\log\frac{n(K+1)\max(\sigma_0^2, \sigma_w^2)}{n(K+1)} \\
&= n(K+1)\log\max(\sigma_0^2, \sigma_w^2).
\end{aligned}$$

Finally, to prove the computational complexity of Algorithm 11, note that the `while` loop is repeated for at most c times. Moreover, the complexity to compute the determinant of an $n(K+1) \times n(K+1)$ matrix, using the Coppersmith-Winograd algorithm [97], is $O((nK)^{2.4})$, which is also the complexity incurred by the multiplication between such two matrices. Additionally, the determinant of at most $c+1$ matrices must be computed so that the

$$\arg\max_{a \in [c]\setminus\mathcal{S}} \left(\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}\right) - \log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}\cup\{a\}\right)\right)$$

can be computed. Also, $O(c)$ time is required to find a maximum element between c available. Therefore, the overall computational complexity of Algorithm 11 is dominated by $O(c^2(nK)^{2.4})$. ■

Proof of Theorem 12: In the following paragraphs, we complete the proof of the theorem in three steps: (*i*) we prove (6.11); (*ii*) we prove that the approximation factor $1 - 1/e$ in (6.11) is the best one can achieve in polynomial time for $(\mathcal{P}_2)/(\mathcal{P}_4)$; and (*iii*) we discuss the computational complexity of the algorithm.

To prove (6.11), we first observe that Theorem 10 implies $\log\det\left(\Sigma_{\hat{z}_K}, \mathcal{S}\right) - \log\det\left(\Sigma_{\hat{z}_K}, \emptyset\right)$ is a supermodular, non-increasing and non-positive set function. Consequently, the results from [94] can be invoked to obtain (6.11).

To prove that the approximation factor $1 - 1/e$ in (6.11) is the best one can achieve in polynomial time for $(\mathcal{P}_2)/(\mathcal{P}_4)$, we recall that in the worst-case $(\mathcal{P}_2)/(\mathcal{P}_4)$ are equivalent to the minimal observability problem (see proof of Theorem 9). Then, the result follows by noticing that the minimal observability problem has the same computational complexity as the set cover problem [7], which cannot be approximated in polynomial time with a factor better than $1 - 1/e$ [13].

Finally, the computational complexity of Algorithm 12 can be derived by following the same steps and reasoning as the one proposed in the proof of Theorem 11 to show the time complexity of Algorithm 11. ■

Proof of Theorem 13: Since the arithmetic mean of a finite set of positive numbers is at least as large as their harmonic mean, the following inequality holds:

$$\mathrm{tr}(\Sigma_{\hat{z}_K}) \geq \frac{(n(K+1))^2}{\mathrm{tr}\left(\sum_{i=1}^{c} s_i L_{0:K}^\top M^{(i)} L_{0:K} + \mathbb{C}(z_K)^{-1}\right)}, \tag{6.22}$$

where we used the closed form for $\Sigma_{\hat{z}_K}$ proved in Theorem 10.

Furthermore, in the denominator of (6.22), for the first term it is $\mathrm{tr}(\sum_{i=1}^{c} s_i L_{0:K}^\top M^{(i)} L_{0:K}) = \sum_{i=1}^{c} s_i \mathrm{tr}(M^{(i)} L_{0:K} L_{0:K}^\top)$, where

$$\mathrm{tr}(L_{0:K}^\top M^{(i)} L_{0:K}) \leq n(K+1)\|\bar{C}\|_2^2 \|\mathbb{C}(v_{1:K})^{-1}\|_2 \|L_{0:K} L_{0:K}^\top\|_2,$$

since $\|I^{(i)}\|_2 = 1$, and for the second term it is $\mathrm{tr}(\mathbb{C}(z_K)^{-1}) \leq n(K+1)\max\{\sigma_0^{(-1)}, \sigma_w^{(-1)}\}$. Therefore,

$$\mathrm{tr}\left(\sum_{i=1}^{c} s_i L_{0:K}^\top M^{(i)} L_{0:K} + \mathbb{C}(z_K)^{-1}\right) \leq$$
$$rn(K+1)\sigma_v^{(-1)}\|\bar{C}\|_2^2\|\bar{L}\|_2 + n(K+1)\max\{\sigma_0^{(-1)}, \sigma_w^{(-1)}\}.$$

Hence, $\mathrm{tr}(\Sigma_{\hat{z}_K}) \geq n(K+1)/(r\sigma_v^{(-1)}\|\bar{C}\|_2^2\|\bar{L}\|_2 + \max\{\sigma_0^{(-1)}, \sigma_w^{(-1)}\})$. ■

CHAPTER 7 : Scheduling Nonlinear Sensors for Stochastic Process Estimation

In this chapter, we focus on activating only a few sensors, among many available, to estimate the state of a stochastic process of interest. This problem is important in applications such as target tracking and simultaneous localization and mapping (SLAM). It is challenging since it involves stochastic systems whose evolution is largely unknown, sensors with nonlinear measurements, and limited operational resources that constrain the number of active sensors at each measurement step. We provide an algorithm applicable to general stochastic processes and nonlinear measurements whose time complexity is linear in the planning horizon and whose performance is a multiplicative factor $1/2$ away from the optimal performance. This is notable because the algorithm offers a significant computational advantage over the polynomial-time algorithm that achieves the best approximation factor $1/e$. In addition, for important classes of Gaussian processes and nonlinear measurements corrupted with Gaussian noise, our algorithm enjoys the same time complexity as even the state-of-the-art algorithms for linear systems and measurements. We achieve our results by proving two properties for the entropy of the batch state vector conditioned on the measurements: a) it is supermodular in the choice of the sensors; b) it has a sparsity pattern (involves block tri-diagonal matrices) that facilitates its evaluation at each sensor set.[1]

7.1. Introduction

Adversarial target tracking and capturing [129, 183], robotic navigation and autonomous construction [128], active perception and simultaneous localization and mapping (SLAM) [22] are only a few of the challenging information gathering problems that benefit from the monitoring capabilities of sensor networks [130]. These problems are challenging because:

- they involve systems whose evolution is largely unknown, modeled either as a stochastic process, such as a Gaussian process [184], or as linear or nonlinear system corrupted with process noise [129],
- they involve nonlinear sensors (e.g., cameras, radios) corrupted with noise [103],
- they involve systems that change over time [127], and as a result, necessitate both spatial and temporal deployment of sensors in the environment, increasing the total number of needed sensors, and at the same time,
- they involve operational constraints, such as limited communication bandwidth and battery life, which limit the number of sensors that can simultaneously be active in the information gathering process [131].

Due to these challenges, we focus on the following question: "How do we select, at each time, only a few of the available sensors so as to monitor effectively a system despite the above challenges?" In particular, we focus on the following sensor scheduling problem:

Problem 1. *Consider a stochastic process, whose realization at time t is denoted by $x(t)$ and a set of m sensors, whose measurements are nonlinear functions of $x(t)$, evaluated at a fixed set of K measurement times $t_1, t_2, \ldots, t_K$. In addition, suppose that at each t_k a*

[1] This chapter is based on the paper by Tzoumas et al. [182].

set of at most $s_k \leq m$ sensors can be used. Select the sensor sets so that the error of the corresponding minimum mean square error estimator of $(x(t_1), x(t_2), \ldots, x(t_K))$ is minimal among all possible sensor sets.

The reason we focus on estimating the batch state vector $(x(t_1), x(t_2), \ldots, x(t_K))$ is that in many control problems we need to have a good estimate of the trajectory taken so far, e.g., for linearisation purposes.

Literature review: There are two classes of sensor scheduling algorithms, that trade-off between the estimation accuracy of the batch state vector and their time complexity [185]: those used for Kalman filtering, and those for batch state estimation. The most relevant papers on batch state estimation are [185] and [135]. However, both of these papers focus on linear systems and measurements. The most relevant papers for Kalman filtering consider algorithms that use: myopic heuristics [115], tree pruning [137], convex optimization [119, 132, 186, 187], quadratic programming [138], Monte Carlo methods [188], or submodular function maximization [5, 139]. However, these papers focus similarly on linear or nonlinear systems and measurements, and do not consider unknown dynamics.

At the same time, [114] focuses on sensor selection algorithms for estimating stochastic processes that are, in contrast to the processes in the present chapter, spatially correlated and not temporally correlated. In more detail, in [114], $x(t_i)$ represents the value of a parameter of interest at a spatial position t_i, and is constant in time. This is notable since in [114] the proposed algorithms for sensor selection become fast when the covariance matrix of $(x(t_1), x(t_2), \ldots, x(t_K))$ is sparse (or can be approximated by a sparse matrix). Notwithstanding, this is not necessarily the case for dynamic stochastic processes, since $x(t_i)$ may be strongly correlated to the trajectory $(x(t_1), x(t_2), \ldots, x(t_{i-1}))$ taken so far in the state space.

Main contributions.:

1. We prove that Problem 1 is NP-hard.

2. We prove that the best approximation factor one can achieve in polynomial time for Problem 1, in the worst case, is $1/e$.

3. We provide Algorithm 13 for Problem 1 that:

   - for all stochastic processes and nonlinear measurements, achieves a solution that is up to a multiplicative factor $1/2$ from the optimal solution with time complexity that is only linear in the planning horizon K. This is important, since it implies that Algorithm 13 offers a significant computational advantage with negligible loss in performance over the polynomial-time algorithm that achieves the best approximation factor of $1/e$,

   - for important classes of Gaussian processes, and nonlinear measurements corrupted with Gaussian noise, has the same time complexity as state-of-the-art algorithms for linear systems and measurements. For example, for Gaussian pro-

cess such as those in target tracking, or those generated by linear or nonlinear systems corrupted with Gaussian noise, Algorithm 13 has the same time complexity as the batch state estimation algorithm in [185], and lower than the relevant Kalman filter scheduling algorithms in [119, 132].

Therefore, Algorithm 13 can enjoy both the estimation accuracy of the batch state scheduling algorithms (compared to the Kalman filtering approach, that only approximates the batch state estimation error with an upper bound [185]) and, surprisingly, even the low time complexity of the Kalman filtering scheduling algorithms for linear systems.

Technical contributions.:

1. *Supermodularity in Problem 1*: We achieve the approximation performance of Algorithm 13, and the linear dependence of its time complexity on the planning horizon, by proving that our estimation metric is a supermodular function in the choice of the utilized sensors. This is important, since this is in contrast to the case of multi-step Kalman filtering for linear systems and measurements, where the corresponding estimation metric is neither supermodular nor submodular [139] [5].

2. *Sparsity in Problem 1*: We achieve the reduced time complexity of Algorithm 13 for Gaussian processes by identifying a sparsity pattern in our estimation metric. Specifically, for Gaussian processes the time complexity of each evaluation of our metric is decided by the sparsity pattern of either the covariance of $(x(t_1), x(t_2), \ldots, x(t_K))$, or the inverse of this covariance. This is important since the two matrices are not usually sparse at the same time, even if one of them is [133].

   In more detail, we identify that for Gaussian processes such as those in target tracking, the first matrix is block tri-diagonal, whereas for those in SLAM, or those generated by linear or nonlinear systems corrupted with Gaussian noise, the second matrix is block tri-diagonal.

Notation: We denote the set of natural numbers $\{1, 2, \ldots\}$ by $\mathbb{N}$, the set of real numbers by $\mathbb{R}$, and the set $\{1, 2, \ldots, n\}$ by $[n]$ ($n \in \mathbb{N}$). The set of real numbers between 0 and 1 is denoted by $[0, 1]$, and the empty set by $\emptyset$. Given a set $\mathcal{X}$, $|\mathcal{X}|$ is its cardinality. In addition, for $n \in \mathbb{N}$, $\mathcal{X}^n$ is the n-times Cartesian product $\mathcal{X} \times \mathcal{X} \times \cdots \times \mathcal{X}$. Matrices are represented by capital letters and vectors by lower-case letters. We write $A \in \mathcal{X}^{n_1 \times n_2}$ ($n_1, n_2 \in \mathbb{N}$) to denote a matrix of n_1 rows and n_2 columns whose elements take values in $\mathcal{X}$; $A^\top$ is its transpose, and $[A]_{ij}$ is its element at the i-th row and j-th column; $\det(A)$ is its determinant. Furthermore, if A is positive definite, we write $A \succ 0$. In the latter case, A^{-1} is its inverse. I is the identity matrix; its dimension is inferred from the context. Similarly for the zero matrix 0. The $\equiv$ denotes equivalence. Moreover, for a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, Ω is the sample space, $\mathcal{F}$ the σ-field, and $\mathbb{P} : \mathcal{F} \mapsto [0, 1]$ the function that assigns probabilities to events in $\mathcal{F}$ [189]. We write $x \sim \mathcal{F}$ to denote a random variable x with probability distribution $\mathcal{F}$; $\mathbb{E}(x)$ is its expected value, and $\Sigma(x)$ its covariance. $x \sim \mathcal{N}(\mu, \Sigma)$ denotes a Gaussian random variable x with mean μ and covariance Σ; with a slight abuse of notation, we equivalently write $x \sim \mathcal{N}(\mathbb{E}(x), \Sigma(x))$. Finally, we write $x|y \sim \mathcal{G}$ to denote that x's

probability distribution given y is $\mathcal{G}$.

7.2. Problem Formulation

This section introduces the system, measurement, and scheduling models and presents the sensor scheduling problem formally.

System 2. *We consider two cases:*

- Continuous time model*: Consider the stochastic process (along with a probability space* $(\Omega, \mathcal{F}, \mathbb{P})$*):*

$$x_\omega(t) : \omega \in \Omega, t \geq t_0 \mapsto \mathbb{R}^n \tag{7.1}$$

  where $n \in \mathbb{N}$*,* t_0 *is the initial time, and* $x_\omega(t)$ *the state vector given the sample* ω*.*

- Discrete time model*: Consider the nonlinear discrete-time system:*

$$x_{k+1} = l_k(x_{1:k}), l_k \sim \mathcal{L}_k, k \in \mathbb{N} \tag{7.2}$$

  where $x_k \in \mathbb{R}^n$ *is the state vector,* $x_{1:k}$ *the batch vector* $(x_1, x_2, \ldots, x_k)$*, and* $\mathcal{L}_k$ *a probability distribution over functions* $l_k : \mathbb{R}^{nk} \mapsto \mathbb{R}^n$*.*

Because the system models (7.1) and (7.2) assume no characteristic structure, they are appropriate for modeling largely unknown dynamics. For example, an instance of (7.1) is the time-indexed Gaussian process system model:

$$x(t) \sim \mathcal{GP}(\mu(t), \Sigma(t, t')), \quad t, t' \geq t_0, \tag{7.3}$$

where $\mu(t)$ is the mean function and $\Sigma(t, t')$ is the covariance function. Similarly, an instance of (7.2) is the state-indexed Gaussian process system model:

$$x_{k+1} = l(x_k), \quad l \sim \mathcal{GP}(\mu(x), \Sigma(x, x')), x, x' \in \mathbb{R}^n. \tag{7.4}$$

Measurement Model 1. *Consider* m *nonlinear sensors that operate in discrete time:*

$$z_{i,k} = g_i(x_k) + v_{i,k}, \quad i \in [m], k \in \mathbb{N} \tag{7.5}$$

where for the continuous-time system in (7.1) *we let* $x_k := x(t_k)$ *at a pre-specified set of measurement times* $t_1, t_2, \ldots$ *and* $v_{i,k}$ *is the measurement noise of sensor* i *at time* k*.*

Assumption 6. $v_{i,k}$ *are independent across* i *and* k*. In addition,* g_i *is one-time differentiable.*

Sensor Scheduling Model 1. *The* m *sensors in* (7.5) *are used at* K *scheduled measurement times* $\{t_1, t_2, \ldots, t_K\}$*. At each* $k \in [K]$*, only* s_k *of the* m *sensors are used (*$s_k \leq m$*), resulting in the batch measurement vector* y_k*:*

$$y_k = S_k z_k, \quad k \in [K], \tag{7.6}$$

where S_k *is a sensor selection matrix, composed of sub-matrices* $[S_k]_{ij}$ *(*$i \in [s_k]$*,* $j \in [m]$*) such that* $[S_k]_{ij} = I$ *if sensor* j *is used at time* k*, and* $[S_k]_{ij} = 0$ *otherwise. We assume that a sensor can be used at most once at each* k*, and as a result, for each* i *there is one* j *such*

that $[S_k]_{ij} = I$ *while for each* j *there is at most one* i *such that* $[S_k]_{ij} = I$.

We now present the sensor scheduling problem formally:

Notation. For $i, j \in \mathbb{N}$, $\phi_{i:j} \equiv (\phi_i, \phi_{i+1}, \ldots, \phi_j)$. In addition, $\mathcal{S}_k \equiv \{j : \text{there exists } i \in [s_k], [S_k]_{ij} = I\}$: $\mathcal{S}_k$ is the set of indices that correspond to utilized sensors at t_k.

Problem 1 (Sensor Scheduling in Stochastic Processes with Nonlinear Observations)**.** *Select at each time* k *a subset of* s_k *sensors, out of the* m *sensors in* (7.5)*, to use in order to minimize the conditional entropy of* $x_{1:K}$ *given the measurements* $y_{1:K}$*:*

$$\begin{aligned} &\underset{\mathcal{S}_k \subseteq [m], k \in [K]}{\text{minimize}} \ \mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}) \\ &\text{subject to} \ \ |\mathcal{S}_k| \leq s_k, k \in [K], \end{aligned}$$

where $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K})$ *denotes the conditional entropy* $\mathbb{H}(x_{1:K}|y_{1:K})$ *of* $x_{1:K}$ *given the measurements* $y_{1:K}$*.*

The conditional entropy $\mathbb{H}(x_{1:K}|y_{1:K})$ captures the estimation accuracy of $x_{1:K}$ given $y_{1:K}$, as we explain in the following two propositions:

Proposition 6. $\mathbb{H}(x_{1:K}|y_{1:K})$ *is a constant factor away from the mutual information of* $x_{1:K}$ *and* $y_{1:K}$*. In particular:*

$$\mathbb{H}(x_{1:K}|y_{1:K}) = -\mathbb{I}(x_{1:K}; y_{1:K}) + \mathbb{H}(x_{1:K}),$$

where $\mathbb{I}(x_{1:K}; y_{1:K})$ *is the mutual information of* $x_{1:K}$ *and* $y_{1:K}$*, and* $\mathbb{H}(x_{1:K})$ *is constant.*

Proposition 7. *Consider the Gaussian process* (7.3) *and suppose that the measurement noise in* (7.5) *is Gaussian,* $v_{i,k} \sim \mathcal{N}(0, \Sigma(v_{i,k}))$*.* $\mathbb{H}(x_{1:K}|y_{1:K})$ *is a constant factor away from* $\log\det(\Sigma(x^\star_{1:K}))$*, where* $\Sigma(x^\star_{1:K})$ *is the error covariance of the minimum mean square estimator* $x^\star_{1:K}$ *of* $x_{1:K}$ *given the measurements* $y_{1:K}$*. In particular:*[2]

$$\mathbb{H}(x_{1:K}|y_{1:K}) = \frac{\log\det(\Sigma(x^\star_{1:K}))}{2} + \frac{nK\log(2\pi e)}{2}.$$

7.3. Main Results

We first prove that Problem 1 is NP-hard, and then derive for it a provably near-optimal approximation algorithm:

Theorem 14. *The problem of sensor scheduling in stochastic processes with nonlinear observations (Problem 1) is NP hard.*

Due to Theorem 14, we need to appeal to approximation algorithms to obtain a solution to Problem 1 in polynomial-time. To this end, we propose an efficient near-optimal algorithm (Algorithm 13 with a subroutine in Algorithm 14) and quantify its performance and time

[2]We explain $x^\star_{1:K}$ and $\log\det(\Sigma(x^\star_{1:K}))$: $x^\star_{1:K}$ is the optimal estimator for $x_{1:K}$, since it minimizes among *all* estimators of $x_{1:K}$ the mean square error $\mathbb{E}(\|x_{1:K} - x^\star_{1:K}\|_2^2)$ ($\|\cdot\|_2$ is the euclidean norm), where the expectation is taken with respect to $y_{1:K}$ [123, Appendix E]. $\log\det(\Sigma(x^\star_{1:K}))$ is an estimation error metric related to $\|x_{1:K} - x^\star_{1:K}\|_2^2$, since when it is minimized, the probability that the estimation error $\|x_{1:K} - x^\star_{1:K}\|_2^2$ is small is maximized [185].

Algorithm 13 Approximation algorithm for Problem 1.

Input: Horizon K, scheduling constraints $s_1, s_2, \ldots, s_K$, error metric $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}) : \mathcal{S}_k \subseteq [m], k \in [K] \mapsto \mathbb{R}$

Output: Sensor sets $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_K)$ that approximate the solution to Problem 1, as quantified in Theorem 15

$k \leftarrow 1$, $\mathcal{S}_{1:0} \leftarrow \emptyset$

while $k \leq K$ **do**

1. Apply Algorithm 14 to

$$\min_{S \subseteq [m]} \{\mathbb{H}(x_{1:K}|\mathcal{S}_{1:k-1}, \mathcal{S}) : |\mathcal{S}| \leq s_k\} \tag{7.7}$$

2. Denote by $\mathcal{S}_k$ the solution Algorithm 14 returns
3. $\mathcal{S}_{1:k} \leftarrow (\mathcal{S}_{1:k-1}, \mathcal{S}_k)$
4. $k \leftarrow k + 1$

end while

complexity in the following theorem.

Theorem 15. *The theorem has two parts:*

1. Approximation performance of Algorithm 13: *Algorithm 13 returns sensors sets* $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_K$ *that:*

   (a) *satisfy all the feasibility constraints of Problem 1:* $|\mathcal{S}_k| \leq s_k, k \in [K]$

   (b) *achieve an error* $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K})$ *such that:*

$$\frac{\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}) - OPT}{MAX - OPT} \leq \frac{1}{2}, \tag{7.8}$$

   where OPT *is the optimal cost of Problem 1, and* $MAX \equiv \max_{\mathcal{S}'_{1:K}} \mathbb{H}(x_{1:K}|\mathcal{S}'_{1:K})$ *is the maximum (worst) cost in Problem 1.*

2. Time complexity of Algorithm 13: *Algorithm 13 has time complexity* $O(\sum_{k=1}^{K} s_k^2 T)$*, where* T *is the time complexity of evaluating* $\mathbb{H}(x_{1:K}|\mathcal{S}'_{1:K}) : \mathcal{S}'_k \subseteq [m]$, $k \in [K] \mapsto \mathbb{R}$ *at an* $\mathcal{S}'_{1:K}$.

In the following paragraphs, we discuss Algorithm 13's approximation quality and time complexity and fully characterize the latter in Theorem 16 and Corollary 7 for Gaussian processes and Gaussian measurement noise.

Supermodularity and monotonicity of $\mathbb{H}(x_{1:K}|y_{1:K})$. We state $\mathbb{H}(x_{1:K}|y_{1:K})$'s properties that are used to prove Theorem 15. In particular, we show that $\mathbb{H}(x_{1:K}|y_{1:K})$ is a non-increasing and supermodular function with respect to the sequence of selected sensors. Then, Theorem 15 follows by combining these two results with results on submodular functions maximization over matroid constraints [12].

Approximation quality of Algorithm 13. Theorem 15 quantifies the worst-case performance of Algorithm 13 across all values of Problem 1's parameters. The reason is that the right-hand side of (7.8) is constant. In particular, (7.8) guarantees that for any instance of Problem 1, the distance of the approximate cost $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K})$ from OPT is at most $1/2$ the distance of the worst (maximum) cost MAX from OPT. This approximation factor is close to the optimal approximation factor $1/e \cong .38$ one can achieve in the worst-case for Problem 1 in polynomial time [141]; the reason is twofold: first, Problem 1 involves the minimization of a non-increasing and supermodular function [96], and second, as we proved in Theorem 14, Problem 1 is in the worst-case equivalent to the minimal observability problem introduced in [7], which cannot be approximated in polynomial time with a better factor than the $1/e$ [13].

Remark 9. *We can improve the* $1/2$ *approximation factor of Algorithm 13 to* $1/e$ *by utilizing the algorithm introduced in [190]. However, this algorithm has time complexity* $O((nK)^{11}T)$*, where* T *is the time complexity of evaluating* $\mathbb{H}(x_{1:K}|\mathcal{S}'_{1:K}) : \mathcal{S}'_k \subseteq [m],\ k \in [K] \mapsto \mathbb{R}$ *at an* $\mathcal{S}'_{1:K}$*.*

Time complexity of Algorithm 13. Algorithm 13's time complexity is broken down into two parts: a) the number of evaluations of $\mathbb{H}(x_{1:K}|y_{1:K})$ required by the algorithm; b) the time complexity of each such evaluation. In more detail:

Number of evaluations of $\mathbb{H}(x_{1:K}|y_{1:K})$ required by Algorithm 13. Algorithm 13 requires at most s_k^2 evaluations of $\mathbb{H}(x_{1:K}|y_{1:K})$ at each $k \in [K]$. Therefore, Algorithm 13 achieves a time complexity that is only linear in K with respect to the number of evaluations of $\mathbb{H}(x_{1:K}|y_{1:K})$; the reason is that $\sum_{k=1}^{K} s_k^2 \leq \max_{k\in[K]}(s_k^2)K$. This is in contrast to the algorithm in Remark 9, that obtains the best approximation factor $1/e$, whose time complexity is of the order $O((nK)^{11})$ with respect to the number of evaluations of $\mathbb{H}(x_{1:K}|y_{1:K})$.[3]

Time complexity of each evaluation of $\mathbb{H}(x_{1:K}|y_{1:K})$. This time complexity depends on the properties of both the stochastic process (7.1) (similarly, (7.2)) and the measurement noise $v_{i,k}$ in (7.5). For the case of Gaussian stochastic processes and measurement noises:

Theorem 16. *Consider the Gaussian process model* (7.3) *and suppose that the measurement noise is Gaussian:* $v_{i,k} \sim \mathcal{N}(0, \Sigma(v_{i,k}))$ *such that* $\Sigma(v_{i,k}) \succ 0$*. The time complexity of evaluating* $\mathbb{H}(x_{1:K}|y_{1:K})$ *depends on the sparsity pattern of* $\Sigma(x_{1:K})$ *and* $\Sigma(x_{1:K})^{-1}$ *as follows.*

- *Each evaluation of* $\mathbb{H}(x_{1:K}|y_{1:K})$ *has time complexity* $O(n^{2.4}K)$*, when* either $\Sigma(x_{1:K})$ or $\Sigma(x_{1:K})^{-1}$ *is exactly sparse (that is, block tri-diagonal).*
- *Each evaluation of* $\mathbb{H}(x_{1:K}|y_{1:K})$ *has time complexity* $O(n^{2.4}K^{2.4})$*, when* both $\Sigma(x_{1:K})$ and $\Sigma(x_{1:K})^{-1}$ *are dense.*

Theorem 16 implies that when $\Sigma(x_{1:K})$ or $\Sigma(x_{1:K})^{-1}$ is exactly sparse, the time complexity of each evaluation of $\mathbb{H}(x_{1:K}|y_{1:K})$ is only linear in K. This is important because $\Sigma(x_{1:K})$ or $\Sigma(x_{1:K})^{-1}$ is exactly sparse for several applications and system models [191]. For example, in adversarial target tracking applications, where the target wants to avoid capture and

[3] We can also speed up Algorithm 13 by implementing in Algorithm 14 the method of lazy evaluations [99]: this method avoids in Step 2 of Algorithm 14 the computation of $\rho_i(\mathcal{S}^{t-1})$ for unnecessary choices of i.

Algorithm 14 Single step greedy algorithm (subroutine in Algorithm 13).

Input: Current iteration k, selected sensor sets $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{k-1})$ up to the current iteration, constraint s_k, error metric $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}) : \mathcal{S}_k \subseteq [m], k \in [K] \mapsto \mathbb{R}$

Output: Sensor set $\mathcal{S}_k$ that approximates the solution to Problem 1 at time k

$\mathcal{S}^0 \leftarrow \emptyset$, $\mathcal{X}^0 \leftarrow [m]$, and $t \leftarrow 1$

Iteration t:

1. If $\mathcal{X}^{t-1} = \emptyset$, **return** $\mathcal{S}^{t-1}$
2. Select $i(t) \in \mathcal{X}^{t-1}$ for which $\rho_{i(t)}(\mathcal{S}^{t-1}) = \max_{i \in \mathcal{X}^{t-1}} \rho_i(\mathcal{S}^{t-1})$, with ties settled arbitrarily, where:

$$\rho_i(\mathcal{S}^{t-1}) \equiv \mathbb{H}(x_{1:K}|\mathcal{S}_{1:k-1}, \mathcal{S}^{t-1}) - \mathbb{H}(x_{1:K}|\mathcal{S}_{1:k-1}, \mathcal{S}^{t-1} \cup \{i\})$$

3.a. If $|\mathcal{S}^{t-1} \cup \{i(t)\}| > s_k$, $\mathcal{X}^{t-1} \leftarrow \mathcal{X}^{t-1} \setminus \{i(t)\}$, and go to Step 1

3.b. If $|\mathcal{S}^{t-1} \cup \{i(t)\}| \leq s_k$, $\mathcal{S}^t \leftarrow \mathcal{S}^{t-1} \cup \{i(t)\}$ and $\mathcal{X}^t \leftarrow \mathcal{X}^{t-1} \setminus \{i(t)\}$

4. $t \leftarrow t + 1$ and continue

randomizes its motion in the environment (by un-correlating its movements), $\Sigma(x_{1:K})$ can be considered tri-diagonal (since this implies $x(t_k)$ and $x(t_{k'})$ are uncorrelated for $|k - k'| > 2$). Similarly, in SLAM, or in system models where the Gaussian process in (7.3) is generated by a linear or nonlinear system corrupted with Gaussian noise, $\Sigma(x_{1:K})^{-1}$ is block tri-diagonal [133]. In particular, for linear systems, $\Sigma(x_{1:K})^{-1}$ is block tri-diagonal [133, Section 3.1], and for nonlinear systems, $\Sigma(x_{1:K})^{-1}$ is efficiently approximated by a block tri-diagonal matrix as follows: for each k, before the k-th iteration of Step 1 in Algorithm 13, we first compute $\tilde{\mu}_{1:K}$ given $y_{1:(k-1)}$ up to k. This step has complexity $O(n^{2.4}K)$ when $\Sigma(x_{1:K})^{-1}$ is sparse [133, Eq. (5)] [192, Section 3.8], and it does not increase the total time complexity of Algorithm 13. Then, we continue as in [133, Section 3.2].

Sparsity in $\mathbb{H}(x_{1:K}|y_{1:K})$. We state the two properties of $\mathbb{H}(x_{1:K}|y_{1:K})$ that result to Theorem 16. In particular, we prove that $\mathbb{H}(x_{1:K}|y_{1:K})$ is expressed in closed form with two different formulas such that the time complexity for the evaluation of $\mathbb{H}(x_{1:K}|y_{1:K})$ using the first formula is decided by the sparsity pattern of $\Sigma(x_{1:K})$, whereas using the second formula is decided by the sparsity pattern of $\Sigma(x_{1:K})^{-1}$. The reason for this dependence is that the rest of the matrices in these formulas are sparser than $\Sigma(x_{1:K})$ or $\Sigma(x_{1:K})^{-1}$; in particular, they are block diagonal.

The full characterization of Algorithm 13's time complexity for Gaussian processes and Gaussian measurement noises follows.

Corollary 7. *Consider the Gaussian process model* (7.3) *and suppose that the measurement noise is Gaussian:* $v_{i,k} \sim \mathcal{N}(0, \Sigma(v_{i,k}))$ *such that* $\Sigma(v_{i,k}) \succ 0$. *The time complexity of Algorithm 13 depends on the sparsity pattern of* $\Sigma(x_{1:K})$ *and* $\Sigma(x_{1:K})^{-1}$ *as follows.*

- *Algorithm 13 has time complexity* $O(n^{2.4}K\sum_{k=1}^{K} s_k^2)$, *when* either $\Sigma(x_{1:K})$ or $\Sigma(x_{1:K})^{-1}$ *is exactly sparse (that is, block tri-diagonal).*

- *Algorithm 13 has time complexity* $O(n^{2.4}K^{2.4}\sum_{k=1}^{K} s_k^2)$*, when* both $\Sigma(x_{1:K})$, $\Sigma(x_{1:K})^{-1}$ *are dense.*

Comparison of Algorithm 13's time complexity for Gaussian processes and Gaussian measurement noises, per Corollary 7, to that of existing scheduling algorithms. The most relevant algorithm to Algorithm 13 is the one provided in [185], where linear systems with additive process noise and measurement noises with *any* distribution are assumed. Algorithm 13 generalizes [185] from linear systems and measurements to Gaussian processes and nonlinear measurements. At the same time, it achieves the same time complexity as the algorithm in [185] when $\Sigma(x_{1:K})$ or $\Sigma(x_{1:K})^{-1}$ is exactly sparse. This is important since the algorithm in [185] has time complexity lower than the-state-of-the-art batch estimation sensor scheduling algorithms, such as the algorithm proposed in [135], and similar to that of the state of the art Kalman filter scheduling algorithms, such as those proposed in [5, 119, 132] (in particular, lower for large K).

7.4. Conclusion Remarks & Future Work

In this chapter, we proposed Algorithm 13 for the NP-hard problem of sensor scheduling for stochastic process estimation. Exploiting the supermodularity and monotonicity of conditional entropy, we proved that the algorithm has an approximation factor 1/2 and linear complexity in the scheduling horizon. It achieves both the accuracy of batch estimation scheduling algorithms and, surprisingly, when the information structure of the problem is sparse, the low time complexity of Kalman filter scheduling algorithms for linear systems. This is the case, for example, in applications such as SLAM and target tracking, and for processes generated by linear or nonlinear systems corrupted with Gaussian noise. Future work will focus on an event-triggered version of the scheduling problem, in which the measurement times are decided online based on the available measurements, and on a decentralized version, in which information is exchanged only among neighboring sensors.

7.5. Appendix: Proof of Results

7.5.1. Proof of Proposition 7

Proof: We first show that the conditional probability distribution of $x_{1:K}$ given $y_{1:K}$ is Gaussian with covariance $\Sigma(x^\star_{1:K})$, and then apply the following lemma:
Lemma 5 (Ref. [181]). *Let* $x \sim \mathcal{N}(\mu, \Sigma)$ *and* $x \in \mathbb{R}^m$*:*

$$\mathbb{H}(x) = \frac{1}{2}\log[(2\pi e)^m \det(\Sigma)].$$

Specifically, due to Assumption 7.6, $(x_{1:K}, y_{1:K})$ are jointly Gaussian. This has a twofold implication: first, the minimum mean square estimator of $x_{1:K}$ given $y_{1:K}$ is linear in $y_{1:K}$ [123, Proposition E.2]; second, the conditional probability distribution of $x_{1:K}$ given $y_{1:K}$ is Gaussian [124], with covariance $\Sigma(x^\star_{1:K})$. Therefore, due to [123, Proposition E.3], this is also the covariance of the minimum mean square estimator of $x_{1:K}$ given $y_{1:K}$. As a result, due to

Lemma 5:

$$\begin{aligned}\mathbb{H}(x_{1:K}|y_{1:K}) &= \mathbb{E}_{y_{1:K}=y'_{1:K}}\left(\mathbb{H}(x_{1:K}|y_{1:K}=y'_{1:K})\right)\\ &= \mathbb{E}_{y_{1:K}=y'_{1:K}}\left(\frac{1}{2}\log[(2\pi e)^{nK}\det(\Sigma(x^{\star}_{1:K}))\right)\\ &= \frac{nK\log(2\pi e)+\log\det(\Sigma(x^{\star}_{1:K}))}{2}. \end{aligned} \tag{7.9}$$

We derive a formula for $\Sigma(x^{\star}_{1:K})$ in the proof of Lemma 8. ■

Proof of Theorem 15

Proof: We first prove that $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K})$ is a non-increasing and supermodular function in the choice of the sensors. Then, we prove Theorem 15 by combining these two results and results on the maximization of submodular functions over matroid constraints [12].

Notation. Given K disjoint finite sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_K$ and $A_i, B_i \in \mathcal{E}_i$, we write $A_{1:K} \preceq B_{1:K}$ to denote that for all $i \in [K]$, $A_i \subseteq B_i$ (A_i is a subset of B_i). Moreover, we denote that $A_i \in \mathcal{E}_i$ for all $i \in [K]$ by $A_{1:K} \in \mathcal{E}_{1:K}$. In addition, given $A_{1:K}, B_{1:K} \in \mathcal{E}_{1:K}$, we write $A_{1:K} \uplus B_{1:K}$ to denote that for all $i \in [K]$, $A_i \cup B_i$ (A_i union B_i).
Definition 22. *Consider K disjoint finite sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_K$. A function $h : \mathcal{E}_{1:K} \mapsto \mathbb{R}$ is* non-decreasing *if and only if for all $A, B \in \mathcal{E}_{1:K}$ such that $A \preceq B$, $h(A) \leq h(B)$; $h : \mathcal{E}_{1:K} \mapsto \mathbb{R}$ is* non-increasing *if $-h$ is non-decreasing.*
Proposition 8. *For any finite $K \in \mathbb{N}$, consider K distinct copies of $[m]$, denoted by $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_K$. The estimation error metric $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}) : \mathcal{R}_{1:K} \mapsto \mathbb{R}$ is a non-increasing function in the choice of the sensors $\mathcal{S}_{1:K}$.*

Proof Consider $A, B \in \mathcal{R}_{1:K}$ such that $A \preceq B$, and denote by $B \setminus A \equiv \{i | i \in B, i \notin A\}$: $\mathbb{H}(x_{1:K}|B) = \mathbb{H}(x_{1:K}|A, B \setminus A) \leq \mathbb{H}(x_{1:K}|A)$ since conditioning can either keep constant or decrease the entropy [181]. ■
Definition 23. *Consider K disjoint finite sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_K$. A function $h : \mathcal{E}_{1:K} \mapsto \mathbb{R}$ is* submodular *if and only if for all $A, B, C \in \mathcal{E}_{1:K}$ such that $A \preceq B$, $h(A \uplus C) - h(A) \geq h(B \uplus C) - h(B)$; $h : \mathcal{E}_{1:K} \mapsto \mathbb{R}$ is* supermodular *if $-h$ is submodular.*
Proposition 9. *For any finite $K \in \mathbb{N}$, consider K distinct copies of $[m]$, denoted by $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_K$; the estimation error metric $\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}) : \mathcal{R}_{1:K} \mapsto \mathbb{R}$ is a set supermodular function in the choice of the sensors $\mathcal{S}_{1:K}$.*

Proof: Let $A, B, C \in \mathcal{E}_{1:K}$ such that $A \preceq B$:

$$\begin{aligned}
\mathbb{H}(x_{1:K}|A) - \mathbb{H}(x_{1:K}|A \uplus C) & \qquad (7.10)\\
= \mathbb{H}(x_{1:K}|A) - \mathbb{H}(x_{1:K}|A, C) & \\
= \mathbb{I}(x_{1:K}; C|A) & \qquad (7.11)\\
= \mathbb{H}(C|A) - \mathbb{H}(C|x_{1:K}, A) & \qquad (7.12)\\
\geq \mathbb{H}(C|B) - \mathbb{H}(C|x_{1:K}, B) & \qquad (7.13)\\
= \mathbb{I}(x_{1:K}; C|B) & \qquad (7.14)\\
= \mathbb{H}(x_{1:K}|B) - \mathbb{H}(x_{1:K}|B, C) & \qquad (7.15)\\
= \mathbb{H}(x_{1:K}|B) - \mathbb{H}(x_{1:K}|B \uplus C). & \qquad (7.16)
\end{aligned}$$

Eq. (7.10) and (7.16) follow from our definition of $\uplus$. (7.11) and (7.12), (7.13) and (7.14), and (7.14) and (7.15) hold due to the definition of mutual information [181]. (7.13) follows from (7.12) due to two reasons: first, $\mathbb{H}(C|A) \geq \mathbb{H}(C|B)$, since $A \preceq B$ and conditioning can either keep constant or decrease the entropy [181]; second, $\mathbb{H}(C|x_{1:K}, A) = \mathbb{H}(C|x_{1:K}, B)$ due to the independence of the measurements given $x_{1:K}$, per Assumption 7.6. ■

Proof of Part 1 of Theorem 15: We use the next result from the literature of maximization of submodular functions over matroid constraints:

Definition 24. *Consider a finite set $\mathcal{E}$ and a collection $\mathcal{C}$ of subsets of $\mathcal{E}$. $(\mathcal{E}, \mathcal{C})$ is:*

- *an* independent system *if and only if:*
  - *$\emptyset \in \mathcal{C}$, where $\emptyset$ denotes the empty set*
  - *for all $X' \subseteq X \subseteq \mathcal{E}$, if $X \in \mathcal{C}$, $X' \in \mathcal{C}$.*
- *a* matroid *if and only if in addition to the previous two properties:*
  - *for all $X', X \in \mathcal{C}$ where $|X'| < |X|$, there exists $x \notin X'$ and $x \in X$ such that $X' \cup \{x\} \in \mathcal{C}$.*

Lemma 6 (Ref. [12])**.** *Consider K independence systems $\{(\mathcal{E}_k, \mathcal{C}_k)\}_{k\in[K]}$, each the intersection of at most P matroids, and a submodular and non-decreasing function $h : \mathcal{E}_{1:K} \mapsto \mathbb{R}$. There exist a polynomial time greedy algorithm that returns an (approximate) solution $\mathcal{S}_{1:K}$ to:*

$$\begin{aligned}
&\underset{\mathcal{S}_{1:K} \preceq \mathcal{E}_{1:K}}{maximize} && h(\mathcal{S}_{1:K})\\
&subject\ to && \mathcal{S}_k \cap \mathcal{E}_k \in \mathcal{C}_k, k \in [K],
\end{aligned} \qquad (7.17)$$

that satisfies:

$$\frac{h(\mathcal{O}) - h(\mathcal{S}_{1:K})}{h(\mathcal{O}) - h(\emptyset)} \leq \frac{P}{1+P}, \qquad (7.18)$$

where $\mathcal{O}$ is an (optimal) solution to (7.17)*.*

Lemma 7. *Problem 1 is an instance of* (7.17) *with $P = 1$.*

Proof: We identify the instance of $\{\mathcal{E}_k, \mathcal{C}_k\}_{k\in[K]}$ and h, respectively, that translate (7.17) to

Problem 1:

Given K distinct copies of $[m]$, denoted by $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_K$, first consider $\mathcal{E}_k = \mathcal{R}_k$ and $\mathcal{C}_k = \{\mathcal{S} | \mathcal{S} \subseteq \mathcal{R}_k, |\mathcal{S}| \leq s_k\}$: $(\mathcal{E}_k, \mathcal{C}_k)$ satisfies the first two points in part 1 of Definition 24, and as a result is an independent system. Moreover, by its definition, $\mathcal{S}_k \cap \mathcal{E}_k \in \mathcal{C}_k$ if and only if $|\mathcal{S}_k| \leq s_k$.

Second, for all $\mathcal{S}_{1:K} \preceq \mathcal{E}_{1:K}$, consider:

$$h(\mathcal{S}_{1:K}) = -\mathbb{H}(x_{1:K}|\mathcal{S}_{1:K}).$$

From Propositions 8 and 9, $h(\mathcal{S}_{1:K})$ is set submodular and non-decreasing. In addition to Lemma 7, the independence system $(\mathcal{E}_k, \mathcal{C}_k)$, where $\mathcal{E}_k = \mathcal{R}_k$ and $\mathcal{C}_k = \{\mathcal{S} | \mathcal{S} \subseteq \mathcal{R}_k, |\mathcal{S}| \leq s_k\}$, satisfies also the point in part 2 of Definition 24; thereby, it is also a matroid and as a result P, as in Lemma 6, is 1. ■ This observation, along with Lemmas 6 and 7 complete the proof of (5.6), since the adaptation to Problem 1 of the greedy algorithm in [12, Theorem 4.1] results to Algorithm 13. ■

Proof of Part 2 of Theorem 15: Algorithm 13 requires for each $k \in [K]$ the application of Algorithm 14 to (7.8). In addition, each such application requires at most s_k^2 evaluations of $\mathbb{H}(x_{1:K}|y_{1:K})$. Therefore, Algorithm 13 has time complexity $O(\sum_{k=1}^{K} s_k^2 T)$. ■ The proof of Theorem 15 is complete. ■

Proof of Theorem 16

Notations. We introduce four notations: first, $S_{1:K}$ is the block diagonal matrix with diagonal elements the sensor selection matrices $S_1, S_2, \ldots, S_K$; second, $C(x_{1:K})$ is the block diagonal matrix with diagonal elements the matrices $S_1 C_1, S_2 C_2, \ldots, S_K C_K$, where $C_k \equiv G(x_k)$ and $G(x(t)) \equiv \partial g(x(t))/\partial x(t)$; third, v_k is the batch measurement noise vector $(v_{1,k}^\top, v_{2,k}^\top, \ldots, v_{m,k}^\top)^\top$; and fourth, $\mu_{1:K} \equiv (\mu(t_1)^\top, \mu(t_2)^\top, \ldots, \mu(t_K)^\top)^\top$.

Proof: We first derive the two formulas for $\mathbb{H}(x_{1:K}|y_{1:K})$: the first formula is expressed in terms of $\Sigma(x_{1:K})^{-1}$, and the second formula is expressed in terms of $\Sigma(x_{1:K})$.

Lemma 8 (Formula of $\mathbb{H}(x_{1:K}|y_{1:K})$ in terms of $\Sigma(x_{1:K})^{-1}$)**.** *Consider the start of the k-th iteration in Algorithm 13. Given the measurements $y_{1:(k-1)}$ up to k, $\mathbb{H}(x_{1:K}|y_{1:K})$ is given by $-T_1' + nK\log(2\pi e)/2$, where:*

$$T_1' \equiv \frac{1}{2}\log\det(\Xi + \Sigma(x_{1:K})^{-1})$$
$$\Xi \equiv C(\tilde{\mu}_{1:K})^\top S_{1:K} \Sigma(v_{1:K})^{-1} S_{1:K}^\top C(\tilde{\mu}_{1:K})$$

and $\tilde{\mu}_{1:K}$ is the maximum a posteriori (MAP) estimate of $x_{1:K}$ given the measurements $y_{1:(k-1)}$ up to k.

Lemma 9 (Formula of $\mathbb{H}(x_{1:K}|y_{1:K})$ in terms of $\Sigma(x_{1:K})$)**.** *Consider the start of the k-th iteration in Algorithm 13. Given the measurements $y_{1:(k-1)}$ up to k, $\mathbb{H}(x_{1:K}|y_{1:K})$ is given*

by $\mathbb{H}(x_{1:K}|y_{1:K}) = T_1 - T_2 + \mathbb{H}(x_{1:K})$*, where:*

$$T_1 \equiv \frac{1}{2}\sum_{k=1}^{K} \log[(2\pi e)^{s_k} \det(S_k \Sigma(v_k) S_k^\top)] \tag{7.19}$$

$$T_2 \equiv \frac{1}{2}\log[(2\pi e)^{\sum_{k=1}^{K} s_k} \det(\Sigma(y_{1:K}))] \tag{7.20}$$

$$\Sigma(y_{1:K}) = S_{1:K}\Sigma(v_{1:K})S_{1:K}^\top + C(\tilde{\mu}_{1:K})\Sigma(x_{1:K})C(\tilde{\mu}_{1:K})^\top,$$

and $\tilde{\mu}_{1:K}$ *is the maximum a posteriori (MAP) estimate of* $x_{1:K}$ *given the measurements* $y_{1:(k-1)}$ *up to* k*.*

We complete the proof for each case of Theorem 16:

- *Time complexity of each evaluation of* $\mathbb{H}(x_{1:K}|y_{1:K})$ *when either* $\Sigma(x_{1:K})$ *or* $\Sigma(x_{1:K})^{-1}$ *is exactly sparse (that is, block tri-diagonal)*: We present the proof only for the case where $\Sigma(x_{1:K})^{-1}$ is exactly sparse since the proof for the case where $\Sigma(x_{1:K})$ is exactly sparse is similar. In particular, consider the formula of $\mathbb{H}(x_{1:K}|y_{1:K})$ in Lemma 8: T_1' involves the log determinant of a matrix that is the sum of two $nK \times nK$ sparse matrices: the first matrix is block diagonal, and the second one is block tri-diagonal. The block diagonal matrix is evaluated in $O(n^{2.4}K)$ time, since the determinant of an $n \times n$ matrix is computed in $O(n^{2.4})$ time using the Coppersmith-Winograd algorithm [97]. Then, T_1' is evaluated in $O(n^{2.4}K)$ [142, Theorem 2].

- *Time complexity of each evaluation of* $\mathbb{H}(x_{1:K}|y_{1:K})$ *when both* $\Sigma(x_{1:K})$ *and* $\Sigma(x_{1:K})^{-1}$ *are dense*: In this case, T_1' (and similarly T_2 in Lemma 9) is the log determinant of a dense $nK \times nK$ matrix. Therefore, it is evaluated in $O((nK)^{2.4})$ time, since the determinant of an $n \times n$ matrix is computed in $O(n^{2.4})$ time using the Coppersmith-Winograd algorithm [97].

■

CHAPTER 8 : LQG Control and Sensing Co-design

Linear-Quadratic-Gaussian (LQG) control is concerned with the design of an optimal controller and estimator for linear Gaussian systems with imperfect state information. Standard LQG control assumes the set of sensor measurements to be fed to the estimator to be given. However, in many problems arising in networked systems and robotics, one may be interested in *designing* a suitable set of sensors for LQG control. In this chapter, we introduce the *LQG control and sensing co-design* problem, where one has to jointly design a suitable sensing, estimation, and control policy. In particular, we consider two dual instances of the co-design problem: the *sensing-constrained LQG control* problem, where the design maximizes the control performance subject to sensing constraints, and the *minimum-sensing LQG control*, where the design minimizes the amount of sensing subject to performance constraints. We focus on the realistic case in which the sensing design has to be selected among a finite set of possible sensing modalities, where each modality is associated with a (possibly) different cost. While we observe that the computation of the optimal sensing design is intractable in general, we present the first scalable LQG co-design algorithms to compute near-optimal policies with provable sub-optimality guarantees. To this end, (i) we show that a separation principle holds, which partially decouples the design of sensing, estimation, and control; (ii) we frame LQG co-design as the optimization of (approximately) supermodular set functions; (iii) we develop novel algorithms to solve the resulting optimization problems; (iv) we prove original results on the performance of these algorithms and establish connections between their suboptimality gap and control-theoretic quantities. We conclude the chapter by discussing two practical applications of the co-design problem, namely, *sensing-constrained formation control* and *resource-constrained robot navigation*.[1]

8.1. Introduction

Traditional approaches to the control of dynamical systems with partially observable state assume the choice of sensors used to observe the system to be given [123]. The choice of sensors usually results from a preliminary design phase in which an expert designer selects a suitable sensor suite that accommodates estimation requirements (e.g., observability, desired estimation error) and system constraints (e.g., size, cost). However, modern control applications, from large networked systems to miniaturized robotics systems, pose serious limitations to the applicability of this traditional paradigm: in particular, in large-scale networked systems (e.g., smart grids, or robot swarms), in which new nodes are continuously added and removed from the network, a manual re-design of the sensors becomes cumbersome and expensive, and it is not scalable; in miniaturized robot systems, while the set of onboard sensors is fixed, it may be desirable to selectively activate only a subset of the sensors during different phases of operation, to minimize power consumption. Overall, in both applications, although a designer has access to a (possibly large) list of potential sensors, due to resource constraints (size, weight, power, cost) the designer can utilize only a subset of them. Thus, the need for online and large-scale sensor selection demands for automated approaches that efficiently select a subset of sensors to maximize system performance.

Motivated by the aforementioned need, in this chapter we consider the problem of jointly

[1] This chapter is based on the paper by Tzoumas et al. [193].

designing control, estimation, and sensing for systems with partially observable state.

Related work in control theory. Related work in control theory focuses on either the co-design of estimation and control in presence of communication constraints [123, 194, 195, 196, 197, 198, 199], or on the design of the system's sensing and actuation [5, 20, 52, 54, 60, 102, 119, 200, 201, 202, 203, 204] (sensor and actuator selection). In more detail:

LQG control design: The line of work [123, 194, 195, 196, 197, 198, 199] assumes the set of sensors and actuators to be given, and either focuses on the co-design of estimation and control over band-limited communication channels, or investigates the trade-offs between communication constraints (e.g., data rate, quantization, delays) and control performance (e.g., stability) in networked control systems. These works provide results on the impact of quantization [194], finite data rates [195, 196], as well as, separation principles for LQG design with communication constraints [197]. More recent work focuses on privacy constraints [198]. In addition, [199] studies rationally inattentive control laws for LQG control and discusses their effectiveness in stabilizing the system. We refer the reader to the surveys [123, 205, 206, 207] for a comprehensive review on LQG control.

Sensor and actuator selection: The line of work [5, 20, 52, 54, 60, 102, 119, 200, 201, 202, 203, 204] focuses on selecting the system's active sensors and actuators, independently of the control task at hand. In particular, [119] proposes a sensor placement algorithm to maximize the accuracy of maximum likelihood estimation over static parameters, whereas [5, 20, 54, 102, 200] focus on maximizing the estimation accuracy for non-static parameters; [5, 20, 54, 200] present sensor scheduling algorithms for optimal Kalman filtering, while [102] presents sensor scheduling algorithms for optimal batch state estimation (smoothing); [60] considers fixed-lag smoothing and investigates sensor scheduling and feature selection for vision-based agile navigation of autonomous robots. Finally, [52, 201, 202, 203, 204] present sensor and actuator selection algorithms to optimize the average observability and controllability of systems.

Extending the focus of the aforementioned works, more recent work focuses on the co-design of control and estimation, as well as, of sensing [208, 209], by augmenting the standard LQG cost with an information-theoretic regularizer, and by optimizing the sensing capabilities of each of the system's sensors using semi-definite programming. The main difference between [208, 209], and our proposal in this chapter is that in [208, 209] the choice of sensors is arbitrary, rather than being restricted to a finite set of available sensors.

Related work on set function optimization. The algorithms for sensor and actuator selection discussed above employ either convex relaxation techniques [20, 54, 119, 200, 202] or combinatorial optimization techniques [5, 52, 60, 102, 201, 203]. The advantage of the combinatorial optimization techniques is that they lead to algorithms with provable suboptimality guarantees and low running time. The literature on combinatorial optimization, which is more relevant for the discussion in this chapter, includes investigation into (i) submodular optimization subject to cardinality constraints [210]; (ii) submodular optimization subject to heterogeneous-cost constraints [44, 211, 212]; and (iii) approximately submodular optimization subject to cardinality constraints [32]. We note that the related literature does not cover the case of approximately submodular optimization subject to heterogeneous-cost

constraints, which is indeed the setup of interest for our LQG control and sensing co-design problems, hence requiring us to develop novel algorithms and results for this case.

Contributions to control theory. We introduce the *LQG control and sensing co-design* problem, that involves the joint design of sensing, control, and estimation, by extending Linear-Quadratic-Gaussian (LQG) control to the case where, besides designing an optimal controller and estimator, one has to choose a set of sensors to observe the system state. We consider the realistic case in which the choice of sensors, rather than being arbitrary (see, e.g., [208]), is restricted to a finite selection from a set of available sensors. In particular, in our formulation each available sensor is associated with a *cost* that quantifies the penalty incurred when using that sensor (trivially, if there is no cost associated to using a sensor, one would always prefer to select and use all available sensors). In more detail, we consider the general case in which each sensor has a potentially different cost, hence capturing the practical scenarios where each sensor may have a different monetary cost, power consumption, or bandwidth utilization.

We formulate two dual instances of the LQG co-design problem. The first instance, named *sensing-constrained LQG control*, involves the joint design of the controller, estimator, and sensing policies that minimize the LQG objective (quantifying tracking performance and control effort) while satisfying a given constraint on the maximum cost of the selected sensors. The second instance, named *minimum-sensing LQG control*, involves the joint design of the controller, estimator, and sensing that minimizes the cost of the selected sensors while satisfying a given bound on the LQG performance.

We then leverage a separation principle[2] to partially decouple the design of control, estimation, and sensing, and we frame the sensor design subproblem as the optimization of (approximately) supermodular set functions. While the computation of the optimal sensing strategies is combinatorial in nature, we provide the first scalable co-design algorithms, which retrieve a near-optimal choice of sensors, as well as the corresponding control and estimation policies. We show that the suboptimality gaps of these algorithms depend on the supermodularity ratio γ_f of the set function f appearing in our problem, and we establish connections between the supermodularity ratio γ_f and control-theoretic quantities, providing also a computable lower bound for γ_f.

Contributions to set function optimization. In proving the aforementioned results, we extend the literature on supermodular optimization. In particular, (i) we provide the first efficient algorithms for the optimization of approximately supermodular functions subject to heterogeneous-cost constraints; and (ii) we improve known suboptimality bounds that also apply to the optimization of (exactly) supermodular functions: specifically, the proposed algorithm for approximate supermodular optimization with heterogeneous-cost constraints can achieve in the exactly supermodular case the approximation bound $(1 - 1/e)$, which is superior to the previously established bound $1/2(1 - 1/e)$ in the literature [44].

Application examples. We motivate the importance of the LQG control and sensing co-

[2]The separation principle leverages standard results in LQG control and follows the line of [208], hence we do not claim it to be an original contribution.

design problem, and demonstrate the effectiveness of the proposed algorithms in numerical experiments, by considering two application scenarios, namely, a *sensing-constrained formation control* scenario and a *resource-constrained robot navigation* scenario. In particular, we present a Monte Carlo analysis for both scenarios, which demonstrates that (i) the proposed sensor selection strategy is near-optimal, and in particular, the resulting LQG-cost (tracking performance) matches the optimal selection in all tested instances for which the optimal selection could be computed via a brute-force approach; (ii) a more naive selection which attempts to minimize the state estimation covariance [5] (rather than the LQG cost) has degraded LQG tracking performance, often comparable to a random selection; and (iii) the selection of a small subset of sensors using the proposed algorithms ensures an LQG cost that is close to the one obtained by using all available sensors, hence providing an effective alternative for control under sensing constraints [60].

Comparison with the preliminary results in [213]. This chapter extends our preliminary results [213], and provides a more comprehensive presentation of the LQG co-design problem, including both *sensing-constrained LQG control* (introduced in [213]) and the *minimum-sensing LQG control* problem (not previously published). Moreover, we generalize the original setup in [213] to account for heterogeneous sensor costs (in [213] each sensor has unit cost) and extend the numerical analysis accordingly. Most of the technical results, including Theorems 17–19, Proposition 10, as well as Algorithms 16–18 are novel, and have not been previously published.

Organization of the rest of the chapter. Section 8.2 formulates the LQG control and sensing co-design problems considered in this chapter. Section 8.3 presents a separation principle and provides scalable, near-optimal algorithms for the co-design problems. Section 8.4 characterizes the running time and approximation performance of the proposed algorithms, and establishes connections between their suboptimality bounds and control-theoretic quantities. Section 8.5 presents two practical examples of co-design problems and provides a numerical analysis of the proposed algorithms. Section 8.6 concludes the chapter. All proofs are given in the Appendix.

Notation. Lowercase letters denote vectors and scalars (e.g., v), and uppercase letters denote matrices (e.g., M). We use calligraphic fonts to denote sets (e.g., $\mathcal{S}$). The identity matrix of size n is denoted with I_n (the dimension index is omitted when it is clear from the context). For a matrix M and a vector v of appropriate dimension, we define $\|v\|_M^2 \triangleq v^\top M v$. For matrices $M_1, M_2, \ldots, M_k$, we let $\operatorname{diag}(M_1, M_2, \ldots, M_k)$ be the block diagonal matrix with diagonal blocks $M_1, M_2, \ldots, M_k$.

8.2. LQG Control and Sensing Co-design: Problem Statement

In this section we formalize the LQGcontrol and sensing co-design problem considered in this chapter. In particular, we present two "dual" statements of the co-design problem: the *sensing-constrained LQGcontrol*, where the design maximizes the control performance subject to sensing constraints, and the *minimum-sensing LQGcontrol*, where the design minimizes sensing subject to performance constraints.

8.2.1. System, sensors, and control policies

We start by introducing the notions of *system*, *sensors*, and *control policies*. These notions are standard, except that only a subset of sensors is actually used to observe the system's state (these are referred to as "active" sensors in Definition 25), and that we associate a cost to each sensor (Definition 26).

System We consider a standard discrete-time (possibly time-varying) linear system with additive Gaussian noise:

$$x_{t+1} = A_t x_t + B_t u_t + w_t, \quad t = 1, 2, \dots, T, \tag{8.1}$$

where $x_t \in \mathbb{R}^n$ represents the state of the system at time t, $u_t \in \mathbb{R}^{m_t}$ represents the control action, w_t represents the process noise, A_t and B_t are matrices of suitable dimensions, and T is a finite horizon. In addition, we consider the system's initial condition x_1 to be a Gaussian random variable with covariance $\Sigma_{1|0}$, and w_t to be a Gaussian random variable with mean zero and covariance W_t, such that w_t is independent of x_1 and $w_{t'}$ for all $t' = 1, 2, \dots, T$, $t' \neq t$.

Sensors We consider the case in which we have a (potentially large) set of available sensors, which can take noisy linear observations of the system's state. In particular, let $\mathcal{V}$ be a set of indices such that each index $i \in \mathcal{V}$ uniquely identifies a sensor that can be used to observe the state of the system. We consider sensors of the form

$$y_{i,t} = C_{i,t} x_t + v_{i,t}, \quad i \in \mathcal{V}, \tag{8.2}$$

where $y_{i,t} \in \mathbb{R}^{p_{i,t}}$ represents the measurement of sensor i at time t, $C_{i,t}$ is a sensing matrix of suitable dimension, and $v_{i,t}$ represents the measurement noise of sensor i. We assume $v_{i,t}$ to be a Gaussian random variable with mean zero and positive definite covariance $V_{i,t}$, such that $v_{i,t}$ is independent of x_1, and of $w_{t'}$ for any $t' \neq t$, and independent of $v_{i',t'}$ for all $t' \neq t$, and any $i' \in \mathcal{V}$, $i' \neq i$.

In this chapter we are interested in the case in which we cannot use all the available sensors and, as a result, we need to select a convenient subset of sensors in the set $\mathcal{V}$ to meet given specifications on the control performance (formalized in Problem 2 and Problem 3 below).

Definition 25 (Active sensor set and measurement model)**.** *Given a set of available sensors $\mathcal{V}$, we say that $\mathcal{S} \subseteq \mathcal{V}$ is an* active sensor set *if we can observe the measurements from each sensor $i \in \mathcal{S}$ for all $t = 1, 2, \dots, T$. Given an active sensor set $\mathcal{S} = \{i_1, i_2 \dots, i_{|\mathcal{S}|}\}$, we define the following quantities*

$$y_t(\mathcal{S}) \triangleq [y_{i_1,t}^\top, y_{i_2,t}^\top, \dots, y_{i_{|\mathcal{S}|},t}^\top]^\top, \tag{8.3}$$

$$C_t(\mathcal{S}) \triangleq [C_{i_1,t}^\top, C_{i_2,t}^\top, \dots, C_{i_{|\mathcal{S}|},t}^\top]^\top, \tag{8.4}$$

$$V_t(\mathcal{S}) \triangleq \operatorname{diag}\left(V_{i_1,t}, V_{i_2,t}, \dots, V_{i_{|\mathcal{S}|},t}\right), \tag{8.5}$$

which lead to the definition of the measurement model*:*

$$y_t(\mathcal{S}) = C_t(\mathcal{S})x_t + v_t(\mathcal{S}), \tag{8.6}$$

where $v_t(\mathcal{S})$ is a zero-mean Gaussian noise with covariance $V_t(\mathcal{S})$. Despite the availability of a possibly large set of sensors $\mathcal{V}$, our observer will only have access to the measurements produced by the active sensors.

In this chapter we focus on the case where each sensor in the set of available sensors $\mathcal{V}$ is associated with a (possibly different) cost, which captures, for instance, the sensor's monetary cost, its power consumption, or its bandwidth utilization.

Definition 26 (Cost of sensor and cost of active sensor set)**.** *Given a set of available sensors $\mathcal{V}$, we denote the* cost of sensor $i \in \mathcal{V}$ *by $c(i) \geq 0$. Moreover, we denote the* cost of an active sensor set $\mathcal{S} \subseteq \mathcal{V}$ *by $c(\mathcal{S})$ and set it equal to the sum of the sensor costs $c(i)$ for all active sensors $i \in \mathcal{S}$:*

$$c(\mathcal{S}) \triangleq \sum_{i \in \mathcal{S}} c(i). \tag{8.7}$$

The following paragraph formalizes how the choice of the active sensors affects the control policies.

Control policies We consider control policies u_t for all $t = 1, 2, \ldots, T$ that are only informed by the measurements collected by the active sensors:

$$u_t = u_t(\mathcal{S}) = u_t\big(y_1(\mathcal{S}), y_2(\mathcal{S}), \ldots, y_t(\mathcal{S})\big), \quad t = 1, 2, \ldots, T.$$

Such policies are called *admissible*.

8.2.2. LQG co-design problems

The LQGco-design problem considered in this chapter consists in the joint design of sensing, estimation, and control strategies to meet given design specifications. We consider two different types of specifications that lead to two co-design problems, named *sensing-constrained LQGcontrol* (Problem 2) and *minimum-sensing LQGcontrol* (Problem 3).

Problem 2 (Sensing-constrained LQGcontrol)**.** *Given a system, a set of available sensors $\mathcal{V}$, and a sensor budget $b \geq 0$, find a sensor set $\mathcal{S} \subseteq \mathcal{V}$ to be active across all times $t = 1, 2, \ldots, T$, with cost $c(\mathcal{S})$ at most b, and an admissible control policy $u_{1:T}(\mathcal{S}) \triangleq \{u_1(\mathcal{S}), u_2(\mathcal{S}), \ldots, u_T(\mathcal{S})\}$ to minimize the LQGcost function, that is:*

$$\min_{\substack{\mathcal{S} \subseteq \mathcal{V}, \\ u_{1:T}(\mathcal{S})}} \sum_{t=1}^{T} \mathbb{E}\left[\|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2\right], \text{ s.t. } c(\mathcal{S}) \leq b, \tag{8.8}$$

where the state-cost matrices $Q_1, Q_2, \ldots, Q_T$ are positive semi-definite, the control-cost matrices $R_1, R_2, \ldots, R_T$ are positive definite, and the expectation is taken with respect to the initial condition x_1, the process noises $w_1, w_2, \ldots, w_T$, and the measurement noises $v_1(\mathcal{S}), v_2(\mathcal{S}), \ldots, v_T(\mathcal{S})$.

The sensing-constrained LQGcontrol Problem 2 models the practical case in which one cannot use all the available sensors due to power, cost, or bandwidth constraints, and needs to compute a suitable set of active sensors and controls that maximize LQG performance. Note that if the budget constraint is relaxed, all sensors are active (no penalty is incurred in using all sensors) and Problem 2 reduces to standard LQG control.

While Problem 2 imposes constraints on the maximum amount of sensing, the following "dual" problem formulation imposes a constraint on the desired LQGperformance.

Problem 3 (Minimum-sensing LQG control)**.** *Given a system, a set of available sensors $\mathcal{V}$, and an upper bound $\kappa > 0$ for the LQGcost, find a minimum-cost sensor set $\mathcal{S} \subseteq \mathcal{V}$ to be active across all times $t = 1, 2, \ldots, T$ and an admissible control policy $u_{1:T}(\mathcal{S}) \triangleq \{u_1(\mathcal{S}), u_2(\mathcal{S}), \ldots, u_T(\mathcal{S})\}$ such that the LQG cost is at most κ:*

$$\min_{\substack{\mathcal{S} \subseteq \mathcal{V}, \\ u_{1:T}(\mathcal{S})}} \; c(\mathcal{S}), \;\; \text{s.t.} \;\; \sum_{t=1}^{T} \mathbb{E}\left[\|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2\right] \leq \kappa, \tag{8.9}$$

where the state-cost matrices $Q_1, Q_2, \ldots, Q_T$ are positive semi-definite, the control-cost matrices $R_1, R_2, \ldots, R_T$ are positive definite, and the expectation is taken with respect to the initial condition x_1, the process noises $w_1, w_2, \ldots, w_T$, and the measurement noises $v_1(\mathcal{S}), v_2(\mathcal{S}), \ldots, v_T(\mathcal{S})$.

The minimum-sensing LQGcontrol Problem 3 models the practical case in which one wants to design a system that guarantees a desired level of performance, while incurring in the smallest sensing cost (again the cost can be monetary or connected to the use of limited resources).

Remark 10 (Case of uniform-cost sensors)**.** *When all sensors $i \in \mathcal{V}$ have the same cost , say $c(i) = \bar{c} > 0$, the sensor budget constraint can be rewritten as a cardinality constraint, since:*

$$c(\mathcal{S}) \leq b \;\; \Leftrightarrow \;\; \sum_{i \in \mathcal{S}} c(i) \leq b \;\; \Leftrightarrow \;\; |\mathcal{S}|\bar{c} \leq b \;\; \Leftrightarrow \;\; |\mathcal{S}| \leq \frac{b}{\bar{c}}, \tag{8.10}$$

which bounds the cardinality of the set of active sensors. Similarly, under the uniform-cost assumption, the objective of Problem 3 becomes the minimal cardinality objective $|\mathcal{S}|$.

Problem 2 and Problem 3 generalize the imperfect state-information LQGcontrol problem from the case where all sensors in the set of available sensors $\mathcal{V}$ are active, and only optimal control policies are to be found [123, Chapter 5], to the case where only a few sensors in $\mathcal{V}$ can be active, and both optimal sensors and control policies are to be found, jointly.

While we noticed that admissible control policies depend on the active sensor set $\mathcal{S}$, it is worth noticing that this in turn implies that the state evolution will also depend on $\mathcal{S}$, per the system's dynamics eq. (8.1); for this reason we write $x_{t+1}(\mathcal{S})$ in eqs. (8.8) and (8.9). Thereby, the intertwining between control and sensing calls for a joint design strategy and, as a result, in the following section we focus on the design of a jointly optimal control and sensing solution to Problem 2 and Problem 3.

8.3. Co-design Principles and Efficient Algorithms

In this section we first present a separation principle that decouples sensing, estimation, and control, and allows designing them in cascade (Section 8.3.1). We then present scalable algorithms for the sensing and control design in both Problem 2 (Section 8.3.2) and Problem 3 (Section 8.3.3). Theoretical guarantees bounding the suboptimality gap of the proposed algorithms are given in Section 8.4.

8.3.1. Separability of optimal sensing and control design

We characterize the jointly optimal control and sensing solutions for Problem 2 and Problem 3, and prove they can be found in two separate steps, where first the sensing design is computed, and second the control design is found (Theorem 17).

Theorem 17 (Separability of optimal sensing and control design)**.** *For any active sensor set $\mathcal{S} \subseteq \mathcal{V}$, let $\hat{x}_t(\mathcal{S})$ be the Kalman estimator of the state x_t, i.e.,*

$$\hat{x}_t(\mathcal{S}) \triangleq \mathbb{E}[x_t | y_1(\mathcal{S}), y_2(\mathcal{S}), \dots, y_t(\mathcal{S})],$$

and $\Sigma_{t|t}(\mathcal{S})$ be $\hat{x}_t(\mathcal{S})$'s error covariance, i.e., $\Sigma_{t|t}(\mathcal{S}) \triangleq \mathbb{E}[(\hat{x}_t(\mathcal{S}) - x_t)(\hat{x}_t(\mathcal{S}) - x_t)^\top]$ [123, Appendix E]. In addition, let the matrices Θ_t and K_t be the solution of the following backward Riccati recursion

$$\begin{aligned} S_t &= Q_t + N_{t+1}, \\ N_t &= A_t^\top (S_t^{-1} + B_t R_t^{-1} B_t^\top)^{-1} A_t, \\ M_t &= B_t^\top S_t B_t + R_t, \\ K_t &= -M_t^{-1} B_t^\top S_t A_t, \\ \Theta_t &= K_t^\top M_t K_t, \end{aligned} \tag{8.11}$$

with boundary condition $N_{T+1} = 0$ (notably, all matrices in eq. (8.11) are independent of the active sensor set $\mathcal{S}$).

1. (Separability in Problem 2) *Let the sensor set $\mathcal{S}^\star$ and the controllers $u_1^\star, u_2^\star, \dots, u_T^\star$ be a solution to the sensing-constrained LQG Problem 2. Then, $\mathcal{S}^\star$ and $u_1^\star$, $u_2^\star, \dots, u_T^\star$ can be computed in cascade as follows:*

$$\mathcal{S}^\star \in \operatorname*{argmin}_{\mathcal{S} \subseteq \mathcal{V}} \sum_{t=1}^{T} tr[\Theta_t \Sigma_{t|t}(\mathcal{S})], \ \text{ s.t. } \ c(\mathcal{S}) \leq b, \tag{8.12}$$

$$u_t^\star = K_t \hat{x}_t(\mathcal{S}^\star), \quad t = 1, \dots, T. \tag{8.13}$$

2. (Separability in Problem 3) *Let the sensor set $\mathcal{S}^\star$ and the controllers $u_1^\star, u_2^\star, \dots, u_T^\star$ be a solution to the minimum-sensing LQG Problem 3. Moreover, define the constant $\bar{\kappa} \triangleq \kappa - \operatorname{tr}\left(\Sigma_{1|0} N_1\right) - \sum_{t=1}^{T} \operatorname{tr}\left(W_t S_t\right)$. Then, $\mathcal{S}^\star$ and $u_1^\star$, $u_2^\star, \dots, u_T^\star$ can be computed*

Algorithm 15 Joint Sensing and Control design for Problem 2.

Input: Time horizon T, available sensor set $\mathcal{V}$, sensor selection budget b, covariance $\Sigma_{1|0}$ of initial condition x_1; for all $t = 1, 2, \ldots, T$, system matrix A_t, input matrix B_t, process noise covariance W_t, and LQG cost matrices Q_t and R_t; for all sensors $i \in \mathcal{V}$, measurement matrix $C_{i,t}$, measurement noise covariance $V_{i,t}$, and sensor cost $c(i)$.

Output: Active sensors $\widehat{\mathcal{S}}$, and controls $\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_T$.

1: Compute the matrices $\Theta_1, \Theta_2, \ldots, \Theta_T$ using the backward Riccati recursion in eq. (8.11).

2: Return the sensor set $\widehat{\mathcal{S}}$ as the sensor set returned by Algorithm 16, which finds a (possibly approximate) solution to the optimization problem in eq. (9.10);

3: Compute the matrices $K_1, K_2, \ldots, K_T$ using the backward Riccati recursion in eq. (8.11).

4: At time $t = 1, 2, \ldots, T$, compute the Kalman estimate of the state x_t, i.e., the estimate:

$$\hat{x}_t \triangleq \mathbb{E}[x_t | y_1(\widehat{\mathcal{S}}), y_2(\widehat{\mathcal{S}}), \ldots, y_t(\widehat{\mathcal{S}})];$$

5: At time $t = 1, 2, \ldots, T$, return the control $\hat{u}_t = K_t \hat{x}_t$.

in cascade as follows:

$$\mathcal{S}^\star \in \operatorname*{argmin}_{\mathcal{S} \subseteq \mathcal{V}} c(\mathcal{S}), \quad \text{s.t.} \sum_{t=1}^{T} tr[\Theta_t \Sigma_{t|t}(\mathcal{S})] \leq \bar{\kappa}, \tag{8.14}$$

$$u_t^\star = K_t \hat{x}_t(\mathcal{S}^\star), \quad t = 1, \ldots, T. \tag{8.15}$$

Remark 11 (Certainty equivalence principle)**.** *The control gain matrices $K_1, K_2, \ldots, K_T$ are the same as the ones that make the controllers $(K_1 x_1,\ K_1 x_2, \ldots, K_T x_T)$ optimal for the perfect state-information version of Problem 2, where the state x_t is known to the controllers [123, Chapter 4].*

Theorem 17 decouples the design of the sensing from the controller design. In particular, it suggests that once an optimal sensor set $\mathcal{S}^\star$ is found, then the optimal controllers are equal to $K_t \hat{x}_t(\mathcal{S}^\star)$, which correspond to the standard LQG control policy. This should not come as a surprise, since for a given sensing strategy, Problem 2 reduces to standard LQG control. Moreover, for a given sensor set, Problem 3 becomes a feasibility problem and, as a result, admits multiple controls that satisfy the LQG cost bound; one such set of controls are the control actions computed in eq. (8.15), since they minimize the LQG cost and, hence, they also belong to Problem 3's feasible set whenever Problem 3 admits a solution.

We conclude the section with a remark providing an intuitive interpretation of the sensor design steps in eqs. (8.12) and (8.14) for Problem 2 and Problem 3, respectively.

Remark 12 (Control-aware sensor design)**.** *In order to provide insight on the function*

Algorithm 16 Sensing design for Problem 2.

Input: Time horizon T, available sensor set $\mathcal{V}$, sensor selection budget b, covariance $\Sigma_{1|0}$ of initial condition x_1, and for all $t = 1, 2, \ldots, T$ and any sensor $i \in \mathcal{V}$, matrix Θ_t, process noise covariance W_t, measurement matrix $C_{i,t}$, measurement noise covariance $V_{i,t}$, and sensor cost $c(i)$.

Output: Sensor set $\widehat{\mathcal{S}}$.

$\widehat{\mathcal{S}}_1 \leftarrow \arg\min_{i \in \mathcal{V}, c(i) \le b} \sum_{t=1}^{T} \mathrm{tr}[\Theta_t \Sigma_{t|t}(\{i\})]$;
$\widehat{\mathcal{S}}_2 \leftarrow \emptyset$; $\quad \mathcal{V}' \leftarrow \mathcal{V}$;
while $\mathcal{V}' \neq \emptyset$ and $c(\widehat{\mathcal{S}}_2) \le b$ **do**
  for all $a \in \mathcal{V}'$ **do**
    $\widehat{\mathcal{S}}_{2,\alpha} \leftarrow \widehat{\mathcal{S}}_2 \cup \{a\}$; $\quad \Sigma_{1|0}(\widehat{\mathcal{S}}_{2,\alpha}) \leftarrow \Sigma_{1|0}$;
    for all $t = 1, \ldots, T$ **do**
      $\Sigma_{t|t}(\widehat{\mathcal{S}}_{2,\alpha}) \leftarrow$
      $[\Sigma_{t|t-1}(\widehat{\mathcal{S}}_{2,\alpha})^{-1} + C_t(\widehat{\mathcal{S}}_{2,\alpha})^\top V_t(\widehat{\mathcal{S}}_{2,\alpha})^{-1} C_t(\widehat{\mathcal{S}}_{2,\alpha})]^{-1}$;
      $\Sigma_{t+1|t}(\widehat{\mathcal{S}}_{2,\alpha}) \leftarrow A_t \Sigma_{t|t}(\widehat{\mathcal{S}}_{2,\alpha}) A_t^\top + W_t$;
    end for
    $\mathrm{gain}_a \leftarrow \sum_{t=1}^{T} \mathrm{tr}\{\Theta_t [\Sigma_{t|t}(\widehat{\mathcal{S}}_2) - \Sigma_{t|t}(\widehat{\mathcal{S}}_{2,\alpha})]\}$;
  end for
  $s \leftarrow \arg\max_{a \in \mathcal{V}'} [\mathrm{gain}_a / c(a)]$;
  $\widehat{\mathcal{S}}_2 \leftarrow \widehat{\mathcal{S}}_2 \cup \{s\}$;
  $\mathcal{V}' \leftarrow \mathcal{V}' \setminus \{s\}$;
end while
if $c(\widehat{\mathcal{S}}_2) > b$ **then**
  $\widehat{\mathcal{S}}_2 \leftarrow \widehat{\mathcal{S}}_2 \setminus \{s\}$;
end if
$\widehat{\mathcal{S}} \leftarrow \arg\min_{\mathcal{S} \in \{\widehat{\mathcal{S}}_1, \widehat{\mathcal{S}}_2\}} \sum_{t=1}^{T} \mathrm{tr}[\Theta_t \Sigma_{t|t}(\mathcal{S})]$.

$\sum_{t=1}^{T} tr[\Theta_t \Sigma_{t|t}(\mathcal{S})]$ *appearing in in eqs.* (8.12) *and* (8.14)*, we rewrite it as:*

$$\begin{aligned}\sum_{t=1}^{T} tr[\Theta_t \Sigma_{t|t}(\mathcal{S})] &= \sum_{t=1}^{T} \mathbb{E}\left(tr\{[x_t - \hat{x}_t(\mathcal{S})]^\top \Theta_t [x_t - \hat{x}_t(\mathcal{S})]\}\right) \\ &= \sum_{t=1}^{T} \mathbb{E}\left(\|K_t x_t - K_t \hat{x}_t(\mathcal{S})\|_{M_t}^2 \right), \end{aligned} \tag{8.16}$$

where in the first line we used the fact that $\Sigma_{t|t}(\mathcal{S}) = \mathbb{E}\left[(x_t - \hat{x}_t(\mathcal{S}))(x_t - \hat{x}_t(\mathcal{S}))^\top\right]$*, and in the second line we substituted the definition of* $\Theta_t = K_t^\top M_t K_t$ *from eq.* (8.11)*.*

From eq. (8.16)*, it is clear that each term* $tr[\Theta_t \Sigma_{t|t}(\mathcal{S})]$ *captures the expected mismatch between the imperfect state-information controller* $u_t(\mathcal{S}) = K_t \hat{x}_t(\mathcal{S})$ *(which is only aware of the measurements from the active sensors) and the perfect state-information controller* $K_t x_t$*. This is an important distinction from the existing sensor selection literature. In particular, while standard sensor selection attempts to minimize the estimation covariance, for instance*

by minimizing

$$\sum_{t=1}^{T} tr[\Sigma_{t|t}(\mathcal{S})] \triangleq \sum_{t=1}^{T} \mathbb{E}\left(\|x_t - \hat{x}_t(\mathcal{S})\|_2^2\right), \tag{8.17}$$

the proposed LQG cost formulation selectively minimizes the estimation error focusing on the states that are most informative for control purposes. For instance, the contribution to the total control mismatch in eq. (8.16) *of all* $x_t - \hat{x}_t(\mathcal{S})$ *in the null space of* K_t *is zero; accordingly, the proposed sensor design approach has no incentive in activating sensors to observe states which are irrelevant for control purposes. Overall, the importance of a state for control purposes is indeed captured by the weighting matrix* Θ_t*. Hence, in contrast to minimizing the cost function in eq.* (8.17)*, minimizing the cost function in eq.* (8.16) *results in a control-aware sensing design.*

8.3.2. Scalable near-optimal co-design algorithms for sensing-constrained LQG control (Problem 2)

This section proposes a practical algorithm for the sensing-constrained LQG control Problem 2. The pseudo-code of the algorithm is presented in Algorithm 15. Algorithm 15 follows the result of Theorem 17 and jointly designs sensing and control by first computing an active sensor set (Algorithm 15's lines 1-2) and then computing a control policy (Algorithm 15's lines 3-5). We discuss each step of the design below.

Near-optimal sensing design for Problem 2. Theorem 17 implies that an optimal sensor design for Problem 2 can be computed by solving the optimization problem in eq. (8.12). To this end, Algorithm 15 (line 1) first computes the matrices $\Theta_1, \Theta_2, \ldots, \Theta_T$, which appear in the objective function of the optimization problem in eq. (8.12) and, as result, they are necessary for its evaluation. Next, since the optimization problem in eq. (8.12) is combinatorial in nature, because it requires to select a subset of sensors out of all the available sensors in $\mathcal{V}$ that has sensor cost at most b and induces the smallest LQG cost, Algorithm 15's line 2 proposes a greedy algorithm, whose pseudo-code is given in Algorithm 16, to compute a (possibly approximate) solution to the problem in eq. (8.12). Our interest towards Algorithm 16 is motivated by that it is scalable and provably close to the solution of the problem in eq. (8.12) (in Section 8.4 we quantify its running time and provide sub-optimality bounds for its performance).

The steps that Algorithm 16 follows to compute a (possibly approximate) solution to the problem in eq. (8.12) are as follows: first, Algorithm 16 creates two candidate active sensor sets $\widehat{\mathcal{S}}_1$ and $\widehat{\mathcal{S}}_2$ (lines 1-2), of which only one will be selected as the (possibly approximate) solution to the problem in eq. (8.12) (line 20). In more detail, Algorithm 16's line 1 lets the set $\widehat{\mathcal{S}}_1$ be composed of a single sensor, namely the sensor $i \in \mathcal{V}$ that achieves the smallest value of the objective function in eq. (8.12) and having cost not exceeding the sensor selection budget ($c(i) \leq b$). Then, Algorithm 16's line 2 initializes the candidate active sensor set $\widehat{\mathcal{S}}_2$ with the empty set, and after the construction of the set $\widehat{\mathcal{S}}_2$ in Algorithm 16's lines 3–19 (which are explained below), Algorithm 16's line 20 computes which of the two sets $\widehat{\mathcal{S}}_1$ and $\widehat{\mathcal{S}}_2$ achieves the smallest value for the objective function in eq. (8.12), and returns this set as the (possibly approximate) solution to the optimization problem in eq. (8.12).

Algorithm 17 Joint Sensing and Control design for Problem 3.

Input: Time horizon T, available sensor set $\mathcal{V}$, LQG-cost bound κ, covariance $\Sigma_{1|0}$ of initial condition x_1; for all $t = 1, 2, \ldots, T$, system matrix A_t, input matrix B_t, LQG cost matrices Q_t and R_t, process noise covariance W_t; and for all sensors $i \in \mathcal{V}$, measurement matrix $C_{i,t}$, measurement noise covariance $V_{i,t}$, and sensor cost $c(i)$.

Output: Active sensors $\widehat{\mathcal{S}}$, and controls $\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_T$.

1: Compute the matrices $\Theta_1, \Theta_2, \ldots, \Theta_T$ using the backward Riccati recursion in eq. (8.11).
2: Return the sensor set $\widehat{\mathcal{S}}$ as the sensor set returned by Algorithm 18, which finds a (possibly approximate) solution to the optimization problem in eq. (8.9) ;
3: Compute the matrices $K_1, K_2, \ldots, K_T$ using the backward Riccati recursion in eq. (8.11).
4: At time $t = 1, 2, \ldots, T$, compute the Kalman estimate of the state x_t, i.e., the estimate:

$$\hat{x}_t \triangleq \mathbb{E}[x_t | y_1(\widehat{\mathcal{S}}), y_2(\widehat{\mathcal{S}}), \ldots, y_t(\widehat{\mathcal{S}})];$$

5: At time $t = 1, 2, \ldots, T$, return the control $\hat{u}_t = K_t \hat{x}_t$.

Lines 3–19 in Algorithm 16 populate the set $\widehat{\mathcal{S}}_2$ as follows: at each iteration of the "while loop" (lines 3-16) a sensor is greedily added to the set $\widehat{\mathcal{S}}_2$, as long as $\widehat{\mathcal{S}}_2$'s sensor cost does not exceed the sensor selection budget b. In particular, for each available sensor (the set $\mathcal{V}'$ contains the available sensors, excluding the ones already included in $\widehat{\mathcal{S}}_2$), the "for loop" (lines 4-12) computes first the estimation covariance resulting by adding the sensor to $\widehat{\mathcal{S}}_2$, and second the corresponding marginal gain in the objective function in eq. (8.12) (line 11). Then, the sensor that induces the largest sensor-cost-normalized marginal gain is selected (line 13), and it is added to the current set $\widehat{\mathcal{S}}_2$ (line 14). Finally, the "if" statement (lines 17-19) ensures that the constructed set $\widehat{\mathcal{S}}_2$ has sensor cost at most b, by possibly removing the sensor that was added in $\widehat{\mathcal{S}}_2$ during the last iteration of the "while" loop in lines 3-16.

Control design for Problem 2. Theorem 17 implies that given an active sensor set, the controls for Problem 2 can be computed according to the eq. (8.13). To this end, Algorithm 15 first computes the matrices $K_1, K_2, \ldots, K_T$ (line 3), and then, at each time $t = 1, 2, \ldots, T$, the Kalman estimate of the current state x_t (line 4), and the corresponding control (line 5).

8.3.3. Scalable near-optimal co-design algorithms for minimum-sensing LQG control (Problem 3)

This section proposes a practical algorithm for the minimum-sensing LQG control Problem 3. The pseudo-code of the algorithm is presented in Algorithm 17. Algorithm 17 follows the result of Theorem 17 and jointly designs sensing and control by first computing an active sensor set (Algorithm 17's lines 1-2) and then computing a control policy (Algorithm 17's lines 2-5). We discuss the first step (sensor design) in the rest of this section, while the second step (control design) is as in Algorithm 15's line 2, and is explained in Section 8.3.2.

Near-optimal sensing design for Problem 3. Theorem 17 implies that an optimal sensor design for Problem 3 can be computed by solving the optimization problem in eq. (8.14). To this end, similarly to Algorithm 15's line 1, Algorithm 17's line 1 computes the matrices

$\Theta_1, \Theta_2, \ldots, \Theta_T$, which are necessary for the evaluation of the cost function appearing in eq. (8.14). Next, Algorithm 17's line 2 calls Algorithm 18 to find a (possibly approximate) solution to the optimization problem in eq. (8.14). Analogously to the previous section, Algorithm 18 is a greedy algorithm that returns a near-optimal solution for the problem in eq. (8.14). The running time and the sub-optimality bounds of the algorithm are analyzed in Section 8.4.

Algorithm 18 computes a (possibly approximate) solution to the optimization problem in eq. (8.14) as follows: first, Algorithm 18 defines the constant $\bar{\kappa}$ (line 1), appearing in the definition of the optimization problem in eq. (8.14), and then initializes the sensor set $\widehat{\mathcal{S}}$ with the empty set (line 2). Afterwards, Algorithm 18 populates $\widehat{\mathcal{S}}$ in lines 3–16 using the following steps: at each iteration of the "while loop" (lines 3-16) a sensor is greedily added to the set $\widehat{\mathcal{S}}$, as long as Problem 3's LQG-cost bound κ has not been met, which eq. (8.14) guarantees to be equivalent to checking whether the second condition in Algorithm 18's line 3 holds. In particular, for each sensor in $\mathcal{V}'$ (set of available sensors, excluding the ones already included in $\widehat{\mathcal{S}}$) the "for loop" (lines 4-12) computes first the estimation covariance resulting by adding the sensor to $\widehat{\mathcal{S}}$, and then the corresponding marginal gain in the objective function in eq. (8.12) (line 11). Then, the sensor that induces the largest sensor-cost-normalized marginal gain is selected (line 13), and added to the current candidate active set $\widehat{\mathcal{S}}$ (line 14). Finally, the added sensor s is removed from $\mathcal{V}'$ (line 15).

In the following section we characterize the approximation and running-time performance of Algorithm 15 and Algorithm 17 for Problem 2 and Problem 3, respectively.

8.4. Performance guarantees for LQG Co-Design

We prove that Algorithm 15 and Algorithm 17 are the first scalable algorithms for the joint sensing and control design Problem 2 and Problem 3, respectively, and that they achieve an objective value that is close to the optimal. We start by introducing the notion of supermodularity ratio (Section 8.4.1), which will enable to bound the sub-optimality gap of Algorithm 15 (Section 8.4.2) and Algorithm 17 (Section 8.3.3). We then establish connections between the supermodularity ratio and control-theoretic quantities (Section 8.4.4).

8.4.1. Supermodularity ratio of monotone functions

This section introduces the notion of *supermodularity ratio* of a monotone set function (Definition 29). We start by defining the notions of monotonicity (Definition 27) and of supermodularity (Definition 28).

Definition 27 (Monotonicity)**.** *Consider any finite set $\mathcal{V}$. The set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* non-increasing *if and only if for any sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, it holds $f(\mathcal{A}) \geq f(\mathcal{B})$.*

Definition 28 (Supermodularity [70, Proposition 2.1])**.** *Consider any finite set $\mathcal{V}$. The set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* supermodular *if and only if for any sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, and any element $v \in \mathcal{V}$, it holds $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\}) \geq f(\mathcal{B}) - f(\mathcal{B} \cup \{v\})$.*

In words, a set function f is supermodular if and only if it satisfies the following diminishing returns property: for any element $v \in \mathcal{V}$, the marginal drop $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\})$ diminishes as the set $\mathcal{A}$ grows; equivalently, for any $\mathcal{A} \subseteq \mathcal{V}$ and $v \in \mathcal{V}$, the drop $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\})$

Algorithm 18 Sensing design for Problem 3.

Input: Time horizon T, available sensor set $\mathcal{V}$, LQGperformance bound κ, covariance $\Sigma_{1|0}$ of initial condition x_1, and for all $t = 1, 2, \ldots, T$ and any sensor $i \in \mathcal{V}$, matrix Θ_t, process noise covariance W_t, measurement matrix $C_{i,t}$, measurement noise covariance $V_{i,t}$, and sensor cost $c(i)$.

Output: Sensor set $\widehat{\mathcal{S}}$.

$\bar{\kappa} \leftarrow \kappa - \operatorname{tr}\left(\Sigma_{1|0} N_1\right) - \sum_{t=1}^{T} \operatorname{tr}\left(W_t S_t\right)$
$\widehat{\mathcal{S}} \leftarrow \emptyset; \quad \mathcal{V}' \leftarrow \mathcal{V};$
while $\mathcal{V}' \neq \emptyset$ and $\sum_{t=1}^{T} \operatorname{tr}[\Theta_t \Sigma_{t|t}(\widehat{\mathcal{S}})] > \bar{\kappa}$ **do**
  for all $a \in \mathcal{V}'$ **do**
    $\widehat{\mathcal{S}}_\alpha \leftarrow \widehat{\mathcal{S}} \cup \{a\}; \quad \Sigma_{1|0}(\widehat{\mathcal{S}}_\alpha) \leftarrow \Sigma_{1|0};$
    for all $t = 1, \ldots, T$ **do**
      $\Sigma_{t|t}(\widehat{\mathcal{S}}_\alpha) \leftarrow$
        $[\Sigma_{t|t-1}(\widehat{\mathcal{S}}_\alpha)^{-1} + C_t(\widehat{\mathcal{S}}_\alpha)^\top V_t(\widehat{\mathcal{S}}_\alpha)^{-1} C_t(\widehat{\mathcal{S}}_\alpha)]^{-1};$
      $\Sigma_{t+1|t}(\widehat{\mathcal{S}}_\alpha) \leftarrow A_t \Sigma_{t|t}(\widehat{\mathcal{S}}_\alpha) A_t^\top + W_t;$
    end for
    $\text{gain}_a \leftarrow \sum_{t=1}^{T} \operatorname{tr}\{\Theta_t [\Sigma_{t|t}(\widehat{\mathcal{S}}) - \Sigma_{t|t}(\widehat{\mathcal{S}}_\alpha)]\};$
  end for
  $s \leftarrow \arg\max_{a \in \mathcal{V}'} [\text{gain}_a / c(a)];$
  $\widehat{\mathcal{S}} \leftarrow \widehat{\mathcal{S}} \cup \{s\};$
  $\mathcal{V}' \leftarrow \mathcal{V}' \setminus \{s\};$
end while

is non-increasing.

Definition 29 (Supermodularity ratio)**.** *Consider any finite set $\mathcal{V}$, and a non-increasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$. We define the* supermodularity ratio *of f as*

$$\gamma_f \triangleq \min_{\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}, v \in \mathcal{V} \setminus \mathcal{B}} \frac{f(\mathcal{A}) - f(\mathcal{A} \cup \{v\})}{f(\mathcal{B}) - f(\mathcal{B} \cup \{v\})}.$$

In words, the supermodularity ratio of a monotone set function f measures how far f is from being supermodular. In particular, as per Definition 29 of the supermodularity ratio, the supermodularity ratio γ_f takes values in $[0, 1]$, and

- $\gamma_f = 1$ if and only if f is supermodular, since if $\gamma_f = 1$, then Definition 29 implies $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\}) \geq f(\mathcal{B}) - f(\mathcal{B} \cup \{v\})$, i.e., the drop $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\})$ is non-increasing as new elements are added in the set $\mathcal{A}$.

- $0 < \gamma_f < 1$ if and only if f is *approximately supermodular*, in the sense that if $\gamma_f < 1$, then Definition 29 implies $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\}) \geq \gamma_f [f(\mathcal{B}) - f(\mathcal{B} \cup \{v\})]$, i.e., the drop $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\})$ is approximately non-increasing as new elements are added in $\mathcal{A}$; specifically, the ratio γ_f captures how much ones needs to discount the drop $f(\mathcal{B}) - f(\mathcal{B} \cup \{v\})$, such that $f(\mathcal{A}) - f(\mathcal{A} \cup \{v\})$ remains greater then, or equal to, $f(\mathcal{B}) - f(\mathcal{B} \cup \{v\})$.

We next use the above supermodularity ratio notion to quantify the sub-optimality gap of Algorithm 15 and Algorithm 17.

8.4.2. Performance analysis for Algorithm 15

In this section we quantify Algorithm 15's running time and approximation performance (Theorem 18 below), using the notion of supermodularity ratio introduced in Section 8.4.1.

Theorem 18 (Performance of Algorithm 15)**.** *For any active sensor set $\mathcal{S} \subseteq \mathcal{V}$, and any admissible control policies $u_{1:T}(\mathcal{S}) \triangleq \{u_1(\mathcal{S}), u_2(\mathcal{S}), \ldots, u_T(\mathcal{S})\}$, let $h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ be Problem 2's cost function, i.e.,*

$$h[\mathcal{S}, u_{1:T}(\mathcal{S})] \triangleq \sum_{t=1}^{T} \mathbb{E}(\|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2).$$

Further define the following set-valued function and scalar:

$$g(\mathcal{S}) \triangleq \min_{u_{1:T}(\mathcal{S})} h[\mathcal{S}, u_{1:T}(\mathcal{S})], \tag{8.18}$$

$$g^\star \triangleq \min_{\mathcal{S}\subseteq\mathcal{V}, u_{1:T}(\mathcal{S})} h[\mathcal{S}, u_{1:T}(\mathcal{S})], \text{ s.t. } c(\mathcal{S}) \leq b;$$

that is, given a sensor set $\mathcal{S} \subseteq \mathcal{V}$, $g(\mathcal{S})$ is the optimal value of $h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ across all admissible control policies $u_{1:T}(\mathcal{S})$, and $g^\star$ is the optimal objective value of Problem 2.

The following results hold true:

1. *(Approximation quality) Algorithm 15 returns an active sensor set $\widehat{\mathcal{S}} \subseteq \mathcal{V}$ having cost $c(\widehat{\mathcal{S}})$ at most b, and the corresponding admissible control policies $u_{1:T}(\widehat{\mathcal{S}})$. The active sensors $\widehat{\mathcal{S}}$ and controls $u_{1:T}(\widehat{\mathcal{S}})$ are such that:*

$$\frac{h[\emptyset, u_{1:T}(\emptyset)] - h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})]}{h[\emptyset, u_{1:T}(\emptyset)] - g^\star} \geq \max\left[\frac{\gamma_g}{2}\left(1 - e^{-\gamma_g}\right), 1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b}\right], \tag{8.19}$$

   where γ_g is the supermodularity ratio of $g(\mathcal{S})$ in eq. (8.18)*.*

2. *(Running time) Algorithm 15 runs in $O(|\mathcal{V}|^2 T n^{2.4})$ time, where n is the system size in eq.* (8.1)*.*

Note that the term $h[\emptyset, u_{1:T}(\emptyset)] - h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})]$ quantifies the marginal gain of selecting the set $\widehat{\mathcal{S}}$, and ineq. (8.19) guarantees that the marginal gain is sufficiently large compared to the optimal marginal gain $h[\emptyset, u_{1:T}(\emptyset)] - g^\star$, in the sense that their ratio is lower bounded by the maximum between $\frac{\gamma_g}{2}\left(1 - e^{-\gamma_g}\right)$ and $1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b}$. We further comment on the bound in ineq. (8.19) in the following proposition and remarks.

Proposition 10 (Extension of the bound in ineq. (8.19) to sensor sets of any cost)**.** *Consider the modified version of Algorithm 15 where Algorithm 16's "if" statement (lines 17-19) is removed. Then, Algorithm 15's approximation performance bound remains as in ineq.* (8.19)*, even when Algorithm 15 returns a set $\widehat{\mathcal{S}}$ of cost $c(\widehat{\mathcal{S}})$ that exceeds Problem 2's budget b.*

Remark 13 (Comparison of bounds in ineq. (8.19))**.** *In Fig. 11 we plot Algorithm 15's*

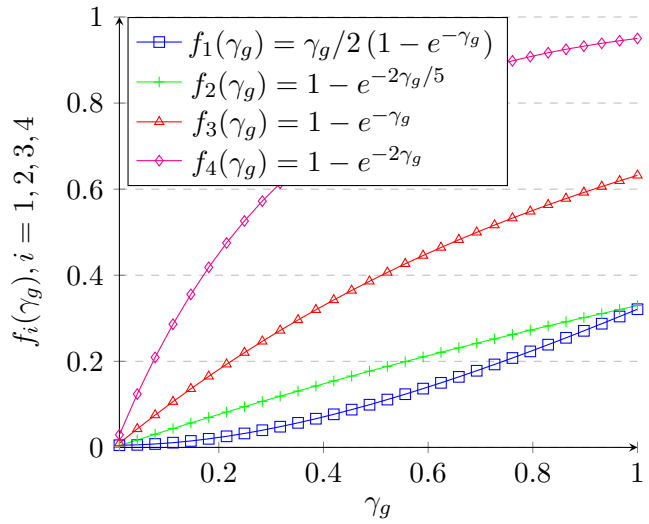


Figure 6: Plot of $f_i(\gamma_g)$ $(i = 1, 2, 3, 4)$ versus supermodularity ratio γ_g of a monotone supermodular function g. By Definition 29 of supermodularity ratio, γ_g takes values between 0 and 1. As γ_g increases from 0 to 1 then: $f_1(\gamma_g)$ increases from 0 to $1/2(1 - e^{-1}) \simeq 0.32$; $f_3(\gamma_g)$ increases from 0 to $1 - e^{-2/5} \simeq 0.32$; $f_2(\gamma_g)$ increases from 0 to $1 - e^{-1} \simeq 0.64$; $f_4(\gamma_g)$ increases from 0 to $1 - e^{-2} \simeq 0.87$.

approximation performance bounds in ineq. (8.19)*, namely the bound* $\gamma_g/2\,(1 - e^{-\gamma_g})$ *(function* $f_1(\gamma_g)$ *in Fig. 11) and the bound* $1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b}$ *(functions* $f_2(\gamma_g)$*,* $f_3(\gamma_g)$*, and* $f_4(\gamma_g)$ *in Fig. 11, which correspond to* $c(\widehat{\mathcal{S}})/b$ *equal to* 2/5*,* 1*, and* 2*, respectively; we note that for the latter case where* $c(\widehat{\mathcal{S}})/b$ *is equal to* 2*, we consider that Algorithm 15 has been modified per Proposition 10 to allow for active sensor sets with costs that exceed the selection budget* b*). We make two observations from Fig. 11: first, we observe that for ratio values* $c(\widehat{\mathcal{S}})/b > 2/5$*, the bound* $1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b}$ *in ineq.* (8.19) *dominates (i.e., is always larger —for all values of* γ_g*— than) the bound* $\gamma_g/2\,(1 - e^{-\gamma_g})$ *(compare plot of* $f_2(\gamma_g)$ *against that of* $f_1(\gamma_g)$*). Also, we observe from Fig. 11 that as the cost ratio* $c(\widehat{\mathcal{S}})/b$ *and the supermodularity ratio* γ_g *increase, the bound* $1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b}$ *tends to* 1 *(see plot of* $f_4(\gamma_g)$*).*

Remark 14 (Novelty of Algorithm 16 and of bound in ineq. (8.19))**.** *Algorithm 16 (used as a subroutine in Algorithm 15) is the first scalable algorithm with provable suboptimality guarantees for the minimization of a (possibly) approximately supermodular set function* g*, that is, a function* g *with supermodularity ratio* γ_g *(possibly) less than* 1*, subject to a heterogeneous-cost constraint. This generalizes existing algorithms for optimization with heterogeneous-cost constraints, which only focus on the special case of (exactly) supermodular functions (see, e.g., [44]), that is, functions* g *with supermodularity ratio* γ_g *(exactly) equal to* 1*.*

In addition, Algorithm 16 offers a tighter approximation performance bound for the optimization of (exactly) supermodular functions. Specifically, although the previous algorithms for the optimization of supermodular functions (see, e.g., [44]) have the same running time as Algorithm 16 and achieve the approximation performance bound $1/2\left(1 - e^{-1}\right)$*, which is*

the same as Algorithm 16's performance bound $\gamma_g/2\,(1-e^{-\gamma_g})$ *for* $\gamma_g = 1$ *(that is, for supermodular functions), Algorithm 16 also achieves the cost-dependent bound* $1-e^{-\gamma_g c(\widehat{\mathcal{S}})/b}$*, which for* $\gamma_g = 1$ *is superior to* $1/2\left(1-e^{-1}\right)$ *when the cost ratio* $c(\widehat{\mathcal{S}})/b$ *is more than* $2/5$ *(Remark 13).*

Theorem 18 guarantees that Algorithm 15 achieves an objective value for Problem 2 that is finitely close to optimal, whenever the supermodularity ratio γ_g is non-zero. In more detail, the extreme values of the bound in ineq. (8.19), as well as their interpretation with respect to Algorithm 15's approximation performance are as follows: the maximum value of the bounds in ineq. (8.19) is 1, which is achieved for supermodularity ratio $\gamma_g = 1$ and ratio $c(\widehat{\mathcal{S}})/b \longrightarrow +\infty$; as discussed in Proposition 10, the latter is possible when Algorithm 15 is modified to return an active sensor set with cost larger than b. On the other hand, when Algorithm 15 is not modified to return an active sensor set with cost larger than the budget b, and it always returns a sensor set with cost at most b, then the maximum value the bound in ineq. (8.19) can take is $1-1/e$ (for $\gamma_g = 1$ and $c(\widehat{\mathcal{S}}) = b$); notably, this is the best bound one can achieve in the worst-case in polynomial time even for supermodular objective functions [214]. The minimum value the bound in ineq. (8.19) is 0, which occurs for $\gamma_g = 0$.

The interpretation of the extreme values 0 and 1 of the bound in ineq. (8.19) is as follows: when the bound in ineq. (8.19) takes the value 1, then ineq. (8.19) implies that the approximate value $h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})]$ to Problem 2 is equal to the (optimal) value $g^\star$ of Problem 2, that is, Algorithm 15 is exact. Finally, when the bound in ineq. (8.19) is 0, ineq. (8.19) implies that $h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})] \leq h[\emptyset, u_{1:T}(\emptyset)]$, which is trivially satisfied[3] and, as a result, it is uninformative on the approximation performance of Algorithm 15. In Section 8.4.4 we present conditions under which the supermodularity ratio in ineq. (8.19) is guaranteed to be non-zero, in which case Algorithm 15 achieves near-optimal approximation performance.

Theorem 18 also ensures that Algorithm 15 is the first scalable algorithm for Problem 2. In particular, Algorithm 15's running time $O(|\mathcal{V}|^2 T n^{2.4})$ is in the worst-case quadratic in the number of the available sensors $|\mathcal{V}|$ (in the case where all the sensors in $\mathcal{V}$ are chosen as active) and linear in the Kalman filter's running time across the time horizon $\{1, 2, \ldots, T\}$; specifically, the contribution $n^{2.4}T$ in Algorithm 15's running time comes from the computational complexity of using the Kalman filter to compute the state estimation error covariances $\Sigma_{t|t}$ for each $t = 1, 2, \ldots, T$ [123, Appendix E].

8.4.3. Performance analysis for Algorithm 17

We quantify Algorithm 17's running time and approximation performance (Theorem 19 below), using the notion of supermodularity ratio introduced in Section 8.4.1.

Theorem 19 (Performance of Algorithm 17)**.** *Consider the notation introduced in the statement of Theorem 18 (Section 8.4.2): for any active sensor set* $\mathcal{S} \subseteq \mathcal{V}$*, and any admissible control policies* $u_{1:T}(\mathcal{S}) \triangleq \{u_1(\mathcal{S}), u_2(\mathcal{S}), \ldots, u_T(\mathcal{S})\}$*, let* $h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ *be the LQG cost func-*

[3] The inequality $h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})] \leq h[\emptyset, u_{1:T}(\emptyset)]$ simply states that a control policy that is informed by the active sensor set $\widehat{\mathcal{S}}$ has better performance than a policy that does not use any sensor; for a more formal proof we refer the reader to Appendix B.

tion in Problem 3's constraint, i.e.,

$$h[\mathcal{S}, u_{1:T}(\mathcal{S})] \triangleq \sum_{t=1}^{T} \mathbb{E}(\|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2);$$

Further define the following set-valued function:

$$g(\mathcal{S}) \triangleq \min_{u_{1:T}(\mathcal{S})} h[\mathcal{S}, u_{1:T}(\mathcal{S})]; \tag{8.20}$$

that is, given a sensor set $\mathcal{S} \subseteq \mathcal{V}$, $g(\mathcal{S})$ is the optimal value of $h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ across all admissible control policies $u_{1:T}(\mathcal{S})$.

Finally, let $b^\star$ be the optimal value of Problem 3, namely:

$$b^\star \triangleq \min_{\mathcal{S}\subseteq\mathcal{V}, u_{1:T}(\mathcal{S})} c(\mathcal{S}), \text{ s.t. } h[\mathcal{S}, u_{1:T}(\mathcal{S})] \leq \kappa.$$

The following results hold true:

1. *(Approximation quality) Algorithm 17 returns an active sensor set $\widehat{\mathcal{S}} \subset \mathcal{V}$ and admissible control policies $u_{1:T}(\widehat{\mathcal{S}})$. Let s_l denote the last sensor added to $\widehat{\mathcal{S}}$ by Algorithm 17: the active sensors $\widehat{\mathcal{S}}$ and controls $u_{1:T}(\widehat{\mathcal{S}})$ are such that:*

$$h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})] \leq \kappa; \tag{8.21}$$

$$c(\widehat{\mathcal{S}}) \leq c(s_l) + \frac{1}{\gamma_g} \log\left(\frac{h[\emptyset, u_{1:T}(\emptyset)] - \kappa}{h[\widehat{\mathcal{S}}_{l-1}, u_{1:T}(\widehat{\mathcal{S}}_{l-1})] - \kappa}\right) b^\star; \tag{8.22}$$

   where $\widehat{\mathcal{S}}_{l-1}$ is the subset of $\widehat{\mathcal{S}}$ that results by removing from $\widehat{\mathcal{S}}$ the last sensor added to it by Algorithm 17, i.e., $\widehat{\mathcal{S}}_{l-1} \triangleq \widehat{\mathcal{S}} \setminus \{s_l\}$; γ_g is the supermodularity ratio of $g(\mathcal{S})$.

2. *(Running time) Algorithm 17 runs in $O(|\mathcal{V}|^2 T n^{2.4})$ time, where n is the maximum system size in eq.* (8.1).

Remark 15 (Novelty of Algorithm 18 and of bound in ineq. (8.22))**.** *Algorithm 18 (used as a subroutine in Algorithm 17) is the first scalable algorithm with provable suboptimality guarantees for the minimum heterogeneous-cost set selection subject to a constraint on a (possibly) approximately supermodular function g, that is, a function g with supermodularity ratio γ_g less than* 1*. In particular, it generalizes previous algorithms that only focus on the special case of (exactly) supermodular functions (see, e.g., [210]), that is, functions g with supermodularity ratio γ_g equal to* 1*. Notably, for the case where the supermodularity ratio γ_g is equal to* 1 *(that is, the set function g is supermodular), and the sensor cost of the last sensor added to the returned set $\widehat{\mathcal{S}}$ by Algorithm 17 is equal to* 1 *($c(s_l) = 1$), then the bound in ineq.* (8.22) *becomes the same as the known bound in the supermodular function optimization literature for minimum cost set-selection [210, Theorem 1].*

Theorem 19, with ineq. (8.21), implies that Algorithm 17 returns a (possibly approximate) solution to Problem 3 that guarantees that the LQG-cost constraint in Problem 3 is satisfied.

Ineq. (8.22) also guarantees that for non-zero supermodularity ratio γ_g Algorithm 17 achieves

an objective value for Problem 3 that is finitely close to the optimal, since for non-zero γ_g the sensor cost of the set returned by Algorithm 17 is up to a finite multiplicative factor away from the optimal sensor set cost $b^\star$. In addition, ineq. (8.22) suggests that the approximation bound increases as the LQG-cost performance bound parameter κ decreases, that is, as we require from Algorithm 17 to find a sensor set that achieves a better (lower) LQG-cost performance.

Theorem 19 also ensures that Algorithm 17 is the first scalable algorithm for Problem 3. Notably, Algorithm 17's running time is equal in the worst-case to the running time of Algorithm 15 (which we discussed in Section 8.4.2).

In the following section we present control-theoretic conditions under which the supermodularity ratio γ_g in both Algorithm 15's and Algorithm 17's approximation bound in ineqs. (8.19) and (8.21) is non-zero, in which case Algorithm 15 and Algorithm 17 achieve near-optimal approximation performance.

8.4.4. Conditions for non-zero supermodularity ratio

In this section we provide conditions such that the supermodularity ratio γ_g in ineqs. (8.19) and (8.21) is non-zero, in which case both Algorithm 15 and Algorithm 17 guarantee a close to optimal approximate performance (per Theorem 18 and Theorem 19, respectively). In particular, we first prove that if the strict inequality $\sum_{t=1}^{T} \Theta_t \succ 0$ holds, where each Θ_t is defined as in eq. (8.11), then the supermodularity ratio γ_g is non-zero (Theorem 20). Then, we prove that the condition $\sum_{t=1}^{T} \Theta_t \succ 0$ holds true in all LQG control problem instances where a zero controller would result in a suboptimal behavior for the system; that is, we prove that $\sum_{t=1}^{T} \Theta_t \succ 0$ holds true in all system instances where LQG control design is necessary to achieve a desired system performance (Theorem 21).

The next theorem provides a non-zero computable bound for the supermodularity ratio γ_g in Theorem 18 and in Theorem 19.

Theorem 20 (Non-zero computable bound for the supermodularity ratio γ_g)**.** *Let the matrices Θ_t for all $t = 1, 2, \ldots, T$ be defined as in eq.* (8.11)*, the set function $g(\mathcal{S})$ be defined as in eq.* (8.18)*, and for any sensor $i \in \mathcal{V}$, the matrix $\bar{C}_{i,t} \triangleq V_{i,t}^{-1/2} C_{i,t}$ be the* whitened *measurement matrix.*

If the strict inequality $\sum_{t=1}^{T} \Theta_t \succ 0$ holds, then the supermodularity ratio γ_g is non-zero. In addition, if we assume (for simplicity in presentation) that the Frobenius norm of each $\bar{C}_{i,t}$ is 1*, i.e.,* $\operatorname{tr}\left(\bar{C}_{i,t}\bar{C}_{i,t}^\top\right) = 1$*, and that $tr[\Sigma_{t|t}(\emptyset)] \leq \lambda_{\max}^2[\Sigma_{t|t}(\emptyset)]$, then γ_g's lower bound is*

$$\gamma_g \geq \frac{\lambda_{\min}(\sum_{t=1}^{T} \Theta_t)}{\lambda_{\max}(\sum_{t=1}^{T} \Theta_t)} \frac{\min_{t\in\{1,2,\ldots,T\}} \lambda_{\min}^2[\Sigma_{t|t}(\mathcal{V})]}{\max_{t\in\{1,2,\ldots,T\}} \lambda_{\max}^2[\Sigma_{t|t}(\emptyset)]} \frac{1 + \min_{i\in\mathcal{V}, t\in\{1,2\ldots,T\}} \lambda_{\min}[\bar{C}_i \Sigma_{t|t}(\mathcal{V}) \bar{C}_i^\top]}{2 + \max_{i\in\mathcal{V}, t\in\{1,2\ldots,T\}} \lambda_{\max}[\bar{C}_i \Sigma_{t|t}(\emptyset) \bar{C}_i^\top]}. \tag{8.23}$$

Ineq. (8.23) suggests two cases under which γ_g can increase, and, correspondingly, the ap-

proximation performance bounds of Algorithm 15 and of Algorithm 17 in ineqs. (8.19) and (8.21), respectively, can improve; in more detail:

Case 1 where the bound of γ_g in ineq. (8.23) *increases:* When the fraction:

$$\lambda_{\min}(\sum_{t=1}^{T}\Theta_t)/\lambda_{\max}(\sum_{t=1}^{T}\Theta_t)$$

increases to 1, then the right-hand-side in ineq. (8.23) increases. Therefore, since the matrices Θ_t weight the states depending on their relevance for control purposes (Remark 12), the right-hand-side in ineq. (8.23) increases when on average all the directions in the state space become equally important for control purposes. Indeed, in the extreme case where $\lambda_{\max}(\Theta_t) = \lambda_{\min}(\Theta_t) = \lambda$, the cost function in eq. (8.12) that Algorithm 15 minimizes to select the active sensor set becomes

$$\sum_{t=1}^{T}\mathrm{tr}[\Theta_t\Sigma_{t|t}(\mathcal{S})] = \lambda\sum_{t=1}^{T}\mathrm{tr}[\Sigma_{t|t}(\mathcal{S})],$$

which matches the cost function in the classical sensor selection where all states are equally important (per eq. (8.17)).

Case 2 where the bound of γ_g in ineq. (8.23) *increases:* When either the numerators of the last two fractions in the right-hand-side of ineq. (8.23) increase or the denominators of the last two fractions in the right-hand-side of ineq. (8.23) decrease, then the right-hand-side in ineq. (8.23) increases. In particular, the numerators of the last two fractions in the right-hand-side of ineq. (8.23) capture the (best) estimation quality when all available sensors in $\mathcal{V}$ are used, via the terms of the form $\lambda_{\min}[\Sigma_{t|t}(\mathcal{V})]$ and $\lambda_{\min}[\bar{C}_{i,t}\Sigma_{t|t}(\mathcal{V})\bar{C}_{i,t}^{\top}]$. Interestingly, this suggests that the right-hand-side of ineq. (8.23) increases when the available sensors in $\mathcal{V}$ are inefficient in achieving low estimation error, that is, when the terms of the form $\lambda_{\min}[\Sigma_{t|t}(\mathcal{V})]$ and $\lambda_{\min}[\bar{C}_{i,t}\Sigma_{t|t}(\mathcal{V})\bar{C}_{i,t}^{\top}]$ increase. Similarly, the denominators of the last two fractions in the right-hand-side of ineq. (8.23) capture the (worst) estimation quality when no sensor is active, via the terms of the form $\lambda_{\max}[\Sigma_{t|t}(\emptyset)]$ and $\lambda_{\max}[\bar{C}_{i,t}\Sigma_{t|t}(\emptyset)\bar{C}_{i,t}^{\top}]$. This suggests that the right-hand-side of ineq. (8.23) increases when the measurement noise increases.

Theorem 20 states that the supermodularity ratio γ_g is non-zero whenever $\sum_{t=1}^{T}\Theta_t \succ 0$. To provide insight on the type of control problems for which this result holds, in the following theorem we translate the technical condition $\sum_{t=1}^{T}\Theta_t \succ 0$ into an equivalent control-theoretic condition.

Theorem 21 (Control-theoretic condition for near-optimal co-design)**.** *Consider the (noiseless, perfect state-information) LQG problem where at any time $t = 1, 2, \ldots, T$, the state x_t is known to each controller u_t and the process noise w_t is zero, i.e., the optimal control problem*

$$\min_{u_{1:T}} \sum_{t=1}^{T}\left[\|x_{t+1}\|_{Q_t}^2 + \|u_t(x_t)\|_{R_t}^2\right]\Big|_{\Sigma_{t|t}=W_t=0}. \tag{8.24}$$

Let A_t be invertible for all $t = 1, 2, \ldots, T$; the strict inequality $\sum_{t=1}^{T}\Theta_t \succ 0$ holds if and

only if for all non-zero initial conditions x_1, the all-zeroes control policy $u^{\circ}_{1:T} \triangleq (0, 0, \ldots, 0)$ is not an optimal solution to the optimal control problem in eq. (8.24)*:*

$$u^{\circ}_{1:T} \notin \arg\min_{u_{1:T}} \sum_{t=1}^{T} [\|x_{t+1}\|^2_{Q_t} + \|u_t(x_t)\|^2_{R_t}]\Big|_{\Sigma_{t|t}=W_t=0}.$$

Theorem 21 suggests that the condition $\sum_{t=1}^{T} \Theta_t \succ 0$ (which ensures a non-zero supermodularity ratio γ_g per Theorem 20) holds if and only if for any non-zero initial condition x_1 the all-zeroes control policy $u^{\circ}_{1:T} = (0, 0, \ldots, 0)$ is suboptimal for the noiseless, perfect state-information LQG problem in eq. (8.24); intuitively, this encompasses most practical control design problems where a zero controller would result in a suboptimal behavior of the system (LQG control design itself would be unnecessary in the case where a zero controller, i.e., no control action, can already attain the desired system performance).

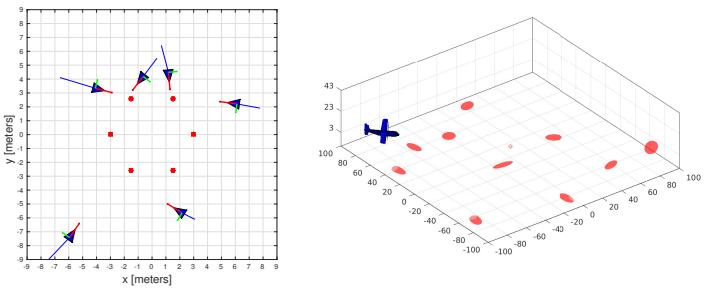
Overall, Algorithm 15 and Algorithm 17 are the first scalable algorithms for Problem 2 and for Problem 3, respectively, and (for the LQG control problem instances where a zero controller would result in a suboptimal behavior for the system and, as a result, for the system instances where LQG control design is necessary to achieve a desired system performance) they achieve close to optimal approximate performance.

8.5. Numerical Experiments

We consider two application scenarios for the proposed sensing-constrained LQG control framework: *sensing-constrained formation control* and *resource-constrained robot navigation*. We present a Monte Carlo analysis for both scenarios, which demonstrates that (i) the proposed sensor selection strategy is near-optimal, and in particular, the resulting LQG-cost (tracking performance) matches the optimal selection in all tested instances for which the optimal selection could be computed via a brute-force approach, (ii) a more naive selection which attempts to minimize the state estimation covariance [5] (rather than the LQG cost) has degraded LQG tracking performance, often comparable to a random selection, (iii) in the considered instances, a clever selection of a small subset of sensors can ensure an LQG cost that is close to the one obtained by using all available sensors, hence providing an effective alternative for control under sensing constraints [60].

8.5.1. Sensing-constrained formation control

Simulation setup. The first application scenario is illustrated in Fig. 7(a). A team of n agents (blue triangles) moves in a 2D scenario. At time $t = 1$, the agents are randomly deployed in a $10\text{m} \times 10\text{m}$ square and their objective is to reach a target formation shape (red stars); in the example of Fig. 7(a) the desired formation has an hexagonal shape, while in general for a formation of n, the desired formation is an equilateral polygon with n vertices. Each robot is modeled as a double-integrator, with state $x_i = [p_i \; v_i]^\top \in \mathbb{R}4$ (p_i is the 2D position of agent i, while v_i is its velocity), and can control its own acceleration $u_i \in \mathbb{R}2$; the process noise is chosen as a diagonal matrix $W = \text{diag}\left([1e^{-2},\ 1e^{-2},\ 1e^{-4},\ 1e^{-4}]\right)$. Each robot i is equipped with a GPS receiver, which can measure the agent position p_i with a covariance $V_{gps,i} = 2 \cdot \mathbf{I}_2$. Moreover, the agents are equipped with lidar sensors allowing each



(a) formation control (b) unmanned aerial robot

Figure 7: Examples of applications of the proposed sensing-constrained LQGcontrol framework: (a) sensing-constrained formation control and (b) resource-constrained robot navigation.

agent i to measure the relative position of another agent j with covariance $V_{lidar,ij} = 0.1 \cdot \mathbf{I}_2$. The agents have very limited on-board resources, hence they can only activate a subset of k sensors. Hence, the goal is to select the subset of k sensors, as well as to compute the control policy that ensure best tracking performance, as measured by the LQG objective.

For our tests, we consider two problem setups. In the first setup, named *homogeneous formation control*, the LQG weigh matrix Q is a block diagonal matrix with 4×4 blocks, with each block i chosen as $Q_i = 0.1 \cdot \mathbf{I}_4$; since each 4×4 block of Q weights the tracking error of a robot, in the homogeneous case the tracking error of all agents is equally important. In the second setup, named *heterogeneous formation control*, the matrix Q is chose as above, except for one of the agents, say robot 1, for which we choose $Q_1 = 10 \cdot \mathbf{I}_4$; this setup models the case in which each agent has a different role or importance, hence one weights differently the tracking error of the agents. In both cases the matrix R is chosen to be the identity matrix. The simulation is carried on over T time steps, and T is also chosen as LQG horizon. Results are averaged over 100 Monte Carlo runs: at each run we randomize the initial estimation covariance $\Sigma_{1|0}$.

Compared techniques. We compare five techniques. All techniques use an LQG-based estimator and controller, and they only differ by the selections of the sensors used. The first approach is the optimal sensor selection, denoted as `optimal`, which attains the minimum of the cost function in eq. (8.12), and that we compute by enumerating all possible subsets; this brute-force approach is only viable when the number of available sensors is small. The second approach is a pseudo-random sensor selection, denoted as `random`*, which selects all the GPS measurements and a random subset of the lidar measurements; note that we do not consider a fully random selection since in practice this often leads to an unobservable system, hence causing divergence of the LQG cost. The third approach, denoted as `logdet`, selects sensors so to minimize the average $\log\det$ of the estimation covariance over the horizon; this approach resembles [5] and is agnostic to the control task. The fourth approach is the proposed sensor selection strategy, described in Algorithm 16, and is denoted as `s-LQG`. Finally, we also report the LQG performance when all sensors are selected; this is clearly infeasible in practice, due to the sensing constraints, and it is only reported for comparison purposes. This approach is denoted as `allSensors`.

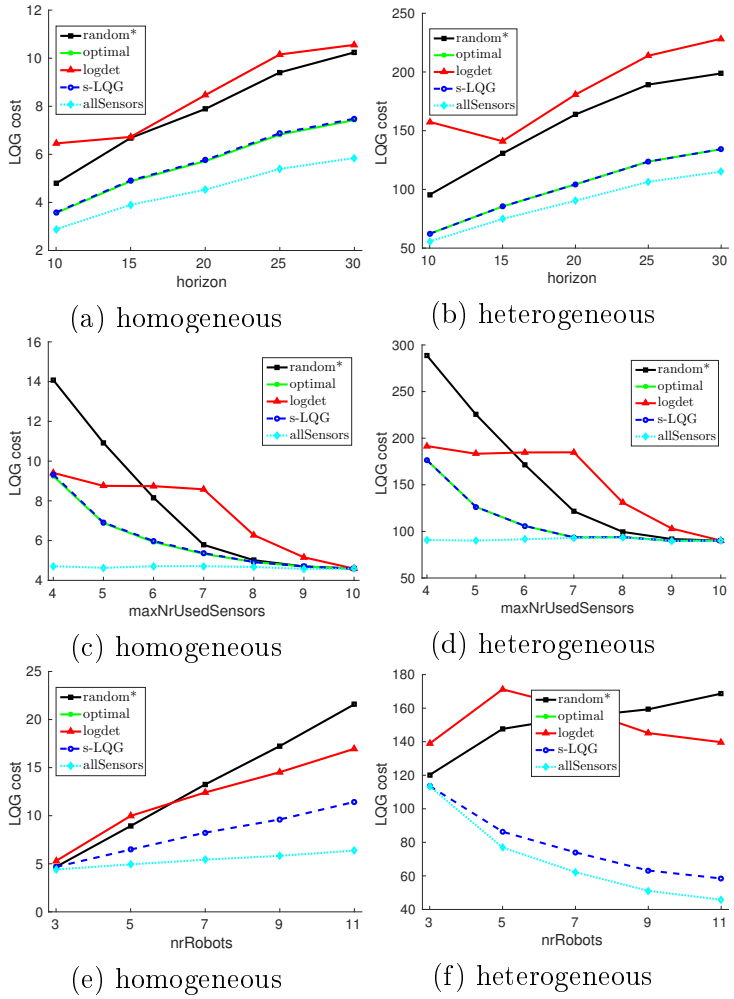
Results. The results of our numerical analysis are reported in Fig. 12. When not specified otherwise, we consider a formation of $n = 4$ agents, which can only use a total of $k = 6$ sensors, and a control horizon $T = 20$. Fig. 12(a) shows the LQG cost attained by the compared techniques for increasing control horizon and for the homogeneous case. We note that, in all tested instance, the proposed approach `s-LQG`matches the optimal selection `optimal`, and both approaches are relatively close to `allSensors`, which selects all the available sensors ($\frac{n+n^2}{2}$). On the other hand `logdet`leads to worse tracking performance, and it is often close to the pseudo-random selection `random`*. These considerations are confirmed by the heterogeneous setup, shown in Fig. 12(b). In this case the separation between the proposed approach and `logdet`becomes even larger; the intuition here is that the heterogeneous case rewards differently the tracking errors at different agents, hence while `logdet`attempts to equally reduce the estimation error across the formation, the proposed approach `s-LQG`selects sensors in a task-oriented fashion, since the matrices Θ_t for all $t = 1, 2, \ldots, T$ in the cost function in eq. (8.12) incorporate the LQG weight matrices.

Fig. 12(c) shows the LQG cost attained by the compared techniques for increasing number of selected sensors k and for the homogeneous case. We note that for increasing number of sensors all techniques converge to `allSensors`(the entire ground set is selected). As in the previous case, the proposed approach `s-LQG`matches the optimal selection `optimal`. Interestingly, while the performance of `logdet`is in general inferior with respect to `s-LQG`, when the number of selected sensors k decreases (for $k < n$ the problem becomes unobservable) the approach `logdet`performs similarly to `s-LQG`. Fig. 12(d) shows the same statistics for the heterogeneous case. We note that in this case `logdet`is inferior to `s-LQG`even in the case with small k. Moreover, an interesting fact is that `s-LQG`matches `allSensors`already for $k = 7$, meaning that the LQG performance of the sensing-constraint setup is indistinguishable from the one using all sensors; intuitively, in the heterogeneous case, adding more sensors may have marginal impact on the LQG cost (e.g., if the cost rewards a small tracking error for robot 1, it may be of little value to take a lidar measurement between robot 3 and 4). This further stresses the importance of the proposed framework as a parsimonious way to control a system with minimal resources.

Fig. 12(e) and Fig. 12(f) show the LQG cost attained by the compared techniques for increasing number of agents, in the homogeneous and heterogeneous case, respectively. To ensure observability, we consider $k = \text{round}\,(1.5n)$, i.e., we select a number of sensors 50% larger than the smallest set of sensors that can make the system observable. We note that `optimal`quickly becomes intractable to compute, hence we omit values beyond $n = 4$. In both figures, the main observation is that the separation among the techniques increases with the number of agents, since the set of available sensors quickly increases with n. Interestingly, in the heterogeneous case `s-LQG`remains relatively close to `allSensors`, implying that for the purpose of LQG control, using a cleverly selected small subset of sensors still ensures excellent tracking performance.

8.5.2. Resource-constrained robot navigation

Simulation setup. The second application scenario is illustrated in Fig. 7(b). An unmanned aerial robot (UAV) moves in a 3D scenario, starting from a randomly selected initial



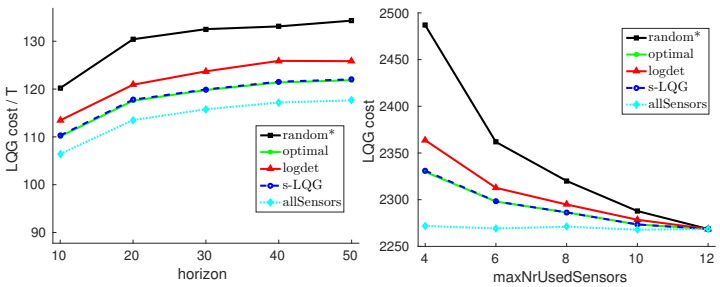
(a) homogeneous (b) heterogeneous

(c) homogeneous (d) heterogeneous

(e) homogeneous (f) heterogeneous

Figure 8: LQGcost for increasing (a)-(b) control horizon T, (c)-(d) number of selected sensors k, and (e)-(f) number of agents n. Statistics are reported for the homogeneous formation control setup (left column), and the heterogeneous setup (right column). Results are averaged over 100 Monte Carlo runs.

location. The objective of the UAV is to land, and more specifically, it has to reach the position [0, 0, 0] with zero velocity. The UAV is modeled as a double-integrator, with state $x_i = [p_i \; v_i]^\top \in \mathbb{R}6$ (p_i is the 3D position of agent i, while v_i is its velocity), and can control its own acceleration $u_i \in \mathbb{R}3$; the process noise is chosen as $W = \mathbf{I}_6$. The UAV is equipped with multiple sensors. It has an on-board GPS receiver, measuring the UAV position p_i with a covariance $2 \cdot \mathbf{I}_3$, and an altimeter, measuring only the last component of p_i (altitude) with standard deviation 0.5m. Moreover, the UAV can use a stereo camera to measure the relative position of ℓ landmarks on the ground; for the sake of the numerical example, we assume the location of each landmark to be known only approximately, and we associate to each landmark an uncertainty covariance (red ellipsoids in Fig. 7(b)), which is randomly generated at the beginning of each run. The UAV has limited on-board resources, hence it can only activate a subset of k sensors. For instance, the resource-constraints may be due to the power consumption of the GPS and the altimeter, or may be due to computational



(a) heterogeneous (b) heterogeneous

Figure 9: LQGcost for increasing (a) control horizon T, and (b) number of selected sensors k. Statistics are reported for the heterogeneous setup. Results are averaged over 100 Monte Carlo runs.

constraints that prevent to run multiple object-detection algorithms to detect all landmarks on the ground. Similarly to the previous case, we phrase the problem as a sensing-constraint LQG problem, and we use $Q = \text{diag}\left([1e^{-3},\ 1e^{-3},\ 10,\ 1e^{-3},\ 1e^{-3},\ 10]\right)$ and $R = \mathbf{I}_3$. Note that the structure of Q reflects the fact that during landing we are particularly interested in controlling the vertical direction and the vertical velocity (entries with larger weight in Q), while we are less interested in controlling accurately the horizontal position and velocity (assuming a sufficiently large landing site). In the following, we present results averaged over 100 Monte Carlo runs: in each run, we randomize the covariances describing the landmark position uncertainty.

Compared techniques. We consider the five techniques discussed in the previous section. As in the formation control case, the pseudo-random selection `random`*always includes the GPS measurement (which alone ensures observability) and a random selection of the other available sensors.

Results. The results of our numerical analysis are reported in Fig. 9. When not specified otherwise, we consider a total of $k = 3$ sensors to be selected, and a control horizon $T = 20$. Fig. 9(a) shows the LQG cost attained by the compared techniques for increasing control horizon. For visualization purposes we plot the cost normalized by the horizon, which makes more visible the differences among the techniques. Similarly to the formation control example, `s-LQG`matches the optimal selection `optimal`, while `logdet`and `random`*have suboptimal performance.

Fig. 9(b) shows the LQG cost attained by the compared techniques for increasing number of selected sensors k. Clearly, all techniques converge to `allSensors`for increasing k, but in the regime in which few sensors are used `s-LQG`still outperforms alternative sensor selection schemes, and matches in all cases the optimal selection `optimal`.

8.6. Concluding Remarks & Future Work

In this chapter, we introduced the LQG control and sensing co-design problem, where one has to jointly design a suitable sensing, estimation, and control policy to steer the behavior of a linear Gaussian systems under resource constraints. We discussed two variants of the problem, named *sensing-constrained LQG control* and *minimum-sensing LQG control*,

which are central in modern control applications ranging from large-scale networked systems to miniaturized robotics networks. While the resulting co-design problems are intractable in general, we provide the first scalable algorithms that can compute a design that is provably close to the optimal one. While developing these algorithms, we also extend the literature on supermodular optimization, by providing the first efficient algorithms for the optimization of (approximately) supermodular functions subject to heterogeneous-cost constraints, and by improving existing performance bounds. Notably, our performance bounds have a clear connection to control-theoretic quantities and are proven to be non-vanishing under very general conditions (we prove that the suboptimality gap is non-vanishing whenever the open loop behavior of the system deviates from the desired closed-loop behavior, hence encompassing most real-world control problems). Finally, we provided illustrative examples and a numerical analysis considering problems in robotics and networked control.

This chapter opens a number of avenues for future research. First, while this chapter provides an introduction to LQG sensing and control co-design, other interesting co-design problems exist; for instance, one may consider actuation-and-control co-design problems, or even sensing-actuation-control co-design. Second, one may extend the LQG co-design problem to account for potential sensor failures, where some of the selected sensors do not work as expected; to this end, one could leverage recent results on *resilient submodular optimization* [56]. Finally, while we currently provide bounds between our sensor design and the optimal sensor design, we find interesting to provide bounds that compare the LQG performance attained when an optimal subset of sensors is used with the LQG performance attained when all available sensors is used.

8.7. Appendix: Proof of Results

8.7.1. Preliminary facts

This appendix contains a set of lemmata that will be used to support the proofs in this chapter (Appendices B–F).

Lemma 10 ([215, Proposition 8.5.5])**.** *Consider two positive definite matrices A_1 and A_2. If $A_1 \preceq A_2$ then $A_2^{-1} \preceq A_1^{-1}$.*

Lemma 11 (Trace inequality [215, Proposition 8.4.13])**.** *Consider a symmetric matrix A, and a positive semi-definite matrix B of appropriate dimension. Then,*

$$\lambda_{\min}(A)\mathrm{tr}\,(B) \leq \mathrm{tr}\,(AB) \leq \lambda_{\max}(A)\mathrm{tr}\,(B)\,.$$

Lemma 12 (Woodbury identity [215, Corollary 2.8.8])**.** *Consider matrices A, C, U and V of appropriate dimensions, such that A, C, and $A+UCV$ are invertible. Then,*

$$(A+UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1}+VA^{-1}U)^{-1}VA^{-1}.$$

Lemma 13 ([215, Proposition 8.5.12])**.** *Consider two symmetric matrices A_1 and A_2, and a positive semi-definite matrix B. If $A_1 \preceq A_2$, then $\mathrm{tr}\,(A_1B) \leq \mathrm{tr}\,(A_2B)$.*

Lemma 14 ([123, Appendix E])**.** *For any sensor set $\mathcal{S} \subseteq \mathcal{V}$, and for all $t = 1, 2, \ldots, T$, let $\hat{x}_t(\mathcal{S})$ be the Kalman estimator of the state x_t, i.e., $\hat{x}_t(\mathcal{S}) \triangleq \mathbb{E}[x_t | y_1(\mathcal{S}), y_2(\mathcal{S}), \ldots, y_t(\mathcal{S})]$, and $\Sigma_{t|t}(\mathcal{S})$ be $\hat{x}_t(\mathcal{S})$'s error covariance, i.e., $\Sigma_{t|t}(\mathcal{S}) \triangleq \mathbb{E}[(\hat{x}_t(\mathcal{S}) - x_t)(\hat{x}_t(\mathcal{S}) - x_t)^\top]$. Then,*

$\Sigma_{t|t}(\mathcal{S})$ *is the solution of the Kalman filtering recursion*

$$\begin{aligned}\Sigma_{t|t}(\mathcal{S}) &= [\Sigma_{t|t-1}(\mathcal{S})^{-1} + C_t(\mathcal{S})^\top V_t(\mathcal{S})^{-1} C_t(\mathcal{S})]^{-1};\\ \Sigma_{t+1|t}(\mathcal{S}) &= A_t \Sigma_{t|t}(\mathcal{S}) A_t^\top + W_t,\end{aligned}$$

with boundary condition $\Sigma_{1|0}(\mathcal{S}) = \Sigma_{1|0}$.

Lemma 15. *For any sensor set* $\mathcal{S} \subseteq \mathcal{V}$, *let* $\Sigma_{1|1}(\mathcal{S})$ *be defined as in eq.* (14), *and consider two sensor sets* $\mathcal{S}_1, \mathcal{S}_2 \subseteq \mathcal{V}$. *If* $\mathcal{S}_1 \subseteq \mathcal{S}_2$, *then* $\Sigma_{1|1}(\mathcal{S}_1) \succeq \Sigma_{1|1}(\mathcal{S}_2)$.

Proof of Lemma 15 Let $\mathcal{D} = \mathcal{S}_2 \backslash \mathcal{S}_1$, and observe that for all $t = 1, 2, \ldots, T$, the notation in Definition 25 implies

$$\begin{aligned}C_t(\mathcal{S}_2)^\top V_t(\mathcal{S}_2)^{-1} C_t(\mathcal{S}_2) &= \sum_{i \in \mathcal{S}_2} C_{i,t}^\top V_{i,t} C_{i,t}\\ &= \sum_{i \in \mathcal{S}_1} C_{i,t}^\top V_{i,t} C_{i,t} + \sum_{i \in \mathcal{D}} C_{i,t}^\top V_{i,t} C_{i,t}\\ &\succeq \sum_{i \in \mathcal{S}_1} C_{i,t}^\top V_{i,t} C_{i,t}\\ &= C_t(\mathcal{S}_1)^\top V_t(\mathcal{S}_1)^{-1} C_t(\mathcal{S}_1). \qquad (8.25)\end{aligned}$$

Therefore, Lemma 10 and ineq. (8.25) imply

$$\begin{aligned}&\Sigma_{1|1}(\mathcal{S}_2) = [\Sigma_{1|0}^{-1} + C_1(\mathcal{S}_2)^\top V_t(\mathcal{S}_2)^{-1} C_t(\mathcal{S}_2)]^{-1} \preceq\\ &[\Sigma_{1|0}^{-1} + C_1(\mathcal{S}_1)^\top V_t(\mathcal{S}_1)^{-1} C_t(\mathcal{S}_1)]^{-1} = \Sigma_{1|1}(\mathcal{S}_1).\end{aligned}$$ ■

Lemma 16. *Let* $\Sigma_{t|t}$ *be defined as in eq.* (14) *with boundary condition* $\Sigma_{1|0}$; *similarly, let* $\bar{\Sigma}_{t|t}$ *be defined as in eq.* (14) *with boundary condition* $\bar{\Sigma}_{1|0}$. *If* $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$, *then* $\Sigma_{t+1|t} \preceq \bar{\Sigma}_{t+1|t}$.

Proof of Lemma 16 We complete the proof in two steps: first, from $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$, it follows $A_t \Sigma_{t|t} A_t^\top \preceq A_t \bar{\Sigma}_{t|t} A_t^\top$. Then, from eq. (14), it its $\Sigma_{t+1|t} = A_t \Sigma_{t|t} A_t^\top + W_t \preceq A_t \bar{\Sigma}_{t|t} A_t^\top + W_t = \bar{\Sigma}_{t+1|t}$. ■

Lemma 17. *Let* $\Sigma_{t|t-1}$ *be defined as in eq.* (14) *with boundary condition* $\Sigma_{1|0}$; *and, let* $\bar{\Sigma}_{t|t-1}$ *be defined as in eq.* (14) *with boundary condition* $\bar{\Sigma}_{1|0}$. *If* $\Sigma_{t|t-1} \preceq \bar{\Sigma}_{t|t-1}$, *then* $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$.

Proof of Lemma 17 From eq. (14), it is $\Sigma_{t|t} = (\Sigma_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t)^{-1} \preceq (\bar{\Sigma}_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t)^{-1} = \bar{\Sigma}_{t|t}$, since Lemma 10 and the condition $\Sigma_{t|t-1} \preceq \bar{\Sigma}_{t|t-1}$ imply $\Sigma_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t \succeq \bar{\Sigma}_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t$, which in turn implies $(\Sigma_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t)^{-1} \preceq (\bar{\Sigma}_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t)^{-1}$. ■

Corollary 8. *Let* $\Sigma_{t|t}$ *be defined as in eq.* (14) *with boundary condition* $\Sigma_{1|0}$; *similarly, let* $\bar{\Sigma}_{t|t}$ *be defined as in eq.* (14) *with boundary condition* $\bar{\Sigma}_{1|0}$. *If* $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$, *then* $\Sigma_{t+i|t+i} \preceq \bar{\Sigma}_{t+i|t+i}$ *for any positive integer* i.

Proof of Corollary 8 If $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$, from Lemma 16 we get $\Sigma_{t+1|t} \preceq \bar{\Sigma}_{t+1|t}$, which, from Lemma 17, implies $\Sigma_{t+1|t+1} \preceq \bar{\Sigma}_{t+1|t+1}$. By repeating the previous argument another $(i-1)$ times, the proof is complete. ■

Corollary 9. *Let $\Sigma_{t|t}$ be defined as in eq. (14) with boundary condition $\Sigma_{1|0}$; similarly, let $\bar{\Sigma}_{t|t}$ be defined as in eq. (14) with boundary condition $\bar{\Sigma}_{1|0}$. If $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$, then $\Sigma_{t+i|t+i-1} \preceq \bar{\Sigma}_{t+i|t+i-1}$ for any positive integer i.*

Proof of Corollary 9 If $\Sigma_{t|t} \preceq \bar{\Sigma}_{t|t}$, from Corollary 8, we get $\Sigma_{t+i-1|t+i-1} \preceq \bar{\Sigma}_{t+i-1|t+i-1}$, which, from Lemma 16, implies $\Sigma_{t+i|t+i-1} \preceq \bar{\Sigma}_{t+i|t+i-1}$. ■

Lemma 18. *Consider positive real numbers a, b, γ, $a_1, a_2, \ldots, a_n$ such that $\sum_{i=1}^n a_i = a$. The function*

$$f(a_1, a_2, \ldots, a_n) = 1 - \prod_{i=1}^{n} \left(1 - \gamma \frac{a_i}{b}\right)$$

achieves its minimum at $a_1 = a_2 = \ldots = a_n = a/n$. In particular,

$$f(a/n, a/n, \ldots, a/n) = 1 - \left(1 - \frac{a\gamma}{bn}\right)^n \geq 1 - e^{-a\gamma/b}.$$

Proof of Lemma 18 To find f's minimum we use the method of Lagrange multipliers. In particular, the partial derivative of $\phi(a_1, a_2, \ldots, a_n) \triangleq f(a_1, a_2, \ldots, a_n) + \lambda \left(\sum_{i=1}^n a_i - a\right)$, where λ is the Lagrangian multiplier, with respect to a_j is as follows:

$$\frac{\partial \phi}{\partial a_j} = \frac{\gamma}{b} \prod_{i \neq j} \left(1 - \gamma \frac{a_i}{b}\right) + \lambda. \tag{8.26}$$

At an f's minimum, the partial derivative in eq. (8.26) is zero for all j, which implies that for all j:

$$\lambda = -\frac{\gamma}{b} \prod_{i \neq j} \left(1 - \gamma \frac{a_i}{b}\right); \tag{8.27}$$

Since λ is constant, eq. (8.27) implies for any j_1 and j_2 that

$$\frac{\gamma}{b} \prod_{i \neq j_1} \left(1 - \gamma \frac{a_i}{b}\right) = \frac{\gamma}{b} \prod_{i \neq j_2} \left(1 - \gamma \frac{a_i}{b}\right),$$

which in turn implies that

$$1 - \gamma \frac{a_{j_1}}{b} = 1 - \gamma \frac{a_{j_2}}{b},$$

from where we conclude $a_{j_1} = a_{j_2}$ (for any j_1 and j_2).

The lower bound for the minimum value of f follows from the fact that $1 - x \leq e^{-x}$ for all real x. ■

Proposition 11 (Monotonicity of cost function in eq. (8.12))**.** *Consider the cost function in eq. (8.12), namely, for any sensor set $\mathcal{S} \subseteq \mathcal{V}$ the set function $\sum_{t=1}^T \operatorname{tr}\left(\Theta_t \Sigma_{t|t}(\mathcal{S})\right)$. Then, for any sensor sets such that $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subseteq \mathcal{V}$, it holds $\sum_{t=1}^T \operatorname{tr}\left(\Theta_t \Sigma_{t|t}(\mathcal{S}_1)\right) \geq \sum_{t=1}^T \operatorname{tr}\left(\Theta_t \Sigma_{t|t}(\mathcal{S}_2)\right)$.*

Proof Lemma 15 implies $\Sigma_{1|1}(\mathcal{S}_1) \succeq \Sigma_{1|1}(\mathcal{S}_2)$, and then, Corollary 8 implies $\Sigma_{t|t}(\mathcal{S}_1) \succeq \Sigma_{t|t}(\mathcal{S}_2)$. Finally, for any $t = 1, 2, \ldots, T$, Lemma 13 implies $\operatorname{tr}\left(\Theta_t \Sigma_{t|t}(\mathcal{S}_1)\right) \geq \operatorname{tr}\left(\Theta_t \Sigma_{t|t}(\mathcal{S}_2)\right)$, since each Θ_t is symmetric. ■

8.7.2. Proof of Theorem 17

We first prove part (1) of Theorem 17 (Appendix B.1), and then we prove part (2) of Theorem 17 (Appendix B.2).

B.1. Proof of part (1) of Theorem 17

We use the following lemma to prove Theorem 17's part (1).

Lemma 19. *For any active sensor set $\mathcal{S} \subseteq \mathcal{V}$, and any admissible control policy $u_{1:T}(\mathcal{S}) \triangleq \{u_1(\mathcal{S}), u_2(\mathcal{S}), \ldots, u_T(\mathcal{S})\}$, let $h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ be Problem 2's cost function, i.e.,*

$$h[\mathcal{S}, u_{1:T}(\mathcal{S})] \triangleq \sum_{t=1}^{T} \mathbb{E}(\|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2).$$

Further define the following set-valued function:

$$g(\mathcal{S}) \triangleq \min_{u_{1:T}(\mathcal{S})} h[\mathcal{S}, u_{1:T}(\mathcal{S})].$$

Consider any sensor set $\mathcal{S} \subseteq \mathcal{V}$, and let $u_{1:T}^{\star}(\mathcal{S})$ be the vector of control policies $(K_1\hat{x}_1(\mathcal{S}), K_2\hat{x}_2(\mathcal{S}), \ldots, K_T\hat{x}_T(\mathcal{S}))$. Then $u_{1:T}^{\star}(\mathcal{S})$ is an optimal control policy:

$$u_{1:T}^{\star}(\mathcal{S}) \in \operatorname*{argmin}_{u_{1:T}(\mathcal{S})} h[\mathcal{S}, u_{1:T}(\mathcal{S})], \tag{8.28}$$

i.e., $g(\mathcal{S}) = h[\mathcal{S}, u_{1:T}^{\star}(\mathcal{S})]$, and in particular, $u_{1:T}^{\star}(\mathcal{S})$ attains a (sensor-dependent) LQG cost equal to:

$$g(\mathcal{S}) = \mathbb{E}(\|x_1\|_{N_1}) + \sum_{t=1}^{T} \left\{ tr[\Theta_t \Sigma_{t|t}(\mathcal{S})] + \operatorname{tr}(W_t S_t) \right\}. \tag{8.29}$$

Proof of Lemma 19 Let $h_t[\mathcal{S}, u_{t:T}(\mathcal{S})]$ be the LQG cost in Problem 2 from time t up to time T, i.e.,

$$h_t[\mathcal{S}, u_{t:T}(\mathcal{S})] \triangleq \sum_{k=t}^{T} \mathbb{E}(\|x_{k+1}(\mathcal{S})\|_{Q_t}^2 + \|u_k(\mathcal{S})\|_{R_t}^2).$$

and define $g_t(\mathcal{S}) \triangleq \min_{u_{t:T}(\mathcal{S})} h_t[\mathcal{S}, u_{t:T}(\mathcal{S})]$. Clearly, $g_1(\mathcal{S})$ matches the LQG cost in eq. (8.29).

We complete the proof inductively. In particular, we first prove Lemma 19 for $t = T$, and then for any other $t \in \{1, 2, \ldots, T-1\}$. To this end, we use the following observation: given any sensor set $\mathcal{S}$, and any time $t \in \{1, 2, \ldots, T\}$,

$$g_t(\mathcal{S}) = \min_{u_t(\mathcal{S})} \left[\mathbb{E}(\|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2) + g_{t+1}(\mathcal{S}) \right], \tag{8.30}$$

with boundary condition $g_{T+1}(\mathcal{S}) = 0$. In particular, eq. (8.30) holds since

$$g_t(\mathcal{S}) = \min_{u_t(\mathcal{S})} \mathbb{E} \left\{ \|x_{t+1}(\mathcal{S})\|_{Q_t}^2 + \|u_t(\mathcal{S})\|_{R_t}^2) + \right.$$
$$\left. \min_{u_{t+1:T}(\mathcal{S})} h_{t+1}[\mathcal{S}, u_{t+1:T}(\mathcal{S})] \right\},$$

where one can easily recognize the second summand to match the definition of $g_{t+1}(\mathcal{S})$.

We prove Lemma 19 for $t = T$. From eq. (8.30), for $t = T$,

$$\begin{aligned} g_T(\mathcal{S}) &= \min_{u_T(\mathcal{S})} \left[\mathbb{E}(\|x_{T+1}(\mathcal{S})\|^2_{Q_T} + \|u_T(\mathcal{S})\|^2_{R_T})\right] \\ &= \min_{u_T(\mathcal{S})} \left[\mathbb{E}(\|A_T x_T + B_T u_T(\mathcal{S}) + w_T\|^2_{Q_T} + \right. \\ &\quad \left. \|u_T(\mathcal{S})\|^2_{R_T})\right], \end{aligned} \tag{8.31}$$

since $x_{T+1}(\mathcal{S}) = A_T x_T + B_T u_T(\mathcal{S}) + w_T$, as per eq. (8.1); we note that for notational simplicity we drop henceforth the dependency of x_T on $\mathcal{S}$ since x_T is independent of $u_T(\mathcal{S})$, which is the variable under optimization in the optimization problem (8.31). Developing eq. (8.31) we get:

$$\begin{aligned} & g_T(\mathcal{S}) \\ = \; & \min_{u_T(\mathcal{S})} \Big[\mathbb{E}(u_T(\mathcal{S})^\top B_T^\top Q_T B_T u_T(\mathcal{S}) + w_T^\top Q_T w_T + \\ & \qquad x_T^\top A_T^\top Q_T A_T x_T + 2x_T^\top A_T^\top Q_T B_T u_T(\mathcal{S}) + \\ & \quad 2x_T^\top A_T^\top Q_T w_T + 2u_T(\mathcal{S})^\top B_T^\top Q_T w_T + \|u_T(\mathcal{S})\|^2_{R_T})\Big] \\ = \; & \min_{u_T(\mathcal{S})} \Big[\mathbb{E}(u_T(\mathcal{S})^\top B_T^\top Q_T B_T u_T(\mathcal{S}) + \|w_T\|^2_{Q_T} + \\ & \quad x_T^\top A_T^\top Q_T A_T x_T + 2x_T^\top A_T^\top Q_T B_T u_T(\mathcal{S}) + \|u_T\|^2_{R_T})\Big], \end{aligned} \tag{8.32}$$

where the latter equality holds since w_T has zero mean and w_T, x_T, and $u_T(\mathcal{S})$ are independent. From eq. (8.32), rearranging the terms, and using the notation in eq. (8.11),

$$g_T(\mathcal{S}) \tag{8.33}$$

$$= \min_{u_T(\mathcal{S})} \Big[\mathbb{E}(u_T(\mathcal{S})^\top (B_T^\top Q_T B_T + R_T) u_T(\mathcal{S}) + \tag{8.34}$$

$$\|w_T\|^2_{Q_T} + x_T^\top A_T^\top Q_T A_T x_T + 2x_T^\top A_T^\top Q_T B_T u_T(\mathcal{S})\Big] \tag{8.35}$$

$$= \min_{u_T(\mathcal{S})} \Big[\mathbb{E}(\|u_T(\mathcal{S})\|^2_{M_T} + \|w_T\|^2_{Q_T} + x_T^\top A_T^\top Q_T A_T x_T + \tag{8.36}$$

$$2x_T^\top A_T^\top Q_T B_T u_T(\mathcal{S})\Big] \tag{8.37}$$

$$= \min_{u_T(\mathcal{S})} \Big[\mathbb{E}(\|u_T(\mathcal{S})\|^2_{M_T} + \|w_T\|^2_{Q_T} + x_T^\top A_T^\top Q_T A_T x_T - \tag{8.38}$$

$$2x_T^\top (-A_T^\top Q_T B_T M_T^{-1}) M_T u_T(\mathcal{S})\Big] \tag{8.39}$$

$$= \min_{u_T(\mathcal{S})} \Big[\mathbb{E}(\|u_T(\mathcal{S})\|^2_{M_T} + \|w_T\|^2_{Q_T} + x_T^\top A_T^\top Q_T A_T x_T - \tag{8.40}$$

$$2x_T^\top K_T^\top M_T u_T(\mathcal{S})\Big] \tag{8.41}$$

$$\tag{8.42}$$

$$\overset{(i)}{=}\min_{u_T(\mathcal{S})}\left[\mathbb{E}(\|u_T(\mathcal{S})-K_Tx_T\|^2_{M_T}+\|w_T\|^2_{Q_T}+\right. \tag{8.43}$$

$$\left. x_T^\top(A_T^\top Q_T A_T - K_T^\top M_T K_T)x_T\right] \tag{8.44}$$

$$=\min_{u_T(\mathcal{S})}\left(\mathbb{E}(\|u_T(\mathcal{S})-K_Tx_T\|^2_{M_T}+\|w_T\|^2_{Q_T}+\right. \tag{8.45}$$

$$\left. x_T^\top(A_T^\top Q_T A_T - \Theta_T)x_T\right) \tag{8.46}$$

$$=\min_{u_T(\mathcal{S})}\left[\mathbb{E}(\|u_T(\mathcal{S})-K_Tx_T\|^2_{M_T}+\|w_T\|^2_{Q_T}+\|x_T\|^2_{N_T}\right] \tag{8.47}$$

$$\overset{(ii)}{=}\min_{u_T(\mathcal{S})}\mathbb{E}(\|u_T(\mathcal{S})-K_Tx_T\|^2_{M_T})+\operatorname{tr}\left(W_TQ_T\right)+ \tag{8.48}$$

$$\mathbb{E}(\|x_T\|^2_{N_T}), \tag{8.49}$$

where equality (i) follows from completion of squares, and equality (ii) holds since $\mathbb{E}(\|w_T\|^2_{Q_T}) = \mathbb{E}\left[\operatorname{tr}\left(w_T^\top Q_T w_T\right)\right] = \operatorname{tr}\left(\mathbb{E}(w_T^\top w_T)Q_T\right) = \operatorname{tr}\left(W_TQ_T\right)$. Now we note that

$$\begin{aligned}&\min_{u_T(\mathcal{S})}\mathbb{E}(\|u_T(\mathcal{S})-K_Tx_T\|^2_{M_T})\\ &=\mathbb{E}(\|K_T\hat{x}_T(\mathcal{S})-K_Tx_T\|^2_{M_T})\\ &=\operatorname{tr}\left(\Theta_T\Sigma_{T|T}(\mathcal{S})\right),\end{aligned} \tag{8.50}$$

since $\hat{x}_T(\mathcal{S})$ is the Kalman estimator of the state x_T, i.e., the minimum mean square estimator of x_T, which implies that $K_T\hat{x}_T(\mathcal{S})$ is the minimum mean square estimator of $K_Tx_T(\mathcal{S})$ [123, Appendix E]. Substituting (8.50) back into eq. (8.49), we get:

$$g_T(\mathcal{S}) = \mathbb{E}(\|x_T\|^2_{N_T}) + \operatorname{tr}\left(\Theta_T\Sigma_{T|T}(\mathcal{S})\right) + \operatorname{tr}\left(W_TQ_T\right),$$

which proves that Lemma 19 holds for $t = T$.

We now prove that if Lemma 19 holds for $t = l+1$, it also holds for $t = l$. To this end, assume eq. (8.30) holds for $t = l+1$. Using the notation in eq. (8.11),

$$\begin{aligned}g_l(\mathcal{S})=&\min_{u_l(\mathcal{S})}\left[\mathbb{E}(\|x_{l+1}(\mathcal{S})\|^2_{Q_l}+\|u_l(\mathcal{S})\|^2_{R_l})+g_{l+1}(\mathcal{S})\right]\\ =&\quad\min_{u_l(\mathcal{S})}\left\{\mathbb{E}(\|x_{l+1}(\mathcal{S})\|^2_{Q_l}+\|u_l(\mathcal{S})\|^2_{R_l})+\right.\\ &\quad\mathbb{E}(\|x_{l+1}(\mathcal{S})\|^2_{N_{l+1}})+\sum_{k=l+1}^{T}\left[\operatorname{tr}\left(\Theta_k\Sigma_{k|k}(\mathcal{S})\right)+\right.\\ &\quad\left.\left.\operatorname{tr}\left(W_kS_k\right)\right]\right\}\\ =&\quad\min_{u_l(\mathcal{S})}\left\{\mathbb{E}(\|x_{l+1}(\mathcal{S})\|^2_{S_l}+\|u_l(\mathcal{S})\|^2_{R_l})+\right.\\ &\quad\left.\sum_{k=l+1}^{T}\left[\operatorname{tr}\left(\Theta_k\Sigma_{k|k}(\mathcal{S})\right)+\operatorname{tr}\left(W_kS_k\right)\right]\right\}\\ =&\quad\sum_{k=l+1}^{T}\left[\operatorname{tr}\left(\Theta_k\Sigma_{k|k}(\mathcal{S})\right)+\operatorname{tr}\left(W_kS_k\right)\right]+\\ &\quad\min_{u_l(\mathcal{S})}\mathbb{E}(\|x_{l+1}(\mathcal{S})\|^2_{S_l}+\|u_l(\mathcal{S})\|^2_{R_l}).\end{aligned} \tag{8.51}$$

In eq. (8.51), the minimization in the last summand can be solved by following the same

steps as for the proof of Lemma 19 for $t = T$, leading to:

$$\begin{aligned} &\min_{u_l(\mathcal{S})} \mathbb{E}(\|x_{l+1}(\mathcal{S})\|_{S_l}^2 + \|u_l(\mathcal{S})\|_{R_l}^2) = \\ &\mathbb{E}(\|x_l\|_{N_l}^2) + \operatorname{tr}\left(\Theta_l \Sigma_{l|l}(\mathcal{S})\right) + \operatorname{tr}\left(W_l Q_l\right), \end{aligned} \tag{8.52}$$

and $u_l(\mathcal{S}) = K_l \hat{x}_l(\mathcal{S})$. Therefore, by substituting eq. (8.52) back into eq. (8.51), we get:

$$g_l(\mathcal{S}) = \mathbb{E}(\|x_l\|_{N_l}^2) + \sum_{k=l}^{T} [\operatorname{tr}\left(\Theta_k \Sigma_{k|k}(\mathcal{S})\right) + \operatorname{tr}\left(W_k S_k\right)]. \tag{8.53}$$

which proves that if Lemma 19 holds for $t = l+1$, it also holds for $t = l$. By induction, this also proves that Lemma 19 holds for $l = 1$, and we already observed that $g_1(\mathcal{S})$ matches the original LQG cost in eq. (8.29), hence concluding the proof. ■

Proof of Theorem 17's part (1) The proof follows from Lemma 19. In particular, eq. (8.12) is a direct consequence of eq. (8.29), since the value of Problem 2 is equal to $\min_{\mathcal{S} \subseteq \mathcal{V}, c(\mathcal{S}) \leq b} g(\mathcal{S})$, and both $\mathbb{E}(\|x_1\|_{N_1}) = \operatorname{tr}\left(\Sigma_{1|0} N_1\right)$ and $\sum_{t=1}^{T} \operatorname{tr}\left(W_t S_t\right)$ in eq. (8.29) are independent of the choice of the sensor set $\mathcal{S}$. Finally, eq. (8.13) directly follows from eq. (8.28). ■

B.1. Proof of part (2) of Theorem 17

We use the following lemma to prove Theorem 17's part (2).

Lemma 20. *The sensor set $\mathcal{S}^\star$ and the controllers $u_{1:T}^\star$ are a solution to Problem 3 if and only if they are a solution to the optimization problem*

$$\min_{\mathcal{S} \subseteq \mathcal{V}, u_{1:T}(\mathcal{S})} c(\mathcal{S}), \quad \text{s.t.} \quad g(\mathcal{S}) \leq \kappa, \tag{8.54}$$

where

$$g(\mathcal{S}) \triangleq \min_{u_{1:T}(\mathcal{S})} h\left[\mathcal{S}, u_{1:T}(\mathcal{S})\right].$$

Proof of Lemma 20 We prove the lemma by contradiction. In particular, first we prove that if the sensor set $\mathcal{S}^\star$ and the controllers $u_{1:T}^\star$ are a solution to Problem 3, then they are also a solution to the problem in eq. (8.54); and second, we prove that if $\mathcal{S}^\star$ and $u_{1:T}^\star$ are a solution to the problem in eq. (8.54), then they are also a solution to Problem 3.

We prove that if the sensor set $\mathcal{S}^\star$ and the controllers $u_{1:T}^\star$ are a solution to Problem 3, then they are also a solution to the problem in eq. (8.54). To this end, let the sensor set $\mathcal{S}^\star$ and the controllers $u_{1:T}^\star$ be a solution to Problem 3, and assume by contradiction that they are not a solution to the problem in eq. (8.54), which instead has solution $\widehat{\mathcal{S}}$ and $\widehat{u}_{1:T}$. By optimality of $\widehat{\mathcal{S}}$ and $\widehat{u}_{1:T}$ (and suboptimality of $\mathcal{S}^\star$ and $u_{1:T}^\star$) in eq. (8.54), it follows $c(\widehat{\mathcal{S}}) < c(\mathcal{S}^\star)$. In addition, it also holds $g(\widehat{\mathcal{S}}) \leq \kappa$, since $(\widehat{\mathcal{S}}, \widehat{u}_{1:T})$ must be feasible for the problem in eq. (8.54). However, the latter implies that $h\left(\widehat{\mathcal{S}}, \widehat{u}_{1:T}\right) \leq \kappa$. Therefore, $(\widehat{\mathcal{S}}, \widehat{u}_{1:T})$ is feasible for Problem 3 and has a better objective value with respect to the optimal solution $(\mathcal{S}^\star, u_{1:T}^\star)$ (we already observed $c(\widehat{\mathcal{S}}) < c(\mathcal{S}^\star)$), leading to contradiction. Hence, if the sensor set $\mathcal{S}^\star$ and the controllers $u_{1:T}^\star$ are a solution to Problem 3, then they are also a solution to (8.54).

We now prove that if the sensor set $\mathcal{S}^\star$ and the controllers $u^\star_{1:T}$ are a solution to the problem in eq. (8.54), then they are also a solution to Problem 3. To this end, let the sensor set $\mathcal{S}^\star$ and the controllers $u^\star_{1:T}$ be a solution to the problem in eq. (8.54), and assume that they are not a solution to Problem 3, which instead has solution $(\widehat{\mathcal{S}}, \widehat{u}_{1:T})$. By optimality of $(\widehat{\mathcal{S}}, \widehat{u}_{1:T})$ (and suboptimality of $\mathcal{S}^\star$ and $u^\star_{1:T}$) in Problem 3 , it follows $c(\widehat{\mathcal{S}}) < c(\mathcal{S}^\star)$. In addition, it is $h\left(\widehat{\mathcal{S}}, \widehat{u}_{1:T}\right) \leq \kappa$, since $(\widehat{\mathcal{S}}, \widehat{u}_{1:T})$ must be feasible for Problem 3, and, as a result, it also holds $g(\widehat{\mathcal{S}}) \leq \kappa$. Therefore, $(\widehat{\mathcal{S}}, \widehat{u}_{1:T})$ is feasible for the problem in eq. (8.54) and has a better objective value with respect to the optimal solution $(\mathcal{S}^\star, u^\star_{1:T})$ (we already observed $c(\widehat{\mathcal{S}}) < c(\mathcal{S}^\star)$), leading to contradiction. Hence, if the sensor set $\mathcal{S}^\star$ and the controllers $u^\star_{1:T}$ are a solution to the problem in eq. (8.54), then they are also a solution to Problem 3. ■

Proof of Theorem 17's part (2) The proof follows from Lemma 19 and Lemma 20. In particular, similarly to the proof of Theorem 17's part (1), Lemma 19, along with eq. (8.29) and the fact that $\mathbb{E}(\|x_1\|_{N_1}) = \operatorname{tr}\left(\Sigma_{1|0} N_1\right)$, implies that if the sensor set $\mathcal{S}^\star$ and the controllers $u^\star_{1:T}$ are a solution to the optimization problem in eq. (8.54), then $\mathcal{S}^\star$ and the controllers $u^\star_{1:T}$ can be computed in cascade as follows:

$$\mathcal{S}^\star \in \operatorname*{argmin}_{\mathcal{S}\subseteq\mathcal{V}} c(\mathcal{S}), \quad \text{s.t.} \quad \operatorname{tr}[\Theta_t \Sigma_{t|t}(\mathcal{S})] \leq$$

$$\kappa - \operatorname{tr}\left(\Sigma_{1|0} N_1\right) - \sum_{t=1}^{T} \operatorname{tr}\left(W_t S_t\right), \tag{8.55}$$

$$u^\star_t = K_t \hat{x}_t(\mathcal{S}^\star), \quad t = 1, \ldots, T. \tag{8.56}$$

In addition, Lemma 20 implies that $(\mathcal{S}^\star, u^\star_{1:T})$ is a solution to Problem 3. As a result, eqs. (8.14)-(8.15) hold true. ■

8.7.3. Proof of Theorem 18 and Proposition 10

We first prove Theorem 18 (Appendix C.1), and then prove Proposition 10 (Appendix C.2).

C.1. Proof of Theorem 18

We consider the following notation: for any sensor set $\mathcal{S} \subseteq \mathcal{V}$, we let $f(\mathcal{S}) \triangleq \sum_{t=1}^{T} \operatorname{tr}[\Theta_t \Sigma_{t|t}(\mathcal{S})]$ be the cost function in eq. (8.12), $\mathcal{S}^\star$ be a solution in eq. (8.12), and $b^\star \triangleq c(\mathcal{S}^\star)$, that is, $b^\star$ is the cost of the sensor set $\mathcal{S}^\star$. In addition, consider the computation of the set $\widehat{\mathcal{S}}_2$ in Algorithm 16 (lines 3-19), where $\widehat{\mathcal{S}}_2$ refers to the set that Algorithm 16 has constructed by the end of the line 19; we let $\mathcal{G} \triangleq \widehat{\mathcal{S}}_2$. We also let s_i be the i-th element added in $\mathcal{G}$ during the i-th iteration of Algorithm 16's "while loop" (lines 3-16). Moreover, we let $\mathcal{G}_i \triangleq \{s_1, s_2, \ldots, s_i\}$, that is, $\mathcal{G}_i$ is the subset of $\mathcal{G}$ constructed during the first i iterations of Algorithm 16's "while loop" (lines 3-16). Finally, we consider that Algorithm 16's "while loop" (lines 3-16) terminates after $l+1$ iterations.

There are two scenarios under which Algorithm 16's "while loop" (lines 3-16) terminates: (i) the trivial scenario where $\mathcal{V}' = \emptyset$, that is, where all available sensors in $\mathcal{V}$ can been chosen by Algorithm 16 as active while satisfying the budget constraint b; and (ii) the

non-trivial scenario where Algorithm 16's "while" loop (lines 3–16) terminates because $c(\mathcal{G}_{l+1}) > b$, that is, where the addition of the sensor s_{l+1} in $\mathcal{G}_l$ makes the sensor cost of $\mathcal{G}_{l+1}$ violate the budget constraint b. Henceforth, we focus on the second, non-trivial scenario, which implies that s_{l+1} will be removed by the "if" statement in Algorithm 16's lines 17–19 and, as a result, $\mathcal{G}_l = \widehat{\mathcal{S}}_2$.

We prove Theorem 18 using the following two lemmas.

Lemma 21 (Generalization of [44, Lemma 2]). *For $i = 1, 2, \ldots, l+1$, it holds*

$$f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i) \geq \frac{\gamma_f c(s_i)}{b^\star}(f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star)).$$

Proof of Lemma 21 Due to the monotonicity of the cost function f in eq. (8.12) (Proposition 11), it holds

$$\begin{aligned} f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star) &\leq f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star \cup \mathcal{G}_{i-1}) \\ &= f(\mathcal{G}_{i-1}) - f[(\mathcal{S}^\star \setminus \mathcal{G}_{i-1}) \cup \mathcal{G}_{i-1}]. \end{aligned}$$

Let $\{z_1, z_2, \ldots, z_m\} \triangleq \mathcal{S}^\star \setminus \mathcal{G}_{i-1}$, and also let

$$d_j \triangleq f(\mathcal{G}_{i-1} \cup \{z_1, z_2, \ldots, z_{j-1}\}) - f(\mathcal{G}_{i-1} \cup \{z_1, z_2, \ldots, z_j\}),$$

for $j = 1, 2, \ldots, m$. Then, $f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star) \leq \sum_{j=1}^m d_j$.

Notice that

$$\frac{d_j}{c(z_j)} \leq \frac{f(\mathcal{G}_{i-1}) - f(\mathcal{G}_{i-1} \cup \{z_j\})}{\gamma_f c(z_j)} \leq \frac{f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i)}{\gamma_f c(s_i)},$$

where the first inequality holds due to the Definition 29 of the supermodularity ratio γ_f, and the second inequality holds due to the greedy rule (Algorithm 16's line 13) and the definitions of $\mathcal{G}_i$, and s_i. Since $\sum_{j=1}^m c(z_j) \leq b^\star$, it holds that

$$f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star) \leq \sum_{j=1}^m d_j \leq b^\star \frac{f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i)}{\gamma_f c(s_i)}.$$

■

Lemma 22 (Adapation of [44, Lemma 3]). *For $i = 1, 2, \ldots, l+1$, it holds*

$$f(\emptyset) - f(G_i) \geq \left[1 - \prod_{j=1}^{i}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right][f(\emptyset) - f(\mathcal{S}^\star)].$$

Proof of Lemma 22 We complete the proof inductively. In particular, for $i = 1$, we need to prove $f(\emptyset) - f(G_1) \geq \gamma_f c(s_1)/b^\star[f(\emptyset) - f(\mathcal{S}^\star)]$, which follows from Lemma 21 for $i = 1$.

Then, we have for $i > 1$:

$$\begin{aligned}
f(\emptyset) - f(\mathcal{G}_i) &= f(\emptyset) - f(\mathcal{G}_{i-1}) + [f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i)] \\
&\geq f(\emptyset) - f(\mathcal{G}_{i-1}) + \\
&\quad \frac{\gamma_f c(s_i)}{b^\star}(f(G_{i-1}) - f(\mathcal{S}^\star)) \\
&= \left(1 - \frac{\gamma_f c(s_i)}{b^\star}\right)[f(\emptyset) - f(\mathcal{G}_{i-1}])+ \\
&\quad \frac{\gamma_f c(s_i)}{b^\star}[f(\emptyset) - f(\mathcal{S}^\star)] \\
&\geq \left(1 - \frac{\gamma_f c(s_i)}{b^\star}\right)\left[1 - \prod_{j=1}^{i-1}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right] \\
&\quad [f(\emptyset) - f(\mathcal{S}^\star)] + \frac{\gamma_f c(s_i)}{b^\star}[f(\emptyset) - f(\mathcal{S}^\star)] \\
&= \left[1 - \prod_{j=1}^{i}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right][f(\emptyset) - f(\mathcal{S}^\star)],
\end{aligned}$$

where we used Lemma 21 for the first inequality and the induction hypothesis for the second inequality. ■

Proof of Theorem 18's part (1) (Algorithm 15's approximation quality) We first prove the approximation bound $\frac{\gamma_g}{2}(1 - e^{-\gamma_g})$ in ineq. (8.19) and, then, the bound $1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b^\star}$.

To prove Algorithm 15's approximation bound $\frac{\gamma_g}{2}(1 - e^{-\gamma_g})$ in ineq. (8.19), we let $b' \triangleq \sum_{j=1}^{l+1} c(s_j)$. It holds

$$\begin{aligned}
f(\emptyset) - f(\mathcal{G}_{l+1}) &\geq \left[1 - \prod_{j=1}^{l+1}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right][f(\emptyset) - f(\mathcal{S}^\star)] \\
&\geq \left(1 - e^{-\gamma_f b'/b^\star}\right)[f(\emptyset) - f(\mathcal{S}^\star)], \\
&\geq \left(1 - e^{-\gamma_f}\right)[f(\emptyset) - f(\mathcal{S}^\star)], \qquad (8.57)
\end{aligned}$$

where the first inequality follows from Lemma 22, the second inequality from Lemma 18, and ineq. (8.57) from the fact that $b'/b^\star \geq 1$ and, as a result, $e^{-\gamma_f b'/b^\star} \leq e^{-\gamma_f}$, that is, $1 - e^{-\gamma_f b'/b^\star} \geq 1 - e^{-\gamma_f}$.

In addition, it is $f(\emptyset) - f(\widehat{\mathcal{S}}_1) \geq \gamma_f[f(\mathcal{G}_l) - f(\mathcal{G}_{l+1})]$ due to the Definition 29 of the supermodularity ratio and, as a result, $f(\emptyset) - f(\widehat{\mathcal{S}}_1) \geq \gamma_f[f(\mathcal{G}_l) - f(\emptyset) + f(\emptyset) - f(\mathcal{G}_{l+1})]$, which

after rearranging its terms gives

$$\begin{aligned}
&\gamma_f[f(\emptyset) - f(\mathcal{G}_{l+1})] \\
&\quad \leq f(\emptyset) - f(\widehat{\mathcal{S}}_1) + \gamma_f[f(\emptyset) - f(\mathcal{G}_l)] \\
&\quad \leq 2\max\left\{f(\emptyset) - f(\widehat{\mathcal{S}}_1), \gamma_f[f(\emptyset) - f(\mathcal{G}_l)]\right\}. \qquad (8.58)
\end{aligned}$$

By substituting ineq. (8.57) in ineq. (8.58) and rearranging the terms we have

$$\max\left\{f(\emptyset) - f(\widehat{\mathcal{S}}_1), \gamma_f[f(\emptyset) - f(\mathcal{G}_l)]\right\}$$

which implies the inequality$\geq \frac{\gamma_f}{2}\left(1 - e^{-\gamma_f}\right)[f(\emptyset) - f(\mathcal{S}^\star)]$,

$$\begin{aligned}
&\max\left[f(\emptyset) - f(\widehat{\mathcal{S}}_1), f(\emptyset) - f(\mathcal{G}_l)\right] \\
&\quad \geq \frac{\gamma_f}{2}\left(1 - e^{-\gamma_f}\right)[f(\emptyset) - f(\mathcal{S}^\star)], \qquad (8.59)
\end{aligned}$$

since γ_f takes values in $[0, 1]$ by the Definition 29 of the supermodularity ratio.

Algorithm 15's approximation bound $\frac{\gamma_g}{2}\left(1 - e^{-\gamma_g}\right)$ in ineq. (8.19) follows from ineq. (8.59) as the combination of the following three observations:

- it is $\mathcal{G}_l = \widehat{\mathcal{S}}_2$, due to the definition of $\mathcal{G}_l$, and, as a result, $f(\emptyset) - f(\mathcal{G}_l) = f(\emptyset) - f(\widehat{\mathcal{S}}_2)$.
- Algorithm 16 returns the set $\widehat{\mathcal{S}}$ such at $\widehat{\mathcal{S}} \in \arg\max_{\mathcal{S} \in \{\widehat{\mathcal{S}}_1, \widehat{\mathcal{S}}_2\}}[f(\emptyset) - f(\mathcal{S})]$ (per Algorithm 16's line 20) and, as a result, the previous observation, along with ineq. (8.59), gives:
$$f(\emptyset) - f(\widehat{\mathcal{S}}) \geq \frac{\gamma_f}{2}\left(1 - e^{-\gamma_f}\right)[f(\emptyset) - f(\mathcal{S}^\star)]. \qquad (8.60)$$
- Finally, Lemma 19 implies that for any sets $\mathcal{S}, \mathcal{S}' \subseteq \mathcal{V}$ it is $g(\mathcal{S}) = f(\mathcal{S}) + \mathbb{E}(\|x_1\|_{N_1}) + \sum_{t=1}^{T} \mathrm{tr}\left(W_t S_t\right)$, where $\mathbb{E}(\|x_1\|_{N_1}) + \sum_{t=1}^{T} \mathrm{tr}\left(W_t S_t\right)$ is independent of $\mathcal{S}$. As a result, for any sets $\mathcal{S}, \mathcal{S}' \subseteq \mathcal{V}$ it is $f(\mathcal{S}) - f(\mathcal{S}') = g(\mathcal{S}) - g(\mathcal{S}')$, which implies $\gamma_f = \gamma_g$ due to the Definition 29 of the supermodularity ratio. In addition, Lemma 19 implies that for any $\mathcal{S} \subseteq \mathcal{V}$ it is $g(\mathcal{S}) = h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ and $g^\star = g(\mathcal{S}^\star)$. Thereby, for any $\mathcal{S} \subseteq \mathcal{V}$ it is $f(\emptyset) - f(\mathcal{S}) = g(\emptyset) - g(\mathcal{S}) = h[\emptyset, u_{1:T}(\emptyset)] - h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ and $f(\emptyset) - f(\mathcal{S}^\star) = g(\emptyset) - g(\mathcal{S}^\star) = h[\emptyset, u_{1:T}(\emptyset)] - g^\star$. Overall, ineq. (8.60) is written as
$$\begin{aligned}
&h[\emptyset, u_{1:T}(\emptyset)] - h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})] \geq \\
&\qquad \frac{\gamma_f}{2}\left(1 - e^{-\gamma_f}\right)\left\{h[\emptyset, u_{1:T}(\emptyset)] - g^\star\right\},
\end{aligned}$$
which implies the bound $\frac{\gamma_g}{2}\left(1 - e^{-\gamma_g}\right)$ in ineq. (8.19).

It remains to prove the bound $1 - e^{-\gamma_g c(\widehat{\mathcal{S}})/b}$ in ineq. (8.19). To this end, we first have:

$$\begin{aligned} f(\emptyset) - f(\mathcal{G}_l) &\geq \left[1 - \prod_{j=1}^{l} \left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right] [f(\emptyset) - f(\mathcal{G}_l)] \\ &\geq \left(1 - e^{-\gamma_f c(\mathcal{G}_l)/b^\star}\right) [f(\emptyset) - f(\mathcal{S}^\star)], \\ &\geq \left(1 - e^{-\gamma_f c(\mathcal{G}_l)/b}\right) [f(\emptyset) - f(\mathcal{S}^\star)], \end{aligned} \tag{8.61}$$

where the first inequality follows from Lemma 22, the second inequality from Lemma 18, and ineq. (8.61) from the fact that $c(\mathcal{G}_l)/b^\star \geq c(\mathcal{G}_l)/b$, since $b^\star \leq b$, which implies $e^{-\gamma_f c(\mathcal{G}_l)/b^\star} \leq e^{-\gamma_f c(\mathcal{G}_l)/b}$, i.e., $1 - e^{-\gamma_f b'/b^\star} \geq 1 - e^{-\gamma_f c(\mathcal{G}_l)/b}$. The rest of the proof is completed using the combination of the three observations in the previous paragraph, that we used to prove Algorithm 15's approximation bound $\frac{\gamma_g}{2}\left(1 - e^{-\gamma_g}\right)$ in ineq. (8.19). ■

Proof of Theorem 18's part (2) (Algorithm 15's running time) We compute Algorithm 15's running time by adding the running times of Algorithm 15's lines 1-5:

Running time of Algorithm 15's line 1 Algorithm 15's line 1 needs $O(Tn^{2.4})$ time, using the Coppersmith algorithm for both matrix inversion and multiplication [216].

Running time of Algorithm 15's line 2 Algorithm 15's line 2 running time is the running time of Algorithm 16, whose running time we show next to be $O(|\mathcal{V}|^2 Tn^{2.4})$. To this end, we first compute the running time of Algorithm 16's line 1, and finally the running time of Algorithm 16's lines 3–16. Algorithm 16's lines 3–16 are repeated at most $|\mathcal{V}|^2$ times, since before the end of each iteration of the "while loop" in line 3 the added element in $\widehat{\mathcal{S}}_2$ (line 14) is removed from $\mathcal{V}'$ (line 15). We now need to find the running time of Algorithm 16's lines 4–16; to this end, we first find the running time of Algorithm 16's lines 4–12, and then the running time of Algorithm 16's line 13. In more detail, the running time of Algorithm 16's lines 4–12 is $O(|\mathcal{V}|Tn^{2.4})$, since Algorithm 16's lines 4–12 are repeated at most $|\mathcal{V}|$ times and Algorithm 16's lines 5–10, as well as, line 11 need $O(Tn^{2.4})$ time, using the Coppersmith-Winograd algorithm for both matrix inversion and multiplication [216]. Moreover, Algorithm 16's line 13 needs $O[|\mathcal{V}|\log(|\mathcal{V}|)]$ time, since it asks for the maximum among at most $|\mathcal{V}|$ values of the $\text{gain}_{(\cdot)}$, which takes $O[|\mathcal{V}|\log(|\mathcal{V}|)]$ time to be found, using, e.g., the merge sort algorithm. In sum, Algorithm 16's running time is upper bounded by $O[|\mathcal{V}|^2 Tn^{2.4} + |\mathcal{V}|^2 \log(|\mathcal{V}|)]$, which is equal to $O(|\mathcal{V}|^2 Tn^{2.4})$.

Running time of Algorithm 15's lines 3-5 Algorithm 15's lines 3-5 need $O(Tn^{2.4})$ time, using the Coppersmith algorithm for both matrix inversion and multiplication [216].

In sum, Algorithm 15's running time is upper bounded by $O(|\mathcal{V}|^2 Tn^{2.4} + 2Tn^{2.4})$ which is equal to $O(|\mathcal{V}|^2 Tn^{2.4})$. ■

C.2. Proof of Proposition 10

The proof of Proposition 10 follows the same steps as the proof of Theorem 18 and for this

reason we omit it.

8.7.4. Proof of Theorem 19

We consider the following notation: for any sensor set $\mathcal{S} \subseteq \mathcal{V}$, we let $f(\mathcal{S}) \triangleq \sum_{t=1}^{T} \mathrm{tr}[\Theta_t \Sigma_{t|t}(\mathcal{S})]$ be the cost function in eq. (8.14), the sensor set $\mathcal{S}^\star$ be a solution to Problem 3, and $b^\star$ be equal to $c(\mathcal{S}^\star)$, that is, $b^\star$ is the optimal value of Problem 3. In addition, consider the computation of the set $\widehat{\mathcal{S}}$ in Algorithm 18 (lines 3-16): we let $\mathcal{G} \triangleq \widehat{\mathcal{S}}$. We also let s_i be the i-th element added in $\mathcal{G}$ during the i-th iteration of Algorithm 18's "while loop" (lines 3-16). Finally, we let $\mathcal{G}_i \triangleq \{s_1, s_2, \ldots, s_i\}$, that is, $\mathcal{G}_i$ is the subset of $\widehat{\mathcal{S}}$ constructed during the first i iterations of Algorithm 18's "while loop" (lines 3-16).

We prove Theorem 19 using the following two lemmas.

Lemma 23 (Adaptation of Lemma 21). *For $i = 1, 2, \ldots, |\mathcal{G}|$, it holds*

$$f(\mathcal{G}_{i-1}) - f(G_i) \geq \frac{\gamma_f c(s_i)}{b^\star}(f(G_{i-1}) - f(\mathcal{S}^\star)).$$

Proof of Lemma 23 Due to the monotonicity of the cost function f in eq. (8.12) (Proposition 11), it holds

$$\begin{aligned} f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star) &\leq f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star \cup \mathcal{G}_{i-1}) \\ &= f(\mathcal{G}_{i-1}) - f[(\mathcal{S}^\star \setminus \mathcal{G}_{i-1}) \cup \mathcal{G}_{i-1}]. \end{aligned}$$

Let $\{z_1, z_2, \ldots, z_m\} \triangleq \mathcal{S}^\star \setminus \mathcal{G}_{i-1}$, and also let

$$d_j \triangleq f(\mathcal{G}_{i-1} \cup \{z_1, z_2, \ldots, z_{j-1}\}) - f(\mathcal{G}_{i-1} \cup \{z_1, z_2, \ldots, z_j\}),$$

for $j = 1, 2, \ldots, m$. Then, $f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star) \leq \sum_{j=1}^{m} d_j$.

Notice that

$$\frac{d_j}{c(z_j)} \leq \frac{f(\mathcal{G}_{i-1}) - f(\mathcal{G}_{i-1} \cup \{z_j\})}{\gamma_f c(z_j)} \leq \frac{f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i)}{\gamma_f c(s_i)},$$

where the first inequality holds due to the Definition 29 of the supermodularity ratio γ_f, and the second inequality holds due to the greedy rule (Algorithm 18's line 13) and the definitions of $\mathcal{G}_i$ and s_i. Since $\sum_{j=1}^{m} c(z_j) \leq b^\star$, it holds that

$$f(\mathcal{G}_{i-1}) - f(\mathcal{S}^\star) \leq \sum_{j=1}^{m} d_j \leq b^\star \frac{f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i)}{\gamma_f c(s_i)}.$$

■

Lemma 24 (Adaptation of Lemma 22). *For $i = 1, 2, \ldots, |\mathcal{G}|$, it holds*

$$f(\emptyset) - f(G_i) \geq \left[1 - \prod_{j=1}^{i}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right][f(\emptyset) - f(\mathcal{S}^\star)].$$

Proof of Lemma 24 We complete the proof inductively. In particular, for $i = 1$, we need to prove $f(\emptyset) - f(G_1) \geq \gamma_f c(s_1)/b^\star [f(\emptyset) - f(\mathcal{S}^\star)]$, which follows from Lemma 23 for $i = 1$.

Then, we have for $i > 1$:

$$\begin{aligned}
f(\emptyset) - f(\mathcal{G}_i) &= f(\emptyset) - f(\mathcal{G}_{i-1}) + [f(\mathcal{G}_{i-1}) - f(\mathcal{G}_i)] \\
&\geq f(\emptyset) - f(\mathcal{G}_{i-1}) + \\
&\quad \frac{\gamma_f c(s_i)}{b^\star}(f(G_{i-1}) - f(\mathcal{S}^\star)) \\
&= \left(1 - \frac{\gamma_f c(s_i)}{b^\star}\right)[f(\emptyset) - f(\mathcal{G}_{i-1}]) + \\
&\quad \frac{\gamma_f c(s_i)}{b^\star}[f(\emptyset) - f(\mathcal{S}^\star)] \\
&\geq \left(1 - \frac{\gamma_f c(s_i)}{b^\star}\right)\left[1 - \prod_{j=1}^{i-1}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right] \\
&\quad [f(\emptyset) - f(\mathcal{S}^\star)] + \frac{\gamma_f c(s_i)}{b^\star}[f(\emptyset) - f(\mathcal{S}^\star)] \\
&= \left[1 - \prod_{j=1}^{i}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right][f(\emptyset) - f(\mathcal{S}^\star)],
\end{aligned}$$

using Lemma 23 for the first inequality and the induction hypothesis for the second inequality. ■

Proof of Theorem 19's part (1) (Algorithm 17's approximation quality) We first observe that ineq. (8.21) holds since Algorithm 17 returns the set $\widehat{\mathcal{S}}$ once the condition $h[\widehat{\mathcal{S}}, u_{1:T}(\widehat{\mathcal{S}})] \leq \kappa$ is satisfied (Algorithm 18's line 3).

It remains to prove ineq. (8.22). Let $l \triangleq |\mathcal{G}|$; then, $\mathcal{G}_l = \mathcal{G}$, by the definition of $\mathcal{G}_i$, and from Lemma 22 for $i = l - 1$ it holds

$$\begin{aligned}
f(\emptyset) - f(\mathcal{G}_{l-1}) &\geq \left[1 - \prod_{j=1}^{l-1}\left(1 - \frac{\gamma_f c(s_j)}{b^\star}\right)\right][f(\emptyset) - f(\mathcal{S}^\star)] \\
&\geq \left(1 - e^{-\gamma_f c(\mathcal{G}_{l-1})/b^\star}\right)[f(\emptyset) - f(\mathcal{S}^\star)], \qquad (8.62)
\end{aligned}$$

where ineq. (8.62) follows from Lemma 18. Moreover, Lemma 19 implies that for any sensor sets $\mathcal{S}, \mathcal{S}' \subseteq \mathcal{V}$ it is $g(\mathcal{S}) = f(\mathcal{S}) + \mathbb{E}(\|x_1\|_{N_1}) + \sum_{t=1}^{T} \operatorname{tr}(W_t S_t)$, where the term $\mathbb{E}(\|x_1\|_{N_1}) + \sum_{t=1}^{T} \operatorname{tr}(W_t S_t)$ is independent of $\mathcal{S}$, and, as a result, it is $f(\mathcal{S}) - f(\mathcal{S}') = g(\mathcal{S}) - g(\mathcal{S}')$, which implies $\gamma_f = \gamma_g$. Moreover, Lemma 19 implies for any $\mathcal{S} \subseteq \mathcal{V}$ that $g(\mathcal{S}) = h[\mathcal{S}, u_{1:T}(\mathcal{S})]$ and, as a result, it is $f(\emptyset) - f(\mathcal{G}_{l-1}) = h[\emptyset, u_{1:T}(\emptyset)] - h[\mathcal{G}_{l-1}, u_{1:T}(\mathcal{G}_{l-1})]$ and $f(\emptyset) - f(\mathcal{S}^\star) = h[\emptyset, u_{1:T}(\emptyset)] - h[\mathcal{S}^\star, u_{1:T}(\mathcal{S}^\star)]$. In sum, ineq. (8.62) is the same as the inequality

$$\begin{aligned}
&h[\emptyset, u_{1:T}(\emptyset)] - h[\mathcal{G}_{l-1}, u_{1:T}(\mathcal{G}_{l-1})] \geq \\
&\quad \left(1 - e^{-\gamma_g c(\mathcal{G}_{l-1})/b^\star}\right)\{h[\emptyset, u_{1:T}(\emptyset)] - h[\mathcal{S}^\star, u_{1:T}(\mathcal{S}^\star)]\},
\end{aligned}$$

which, by letting $\beta \triangleq 1 - e^{-\gamma_g c(\mathcal{G}_{l-1})/b^\star}$ and rearranging its terms, is simplified to the

inequality

$$h[\mathcal{G}_{l-1}, u_{1:T}(\mathcal{G}_{l-1})] \leq (1-\beta)h[\emptyset, u_{1:T}(\emptyset)] + \beta h[\mathcal{S}^\star, u_{1:T}(\mathcal{S}^\star)]$$
$$\leq (1-\beta)h[\emptyset, u_{1:T}(\emptyset)] + \beta\kappa, \tag{8.63}$$

where the second inequality holds because $\mathcal{S}^\star$ is a solution to Problem 3 and, as result, $h[\mathcal{S}^\star, u_{1:T}(\mathcal{S}^\star)] \leq \kappa$. To complete the proof, we recall that Algorithm 18 returns the set $\mathcal{G} = \mathcal{G}_l$ when for $i = l$ it is the first time that $h[\mathcal{G}_i, u_{1:T}(\mathcal{G}_i)] \leq \kappa$. Therefore, $h[\mathcal{G}_{l-1}, u_{1:T}(\mathcal{G}_{l-1})] > \kappa$ and, as a result, there exists a real number $\epsilon > 0$ such that $h[\mathcal{G}_{l-1}, u_{1:T}(\mathcal{G}_{l-1})] = (1+\epsilon)\kappa$, and ineq. (8.63) gives

$$\begin{aligned}
&(1+\epsilon)\kappa \leq (1-\beta)h[\emptyset, u_{1:T}(\emptyset)] + \beta\kappa \Rightarrow \\
&\epsilon\kappa \leq (1-\beta)h[\emptyset, u_{1:T}(\emptyset)] - (1-\beta)\kappa \Rightarrow \\
&\epsilon\kappa \leq (1-\beta)\{h[\emptyset, u_{1:T}(\emptyset)] - \kappa\} \Rightarrow \\
&\epsilon\kappa \leq e^{-\gamma_g c(\mathcal{G}_{l-1})/b^\star}\{h[\emptyset, u_{1:T}(\emptyset)] - \kappa\} \Rightarrow \\
&\log\left(\frac{\epsilon\kappa}{h[\emptyset, u_{1:T}(\emptyset)] - \kappa}\right) \leq -\gamma_g c(\mathcal{G}_{l-1})/b^\star \Rightarrow \\
&c(\mathcal{G}_{l-1}) \leq \frac{1}{\gamma_g}\log\left(\frac{h[\emptyset, u_{1:T}(\emptyset)] - \kappa}{\epsilon\kappa}\right) b^\star \Rightarrow \\
&c(\mathcal{G}) \leq c(s_l) + \frac{1}{\gamma_g}\log\left(\frac{h[\emptyset, u_{1:T}(\emptyset)] - \kappa}{\epsilon\kappa}\right) b^\star,
\end{aligned}$$

where the latter holds since $\mathcal{G} = \mathcal{G}_{l-1} \cup \{s_l\}$, due to the definitions of $\mathcal{G}$, $\mathcal{G}_{l-1}$, and s_l, and since $c(\mathcal{G}) = c(\mathcal{G}_{l-1}) + c(s_l)$. Finally, since the definition of ϵ implies $\epsilon\kappa = h[\mathcal{G}_{l-1}, u_{1:T}(\mathcal{G}_{l-1})] - \kappa$, and the definition of $\mathcal{G}$ is $\mathcal{G} = \widehat{\mathcal{S}}$, the proof of ineq. (8.21) is complete. ■

Proof of Theorem 19's part (2) (Algorithm 17's running time) The proof is similar to the proof of Theorem 18's part (2) (Algorithm 15's running time) and for this reason we omit it. ■

8.7.5. Proof of Theorem 20

Proof of Theorem 20 We complete the proof by first deriving a lower bound for the numerator of the supermodularity ratio γ_g, and then, by deriving an upper bound for the denominator of the supermodularity ratio γ_g.

We use the following notation: $c \triangleq \mathbb{E}(x_1^\top N_1 x_1) + \sum_{t=1}^T \operatorname{tr}(W_t S_t)$, and for any sensor set $\mathcal{S} \subseteq \mathcal{V}$, and time $t = 1, 2, \ldots, T$, $f_t(\mathcal{S}) \triangleq \operatorname{tr}\left(\Theta_t \Sigma_{t|t}(\mathcal{S})\right)$. Then, the cost function $g(\mathcal{S})$ in eq. (8.18) is written as $g(\mathcal{S}) = c + \sum_{t=1}^T f_t(\mathcal{S})$, due to eq. (8.29) in Lemma 19.

Lower bound for the numerator of the supermodularity ratio γ_g Per the supermodularity ratio Definition 29, the numerator of the submodularity ratio γ_g is of the form

$$\sum_{t=1}^T [f_t(\mathcal{S}) - f_t(\mathcal{S} \cup \{v\})], \tag{8.64}$$

for some sensor set $\mathcal{S} \subseteq \mathcal{V}$, and sensor $v \in \mathcal{V}$; to lower bound the sum in (8.64), we lower bound each $f_t(\mathcal{S}) - f_t(\mathcal{S} \cup \{v\})$. To this end, from eq. (14) in Lemma 14, observe

$$\Sigma_{t|t}(\mathcal{S} \cup \{v\}) = [\Sigma_{t|t-1}^{-1}(\mathcal{S} \cup \{v\}) + \sum_{i \in \mathcal{S} \cup \{v\}} \bar{C}_{i,t}^\top \bar{C}_{i,t}]^{-1}.$$

Define $\Omega_t = \Sigma_{t|t-1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S}}^T \bar{C}_{i,t}^\top \bar{C}_{i,t}$, and $\bar{\Omega}_t = \Sigma_{t|t-1}^{-1}(\mathcal{S} \cup \{v\}) + \sum_{i \in \mathcal{S}}^T \bar{C}_{i,t}^\top \bar{C}_{i,t}$; using the Woodbury identity in Lemma 12,

$$f_t(\mathcal{S} \cup \{v\}) = \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1}\right) - \\ \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{\Omega}_t^{-1}\right).$$

Therefore, for any time $t \in \{1, 2 \ldots, T\}$,

$$\begin{aligned} & f_t(\mathcal{S}) - f_t(\mathcal{S} \cup \{v\}) = \\ & \mathrm{tr}\left(\Theta_t \Omega_t^{-1}\right) - \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1}\right) + \\ & \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{\Omega}_t^{-1}\right) \geq \\ & \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{\Omega}_t^{-1}\right), \end{aligned} \tag{8.65}$$

where ineq. (8.65) holds because $\mathrm{tr}\left(\Theta_t \Omega_t^{-1}\right) \geq \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1}\right)$. In particular, the inequality $\mathrm{tr}\left(\Theta_t \Omega_t^{-1}\right) \geq \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1}\right)$ is implied as follows: Lemma 15 implies $\Sigma_{1|1}(\mathcal{S}) \succeq \Sigma_{1|1}(\mathcal{S} \cup \{v\})$. Then, Corollary 9 implies $\Sigma_{t|t-1}(\mathcal{S}) \succeq \Sigma_{t|t-1}(\mathcal{S} \cup \{v\})$, and as a result, Lemma 10 implies $\Sigma_{t|t-1}(\mathcal{S})^{-1} \preceq \Sigma_{t|t-1}(\mathcal{S} \cup \{v\})^{-1}$. Now, $\Sigma_{t|t-1}(\mathcal{S})^{-1} \preceq \Sigma_{t|t-1}(\mathcal{S} \cup \{v\})^{-1}$ and the definition of Ω_t and of $\bar{\Omega}_t$ imply $\Omega_t \preceq \bar{\Omega}_t$. Next, Lemma 10 implies $\Omega_t^{-1} \succeq \bar{\Omega}_t^{-1}$. As a result, since also Θ_t is a symmetric matrix, Lem- ma 13 gives the desired inequality $\mathrm{tr}\left(\Theta_t \Omega_t^{-1}\right) \geq \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1}\right)$.

Continuing from the ineq. (8.65),

$$\begin{aligned} & f_t(\mathcal{S}) - f_t(\mathcal{S} \cup \{v\}) \geq \\ & \mathrm{tr}\left(\bar{C}_{v,t} \bar{\Omega}_t^{-1} \Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top)^{-1}\right) \geq \\ & \lambda_{\min}((I + \bar{C}_{v,t} \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top)^{-1}) \mathrm{tr}\left(\bar{C}_{v,t} \bar{\Omega}_t^{-1} \Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top\right), \end{aligned} \tag{8.66}$$

where ineq. (8.66) holds due to Lemma 11. From ineq. (8.66),

$$\begin{aligned} & f_t(\mathcal{S}) - f_t(\mathcal{S} \cup \{v\}) \geq \\ & = \lambda_{\max}^{-1}(I + \bar{C}_{v,t} \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top) \mathrm{tr}\left(\bar{C}_{v,t} \bar{\Omega}_t^{-1} \Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top\right) \\ & \geq \lambda_{\max}^{-1}(I + \bar{C}_{v,t} \Sigma_{t|t}(\emptyset) \bar{C}_{v,t}^\top) \mathrm{tr}\left(\bar{C}_{v,t} \bar{\Omega}_t^{-1} \Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top\right) \\ & = \lambda_{\max}^{-1}(I + \bar{C}_{v,t} \Sigma_{t|t}(\emptyset) \bar{C}_{v,t}^\top) \mathrm{tr}\left(\Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top \bar{C}_{v,t} \bar{\Omega}_t^{-1}\right), \end{aligned} \tag{8.67}$$

where we used $\bar{\Omega}_t^{-1} \preceq \Sigma_{t|t}(\emptyset)$, which holds because of the following: the definition of $\bar{\Omega}_t$ implies $\bar{\Omega}_t \succeq \Sigma_{t|t-1}^{-1}(\mathcal{S} \cup \{v\})$, and as a result, from Lemma 10 we get $\bar{\Omega}_t^{-1} \preceq \Sigma_{t|t-1}(\mathcal{S} \cup \{v\})$. In addition, Corollary 9 and the fact that $\Sigma_{1|1}(\mathcal{S} \cup \{v\}) \preceq \Sigma_{1|1}(\emptyset)$, which holds due to Lemma 15, imply $\Sigma_{t|t-1}(\mathcal{S} \cup \{v\}) \preceq \Sigma_{t|t-1}(\emptyset)$. Finally, from eq. (14) in Lemma 14 it is $\Sigma_{t|t-1}(\emptyset) = \Sigma_{t|t}(\emptyset)$. Overall, the desired inequality $\bar{\Omega}_t^{-1} \preceq \Sigma_{t|t}(\emptyset)$ holds.

Consider a time $t' \in \{1, 2 \ldots, T\}$ such that for any time $t \in \{1, 2, \ldots, T\}$ it is $\bar{\Omega}_{t'}^{-1} \bar{C}_{v,t'}^\top \bar{C}_{v,t'} \bar{\Omega}_{t'}^{-1} \preceq \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top \bar{C}_{v,t} \bar{\Omega}_t^{-1}$, and let Φ be the matrix $\bar{\Omega}_{t'}^{-1} \bar{C}_{v,t'}^\top \bar{C}_{v,t'} \bar{\Omega}_{t'}^{-1}$; similarly, let l be the

$$\min_{t \in \{1,2 \ldots, T\}, v \in \mathcal{V}} \lambda_{\max}^{-1}(I + \bar{C}_{v,t} \Sigma_{t|t}(\emptyset) \bar{C}_{v,t}^\top).$$

Summing ineq. (8.67) across all times $t \in \{1, 2 \ldots, T\}$, and using Lemmata 13 and 11,

$$\begin{aligned}
g(\mathcal{S}) - g(\mathcal{S} \cup \{v\}) &\geq l \sum_{t=1}^{T} \operatorname{tr}\left(\Theta_t \bar{\Omega}_t^{-1} \bar{C}_{v,t}^\top \bar{C}_{v,t} \bar{\Omega}_t^{-1}\right) \\
&\geq l \sum_{t=1}^{T} \operatorname{tr}\left(\Theta_t \Phi\right) \\
&= l \operatorname{tr}\left(\Phi \sum_{t=1}^{T} \Theta_t\right) \\
&\geq l \lambda_{\min}\left(\sum_{t=1}^{T} \Theta_t\right) \operatorname{tr}\left(\Phi\right) \\
&> 0,
\end{aligned}$$

which is non-zero because $\sum_{t=1}^{T} \Theta_t \succ 0$ and Φ is a non-zero positive semi-definite matrix.

Finally, we lower bound $\operatorname{tr}(\Phi)$, using Lemma 11:

$$\begin{aligned}
\operatorname{tr}\left(\Phi\right) &= \operatorname{tr}\left(\bar{\Omega}_{t'}^{-1} \bar{C}_{v,t'}^\top \bar{C}_{v,t'} \bar{\Omega}_{t'}^{-1}\right) \\
&= \operatorname{tr}\left(\bar{\Omega}_{t'}^{-2} \bar{C}_{v,t'}^\top \bar{C}_{v,t'}\right) \\
&\geq \lambda_{\min}(\bar{\Omega}_{t'}^{-2}) \operatorname{tr}\left(\bar{C}_{v,t'}^\top \bar{C}_{v,t'}\right) \\
&= \lambda_{\min}^2(\bar{\Omega}_{t'}^{-1}) \operatorname{tr}\left(\bar{C}_{v,t'}^\top \bar{C}_{v,t'}\right) \\
&\geq \lambda_{\min}^2(\Sigma_{t'|t'}(\mathcal{V})) \operatorname{tr}\left(\bar{C}_{v,t'}^\top \bar{C}_{v,t'}\right), \qquad (8.68)
\end{aligned}$$

where ineq. (8.68) holds because $\bar{\Omega}_{t'}^{-1} \succeq \Sigma_{t'|t'}(\mathcal{V})$. In particular, the inequality $\bar{\Omega}_{t'}^{-1} \succeq \Sigma_{t'|t'}(\mathcal{S} \cup \{v\})$ is derived by applying Lemma 10 to the inequality $\bar{\Omega}_{t'} \preceq \bar{\Omega}_{t'} + \bar{C}_{v,t}^\top \bar{C}_{v,t}^\top = \Sigma_{t'|t'}^{-1}(\mathcal{S} \cup \{v\})$, where the equality holds by the definition of $\bar{\Omega}_{t'}$. In addition, due to Lemma 15 it is $\Sigma_{1|1}(\mathcal{S} \cup \{v\}) \succeq \Sigma_{1|1}(\mathcal{V})$, and as a result, from Corollary 8 it also is $\Sigma_{t'|t'}(\mathcal{S} \cup \{v\}) \succeq \Sigma_{t'|t'}(\mathcal{V})$. Overall, the desired inequality $\bar{\Omega}_{t'}^{-1} \succeq \Sigma_{t'|t'}(\mathcal{V})$ holds.

Upper bound for the denominator of the supermodularity ratio γ_g The denominator of the submodularity ratio γ_g is of the form

$$\sum_{t=1}^{T} [f_t(\mathcal{S}') - f_t(\mathcal{S}' \cup \{v\})],$$

for some sensor set $\mathcal{S}' \subseteq \mathcal{V}$, and sensor $v \in \mathcal{V}$; to upper bound it, from eq. (14) in Lemma 14 of Appendix A, observe

$$\Sigma_{t|t}(\mathcal{S}' \cup \{v\}) = [\Sigma_{t|t-1}^{-1}(\mathcal{S}' \cup \{v\}) + \sum_{i \in \mathcal{S}' \cup \{v\}} \bar{C}_{i,t}^\top \bar{C}_{i,t}]^{-1},$$

and let $H_t = \Sigma_{t|t-1}^{-1}(\mathcal{S}') + \sum_{i\in\mathcal{S}'} \bar{C}_{i,t}^\top \bar{C}_{i,t}$, and $\bar{H}_t = \Sigma_{t|t-1}^{-1}(\mathcal{S}' \cup \{v\}) + \sum_{i\in\mathcal{S}'} \bar{C}_{i,t}^\top \bar{C}_{i,t}$; using the Woodbury identity in Lemma 12,

$$f_t(\mathcal{S}' \cup \{v\}) = \mathrm{tr}\left(\Theta_t \bar{H}_t^{-1}\right) - \mathrm{tr}\left(\Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{H}_t^{-1}\right).$$

Therefore,

$$\sum_{t=1}^{T} [f_t(\mathcal{S}') - f_t(\mathcal{S}' \cup \{v\})] = \sum_{t=1}^{T} [\mathrm{tr}\left(\Theta_t H_t^{-1}\right) - \mathrm{tr}\left(\Theta_t \bar{H}_t^{-1}\right) + \mathrm{tr}\left(\Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{H}_t^{-1}\right)] \leq$$

$$\sum_{t=1}^{T} [\mathrm{tr}\left(\Theta_t H_t^{-1}\right) + \mathrm{tr}\left(\Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{H}_t^{-1}\right)], \tag{8.69}$$

where ineq. (8.69) holds since $\mathrm{tr}\left(\Theta_t \bar{H}_t^{-1}\right)$ is non-negative. In eq. (8.69), the second term in the sum is upper bounded as follows, using Lemma 11:

$$\begin{aligned}
&\mathrm{tr}\left(\Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top)^{-1} \bar{C}_{v,t} \bar{H}_t^{-1}\right) = \\
&\mathrm{tr}\left(\bar{C}_{v,t} \bar{H}_t^{-1} \Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top (I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top)^{-1}\right) \leq \\
&\mathrm{tr}\left(\bar{C}_{v,t} \bar{H}_t^{-1} \Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top\right) \lambda_{\max}[(I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top)^{-1}] = \\
&\mathrm{tr}\left(\bar{C}_{v,t} \bar{H}_t^{-1} \Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top\right) \lambda_{\min}^{-1}(I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top) \leq \\
&\mathrm{tr}\left(\bar{C}_{v,t} \bar{H}_t^{-1} \Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top\right) \lambda_{\min}^{-1}(I + \bar{C}_{v,t} \Sigma_{t|t}(\mathcal{V}) \bar{C}_{v,t}^\top),
\end{aligned} \tag{8.70}$$

since $\lambda_{\min}(I + \bar{C}_{v,t} \bar{H}_t^{-1} \bar{C}_{v,t}^\top) \geq \lambda_{\min}(I + \bar{C}_{v,t} \Sigma_{t|t}(\mathcal{V}) \bar{C}_{v,t}^\top)$, because $\bar{H}_t^{-1} \succeq \Sigma_{t|t}(\mathcal{V})$. In particular, the inequality $\bar{H}_t^{-1} \succeq \Sigma_{t|t}(\mathcal{V})$ is derived as follows: first, it is $\bar{H}_t \preceq \bar{H}_t + \bar{C}_{v,t}^\top \bar{C}_{v,t} = \Sigma_{t|t}(\mathcal{S}' \cup \{v\})^{-1}$, where the equality holds by the definition of $\bar{H}_t$, and now Lemma 10 implies $\bar{H}_t^{-1} \succeq \Sigma_{t|t}(\mathcal{S}' \cup \{v\})$. In addition, $\Sigma_{t|t}(\mathcal{S}' \cup \{v\}) \succeq \Sigma_{t|t}(\mathcal{V})$ is implied from Corollary 8, since Lemma 15 implies $\Sigma_{1|1}(\mathcal{S}' \cup \{v\}) \succeq \Sigma_{1|1}(\mathcal{V})$. Overall, the desired inequality $\bar{H}_t^{-1} \succeq \Sigma_{t|t}(\mathcal{V})$ holds.

Let $l' = \max_{t\in\{1,2\ldots,T\}, v\in\mathcal{V}} \lambda_{\min}^{-1}(I + \bar{C}_{v,t} \Sigma_{t|t}(\mathcal{V}) \bar{C}_{v,t}^\top)$. From ineqs. (8.69) and (8.70),

$$\sum_{t=1}^{T} [f_t(\mathcal{S}') - f_t(\mathcal{S}' \cup \{v\})] \leq \sum_{t=1}^{T} [\mathrm{tr}\left(\Theta_t H_t^{-1}\right) + l' \mathrm{tr}\left(\Theta_t \bar{H}_t^{-1} \bar{C}_{v,t}^\top \bar{C}_{v,t} \bar{H}_t^{-1}\right)]. \tag{8.71}$$

Consider times $t' \in \{1, 2\ldots, T\}$ and $t'' \in \{1, 2\ldots, T\}$ such that for any time $t \in \{1, 2, \ldots, T\}$, it is $H_{t'}^{-1} \succeq H_t^{-1}$ and $\bar{H}_{t''}^{-1} \bar{C}_{v,t''}^\top \bar{C}_{v,t''} \bar{H}_{t''}^{-1} \succeq \bar{H}_t^{-1} \bar{C}_{v,t}^\top \bar{C}_{v,t} \bar{H}_t^{-1}$, and let $\Xi = H_{t'}^{-1}$ and

$\Phi' = \bar{H}_{t'}^{-1}\bar{C}_{v,t'}^{\top}\bar{C}_{v,t'}\bar{H}_{t'}^{-1}$. From ineq. (8.71), and Lemma 13,

$$\sum_{t=1}^{T}[f_t(\mathcal{S}') - f_t(\mathcal{S}' \cup \{v\})] \leq$$
$$\sum_{t=1}^{T}[\mathrm{tr}\left(\Theta_t \Xi\right) + l'\mathrm{tr}\left(\Theta_t \Phi'\right)] \leq$$
$$\mathrm{tr}\left(\Xi \sum_{t=1}^{T}\Theta_t\right) + l'\mathrm{tr}\left(\Phi' \sum_{t=1}^{T}\Theta_t\right) \leq$$
$$(\mathrm{tr}\left(\Xi\right) + l'\mathrm{tr}\left(\Phi'\right))\lambda_{\max}(\sum_{t=1}^{T}\Theta_t). \tag{8.72}$$

Finally, we upper bound $\mathrm{tr}\left(\Xi\right) + l'\mathrm{tr}\left(\Phi'\right)$ in ineq. (8.72), using Lemma 11:

$$\mathrm{tr}\left(\Xi\right) + l'\mathrm{tr}\left(\Phi'\right) \leq$$
$$\mathrm{tr}\left(H_{t'}^{-1}\right) + \tag{8.73}$$
$$l'\lambda_{\max}^2(\bar{H}_{t''}^{-1})\mathrm{tr}\left(\bar{C}_{v,t''}^{\top}\bar{C}_{v,t''}\right) \leq$$
$$\mathrm{tr}\left(\Sigma_{t'|t'}(\emptyset)\right) + l'\lambda_{\max}^2(\Sigma_{t''|t''}(\emptyset))\mathrm{tr}\left(\bar{C}_{v,t''}^{\top}\bar{C}_{v,t''}\right), \tag{8.74}$$

where ineq. (8.74) holds because $H_{t'}^{-1} \preceq \Sigma_{t'|t'}(\emptyset)$, and similarly, $\bar{H}_{t''}^{-1} \preceq \Sigma_{t''|t''}(\emptyset)$. In particular, the inequality $H_{t'}^{-1} \preceq \Sigma_{t'|t'}(\emptyset)$ is implied as follows: first, by the definition of $H_{t'}$, it is $H_{t'}^{-1} = \Sigma_{t'|t'}(\mathcal{S}')$; and finally, Corollary 8 and the fact that $\Sigma_{1|1}(\mathcal{S}') \preceq \Sigma_{1|1}(\emptyset)$, which holds due to Lemma 15, imply $\Sigma_{t'|t'}(\mathcal{S}') \preceq \Sigma_{t'|t'}(\emptyset)$. In addition, the inequality $\bar{H}_{t''}^{-1} \preceq \Sigma_{t''|t''}(\emptyset)$ is implied as follows: first, by the definition of $\bar{H}_{t''}$, it is $\bar{H}_{t''} \succeq \Sigma_{t''|t''-1}^{-1}(\mathcal{S}' \cup \{v\})$, and as a result, Lemma 10 implies $\bar{H}_{t''}^{-1} \preceq \Sigma_{t''|t''-1}(\mathcal{S}' \cup \{v\})$. Moreover, Corollary 9 and the fact that $\Sigma_{1|1}(\mathcal{S} \cup \{v\}) \preceq \Sigma_{1|1}(\emptyset)$, which holds due to Lemma 15, imply $\Sigma_{t''|t''-1}(\mathcal{S}' \cup \{v\}) \preceq \Sigma_{t''|t''-1}(\emptyset)$. Finally, from eq. (14) in Lemma 14 it is $\Sigma_{t''|t''-1}(\emptyset) = \Sigma_{t''|t''}(\emptyset)$. Overall, the desired inequality $\bar{H}_{t''}^{-1} \preceq \Sigma_{t''|t''}(\emptyset)$ holds. ■

8.7.6. Proof of Theorem 21

For the proof of Theorem 21, we use Lemmata 25-28 below.

Lemma 25 (System-level condition for near-optimal co-design)**.** *Let N_1 be defined as in eq.* (8.11)*. The control policy $u_{1:T}^{\circ} \triangleq (0, 0, \ldots, 0)$ is suboptimal for the LQG problem in eq.* (8.24) *for all non-zero initial conditions x_1 if and only if*

$$\textstyle\sum_{t=1}^{T} A_1^{\top} \cdots A_t^{\top} Q_t A_t \cdots A_1 \succ N_1. \tag{8.75}$$

Proof of Lemma 25 For any initial condition x_1, eq. (8.29) in Lemma 19 implies for the noiseless perfect state information LQG problem in eq. (8.24):

$$\min_{u_{1:T}} \sum_{t=1}^{T} [\|x_{t+1}\|_{Q_t}^2 + \|u_t(x_t)\|_{R_t}^2]\big|_{\Sigma_{t|t}=W_t=0} = x_1^{\top} N_1 x_1, \tag{8.76}$$

since $\mathbb{E}(\|x_1\|_{N_1}^2) = x_1^{\top} N_1 x_1$, because x_1 is known ($\Sigma_{1|1} = 0$), and $\Sigma_{t|t}$ and W_t are zero.

In addition, for $u_{1:T} = (0, 0, \ldots, 0)$, the objective function in the noiseless perfect state information LQG problem in eq. (8.24) is

$$\begin{aligned}
&\sum_{t=1}^{T} [\|x_{t+1}\|_{Q_t}^2 + \|u_t(x_t)\|_{R_t}^2]\big|_{\Sigma_{t|t}=W_t=0} \\
&= \sum_{t=1}^{T} x_{t+1}^\top Q_t x_{t+1} \qquad (8.77) \\
&= x_1^\top \sum_{t=1}^{T} A_1^\top A_2^\top \cdots A_t^\top Q_t A_t A_{t-1} \cdots A_1 x_1,
\end{aligned}$$

since $x_{t+1} = A_t x_t = A_t A_{t-1} x_{t-1} = \ldots = A_t A_{t-1} \cdots A_1 x_1$ when all $u_1, u_2, \ldots, u_T$ are zero.

From eqs. (8.76) and (8.77), the inequality

$$x_1^\top N_1 x_1 < x_1^\top \sum_{t=1}^{T} A_1^\top A_2^\top \cdots A_t^\top Q_t A_t A_{t-1} \cdots A_1 x_1$$

holds for any non-zero x_1 if and only if

$$N_1 \prec \sum_{t=1}^{T} A_1^\top \cdots A_t^\top Q_t A_t A_{t-1} \cdots A_1.$$

■

Lemma 26. *For any $t = 1, 2, \ldots, T$,*

$$\Theta_t = A_t^\top S_t A_t + Q_{t-1} - S_{t-1}.$$

Proof of Lemma 26 Using the Woobury identity in Lemma 12, and the notation in eq. (8.11),

$$\begin{aligned}
N_t &= A_t^\top (S_t^{-1} + B_t R_t^{-1} B_t^\top)^{-1} A_t \\
&= A_t^\top (S_t - S_t B_t M_t^{-1} B_t^\top S_t) A_t \\
&= A_t^\top S_t A_t - \Theta_t.
\end{aligned}$$

The latter, gives $\Theta_t = A_t^\top S_t A_t - N_t$. In addition, from eq. (8.11), $-N_t = Q_{t-1} - S_{t-1}$, since $S_t = Q_t + N_{t+1}$. ■

Lemma 27. $\sum_{t=1}^{T} A_1^\top A_2^\top \cdots A_t^\top Q_t A_t A_{t-1} \cdots A_1 \succ N_1$ *if and only if*

$$\sum_{t=1}^{T} A_1^\top A_2^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} A_{t-2} \cdots A_1 \succ 0.$$

Proof of Lemma 27 For $i = t-1, t-2, \ldots, 1$, we pre- and post-multiply the identity in Lemma 26 with $A_i^\top$ and A_i, respectively:

$$\Theta_t = A_t^\top S_t A_t + Q_{t-1} - S_{t-1} \Rightarrow \tag{8.78}$$
$$A_{t-1}^\top \Theta_t A_{t-1} = A_{t-1}^\top A_t^\top S_t A_t A_{t-1} + A_{t-1}^\top Q_{t-1} A_{t-1} - \tag{8.79}$$
$$A_{t-1}^\top S_{t-1} A_{t-1} \Rightarrow \tag{8.80}$$
$$A_{t-1}^\top \Theta_t A_{t-1} = A_{t-1}^\top A_t^\top S_t A_t A_{t-1} + A_{t-1}^\top Q_{t-1} A_{t-1} - \tag{8.81}$$
$$\Theta_{t-1} + Q_{t-2} - S_{t-2} \Rightarrow \tag{8.82}$$
$$\Theta_{t-1} + A_{t-1}^\top \Theta_t A_{t-1} = A_{t-1}^\top A_t^\top S_t A_t A_{t-1} + \tag{8.83}$$
$$A_{t-1}^\top Q_{t-1} A_{t-1} + Q_{t-2} - S_{t-2} \Rightarrow \tag{8.84}$$
$$\ldots \Rightarrow \tag{8.85}$$
$$\Theta_2 + A_2^\top \Theta_3 A_2 + \ldots + A_2^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} \cdots A_2 = \tag{8.86}$$
$$A_2^\top \cdots A_t^\top S_t A_t \cdots A_2 + A_2^\top \cdots A_{t-1}^\top Q_{t-1} A_{t-1} \cdots A_2 + \tag{8.87}$$
$$\ldots + A_2^\top Q_2 A_2 + Q_1 - S_1 \Rightarrow \tag{8.88}$$
$$\Theta_1 + A_1^\top \Theta_2 A_1 + \ldots + A_1^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} \cdots A_1 = \tag{8.89}$$
$$A_1^\top \cdots A_t^\top S_t A_t \cdots A_1 + A_1^\top \cdots A_{t-1}^\top Q_{t-1} A_{t-1} \cdots A_1 + \tag{8.90}$$
$$\ldots + A_1^\top Q_1 A_1 - N_1 \Rightarrow \tag{8.91}$$
$$\sum_{t=1}^{T} A_1^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} \cdots A_1 = \tag{8.92}$$
$$\sum_{t=1}^{T} A_1^\top \cdots A_t^\top Q_t A_t \cdots A_1 - N_1. \tag{8.93}$$

The last equality in eq. (8.93) implies Lemma 27. ■

Lemma 28. *Consider for any $t = 1, 2, \ldots, T$ that A_t is invertible. It holds:*

$$\sum_{t=1}^{T} A_1^\top A_2^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} A_{t-2} \cdots A_1 \succ 0$$

if and only if

$$\sum_{t=1}^{T} \Theta_t \succ 0.$$

Proof of Lemma 28 Let $U_t = A_{t-1} A_{t-2} \cdots A_1$.

We first prove that for any non-zero vector z, if it is $\sum_{t=1}^{T} A_1^\top A_2^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} A_{t-2} \cdots A_1 \succ 0$, then $\sum_{t=1}^{T} z^\top \Theta_t z > 0$. In particular, since U_t is invertible, —because for any $t \in$

$\{1, 2, \ldots, T\}$, A_t is,—

$$\begin{aligned}\sum_{t=1}^{T} z^\top \Theta_t z &= \sum_{t=1}^{T} z^\top U_t^{-\top} U_t^\top \Theta_t U_t U_t^{-1} z \\ &= \sum_{t=1}^{T} \mathrm{tr}\left(\phi_t \phi_t^\top U_t^\top \Theta_t U_t\right),\end{aligned} \tag{8.94}$$

where we let $\phi_t = U_t^{-1} z$. Consider a time t' such that for any time $t \in \{1, 2 \ldots, T\}$, $\phi_{t'} \phi_{t'}^\top \preceq \phi_t \phi_t^\top$. From eq. (8.94), using Lemmata 13 and 11,

$$\begin{aligned}\sum_{t=1}^{T} z^\top \Theta_t z &\geq \sum_{t=1}^{T} \mathrm{tr}\left(\phi_{t'} \phi_{t'}^\top U_t^\top \Theta_t U_t\right) \\ &\geq \mathrm{tr}\left(\phi_{t'} \phi_{t'}^\top \sum_{t=1}^{T} U_t^\top \Theta_t U_t\right) \\ &\geq \mathrm{tr}\left(\phi_{t'} \phi_{t'}^\top\right) \lambda_{\min}(\sum_{t=1}^{T} U_t^\top \Theta_t U_t) \\ &= \|\phi_{t'}\|_2^2 \lambda_{\min}(\sum_{t=1}^{T} U_t^\top \Theta_t U_t) \\ &> 0.\end{aligned}$$

We finally prove that for any non-zero vector z, if $\sum_{t=1}^{T} \Theta_t \succ 0$, then"

$$\sum_{t=1}^{T} z A_1^\top \cdots A_{t-1}^\top \Theta_t A_{t-1} \cdots A_1 z \succ 0.$$

In particular,

$$\sum_{t=1}^{T} z^\top U_t^\top \Theta_t U_t z = \sum_{t=1}^{T} \mathrm{tr}\left(\xi_t^\top \Theta_t \xi_t\right), \tag{8.95}$$

where we let $\xi_t = U_t z$. Consider time t' such that for any time $t \in \{1, 2 \ldots, T\}$, $\xi_{t'} \xi_{t'}^\top \preceq \xi_t \xi_t^\top$. From eq. (8.94), using Lemmata 13 and 11,

$$\begin{aligned}\sum_{t=1}^{T} \mathrm{tr}\left(\xi_t^\top \Theta_t \xi_t\right) &\geq \mathrm{tr}\left(\xi_{t'} \xi_{t'}^\top \sum_{t=1}^{T} \Theta_t\right) \\ &\geq \mathrm{tr}\left(\xi_{t'} \xi_{t'}^\top\right) \lambda_{\min}(\sum_{t=1}^{T} \Theta_t) \\ &= \|\xi_{t'}\|_2^2 \lambda_{\min}(\sum_{t=1}^{T} \Theta_t) \\ &> 0.\end{aligned}$$

■

Proof of Theorem 21 Theorem 21 follows from the sequential application of Lemmata 25, 27, and 28. ■

Part III

RESILIENT SUBMODULAR MAXIMIZATION

CHAPTER 9 : Resilient Non-Submodular Maximization over Matroid Constraints

Applications in control, robotics, and optimization motivate the design of systems by selecting system elements, such as actuators, sensors, or data, subject to complex design constraints that require the system elements not only to be a few in number, but also, to satisfy *heterogeneity* or *global-interdependency* constraints; in particular, *matroid constraints*. However, in failure-prone and adversarial environments, sensors get attacked; actuators fail; data get deleted. Thence, traditional matroid-constrained design paradigms become insufficient and, in contrast, *resilient* matroid-constrained designs against attacks, failures, or deletions become important. In general, resilient matroid-constrained design problems are computationally hard. Also, even though they often involve objective functions that are monotone and (possibly) submodular, no scalable approximation algorithms are known for their solution. In this chapter, we provide the first algorithm, that achieves the following characteristics: *system-wide resiliency*, i.e., the algorithm is valid for any number of denial-of-service attacks, deletions, or failures; *minimal running time*, i.e., the algorithm terminates with the same running time as state-of-the-art algorithms for (non-resilient) matroid-constrained optimization; and *provable approximation performance*, i.e., the algorithm guarantees for monotone objective functions a solution close to the optimal. We quantify the algorithm's approximation performance using a notion of curvature for monotone (not necessarily submodular) set functions. Finally, we support our theoretical analyses with numerical experiments, by considering a control-aware sensor selection scenario, namely, *sensing-constrained robot navigation.*[1]

9.1. Introduction

Applications in control, robotics, and optimization require the design of systems in problems such as:

- (*Control*) *Leader selection*: In multi-robot systems, how should we choose a few leaders both to maximize the systems' capability to monitor phenomena despite communication noise, and to satisfy interdependency constraints where each robot must be controllable by the leaders? [218]

- (*Robotics*) *Target tracking*: At a team of flying robots, how should we select the robots' motions to maximize the team's capability for tracking adversarial targets in urban environments, subject to heterogeneity constraints where each robot has different motion capabilities? [6]

- (*Optimization*) *Data selection*: Given a flood of heterogeneous driving data, collected from the smart-phones of several types of drivers (e.g., truck or commercial vehicle drivers), which few data should we process from each driver-type to enable the prediction of car traffic? [219]

In particular, all the above applications motivate the design of systems by selecting system elements, such as actuators, sensors, or data, subject to complex design constraints that require the system elements not only to be a few in number, but also, to satisfy *heterogene-*

[1] This chapter is based on the paper by Tzoumas et al. [217].

ity or *global-interdependency* constraints. Additional applications in control, robotics, and optimization that involve such complex design constraints are:

- (*Control*) Sparse actuation and sensing design [4, 5, 10, 53, 58]; stabilization and voltage control in power grids [1, 25]; and synchronization in complex networks [9];
- (*Robotics*) Task allocation in collaborative multi-robot systems [11]; and agile autonomous robot navigation and sparse visual-cue selection [220];
- (*Optimization*) Sparse signal recovery and subset column selection [221, 222, 223]; and sparse approximation, dictionary and feature selection [224, 225, 226].

In more detail, all the aforementioned applications [1, 4, 5, 6, 9, 10, 11, 25, 53, 58, 218, 219, 220, 221, 222, 223, 224, 225, 226] require the solution to an optimization problem of the form:

$$\max_{\mathcal{A}\subseteq\mathcal{V},\, \mathcal{A}\in\mathcal{I}} \quad f(\mathcal{A}). \tag{9.1}$$

where the set $\mathcal{I}$ represents a collection of complex design constraints —called *matroids* [12]— that enforce *heterogeneity* or *global-interdependency* across the elements in $\mathcal{A}$; and the objective function f is monotone and (possibly) non-submodular; submodularity is a diminishing returns property. The problem in eq. (9.1) is combinatorial, and, specifically, it is NP-hard [13]; notwithstanding, approximation algorithms have been proposed for its solution, such as the greedy [12, 13, 31, 32, 33].

But in all the above critical applications, actuators can fail [23]; sensors can get cyber-attacked [24]; and data can get deleted [36]. Hence, in such failure-prone and adversarial scenarios, *resilient* matroid-constrained designs against denial-of-service attacks, deletions, or failures become important.

In this chapter, we formalize for the first time a problem of *resilient non-submodular maximization*, that goes beyond the traditional problem in eq. (9.1), and guards against attacks, failures, and deletions. In particular, we introduce the following resilient re-formulation of the problem in eq. (9.1):

$$\max_{\mathcal{A}\subseteq\mathcal{V},\, \mathcal{A}\in\mathcal{I}} \quad \min_{\mathcal{B}\subseteq\mathcal{A},\, \mathcal{B}\in\mathcal{I}'} \quad f(\mathcal{A}\setminus\mathcal{B}). \tag{9.2}$$

where the set $\mathcal{I}'$ represents the collection of possible set-removals $\mathcal{B}$ —attacks, failures, or deletions— from $\mathcal{A}$, each of some specified cardinality. Overall, the problem in eq. (9.2) maximizes f despite *worst-case* failures that compromise the maximization in eq. (9.1). Therefore, the problem formulation in eq. (9.2) is suitable in scenarios where there is no prior on the removal mechanism, as well as, in scenarios where protection against worst-case removals is essential, such as in expensive experiment designs, or missions of adversarial-target tracking.

Particularly, the optimization problem in eq. (9.2) may be interpreted as a 2-stage perfect information sequential game between two players [26, Chapter 4], namely, a "maximization" player (designer), and a "minimization" player (attacker), where the designer plays first, and selects $\mathcal{A}$ to maximize the objective function f, and, in contrast, the attacker plays second,

and selects $\mathcal{B}$ to minimize the objective function f. In particular, *the attacker first observes the designer's selection* $\mathcal{A}$, and then, selects $\mathcal{B}$ such that $\mathcal{B}$ is a worst-case set removal from $\mathcal{A}$.

In sum, the optimization problem in eq. (9.2) goes beyond traditional (non-resilient) optimization [12, 13, 31, 32, 33] by proposing *resilient* optimization; beyond merely cardinality-constrained resilient optimization [34, 35, 56] by proposing *matroid-constrained* resilient optimization; and beyond protection against non-adversarial set-removals [36, 37] by proposing protection against *worst-case* set-removals. Hence, the problem in eq. (9.2) aims to protect the complex design of systems, per *heterogeneity* or *global-interdependency* constraints, against attacks, failures, or deletion, which is a vital objective both for the safety of critical infrastructures, such as power grids [1, 25], and for the safety of critical missions, such as multi-target surveillance with teams of mobile robots [6].

Contributions. In this chapter, we make the contributions:

- (*Problem*) We formalize the problem of *resilient maximization over matroid constraints* against denial-of-service removals, per eq. (9.2). This is the first work to formalize, address, and motivate this problem.

- (*Solution*) We develop the first algorithm for the problem of resilient maximization over matroid constraints in eq. (9.2), and prove it enjoys the following properties:

  - *system-wide resiliency*: the algorithm is valid for any number of removals;

  - *minimal running time*: the algorithm terminates with the same running time as state-of-the-art algorithms for (non-resilient) matroid-constrained optimization;

  - *provable approximation performance*: the algorithm ensures for functions f that are monotone and (possibly) submodular —as it holds true in all above applications [1, 4, 5, 6, 9, 10, 11, 25, 53, 58, 218, 219, 220, 221, 222, 223, 224, 225, 226]— a solution close-to-optimal.

    To quantify the algorithm's approximation performance, we use a notion of curvature for monotone (not necessarily submodular) set functions.

- (*Simulations*) We demonstrate the necessity for the resilient re-formulation of the problem in eq. (9.1) by conducting numerical experiments in various scenarios of sensing-constrained autonomous robot navigation, varying the number of sensor failures. In addition, via the experiments we demonstrate the benefits of our approach.

Overall, the proposed algorithm herein enables the resilient re-formulation and solution of all aforementioned matroid-constrained applications [1, 4, 5, 6, 9, 10, 11, 25, 53, 58, 218, 219, 220, 221, 222, 223, 224, 225, 226]; we describe in detail the matroid-constraints involved in all aforementioned application in Section 9.2. Moreover, the proposed algorithm enjoys minimal running time, and provable approximation guarantees.

Organization of the rest of the chapter. Section 9.2 formulates the problem of resilient maximization over matroid constraints (Problem 4), and describes types of matroid

constraints in control, robotics, and optimization. Section 9.3 presents the first scalable, near-optimal algorithm for Problem 4. Section 9.4 presents the main result in this chapter, which characterizes the scalability and performance guarantees of the proposed algorithm. Section 9.5 presents numerical experiments over a control-aware sensor selection scenario. Section 9.6 concludes the chapter. All proofs are found in the chapter's Appendix.

Notation. Calligraphic fonts denote sets (e.g., $\mathcal{A}$). Given a set $\mathcal{A}$, then $2^{\mathcal{A}}$ denotes the power set of $\mathcal{A}$; $|\mathcal{A}|$ denotes $\mathcal{A}$'s cardinality; given also a set $\mathcal{B}$, then $\mathcal{A} \setminus \mathcal{B}$ denotes the set of elements in $\mathcal{A}$ that are not in $\mathcal{B}$; and the $(\mathcal{A}, \mathcal{B})$ is equivalent to $\mathcal{A} \cup \mathcal{B}$. Given a ground set $\mathcal{V}$, a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$, and an element $x \in \mathcal{V}$, the $f(x)$ denotes $f(\{x\})$.

9.2. Resilient Non-Submodular Maximization over Matroid Constraints

We formally define *resilient non-submodular maximization over matroid constraints.* We start with some basic definitions.

Definition 30 (Monotonicity)**.** *Consider a finite ground set $\mathcal{V}$. Then, a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* non-decreasing *if and only if for any sets $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{V}$, it holds $f(\mathcal{A}) \leq f(\mathcal{A}')$.*

Definition 31 (Matroid [30, Section 39.1])**.** *Consider a finite ground set $\mathcal{V}$, and a non-empty collection of subsets of $\mathcal{V}$, denoted by $\mathcal{I}$. Then, the pair $(\mathcal{V}, \mathcal{I})$ is called a* matroid *if and only if the following conditions hold:*

- *for any set $\mathcal{X} \subseteq \mathcal{V}$ such that $\mathcal{X} \in \mathcal{I}$, and for any set such that $\mathcal{Z} \subseteq \mathcal{X}$, it holds $\mathcal{Z} \in \mathcal{I}$;*
- *for any sets $\mathcal{X}, \mathcal{Z} \subseteq \mathcal{V}$ such that $\mathcal{X}, \mathcal{Z} \in \mathcal{I}$ and $|\mathcal{X}| < |\mathcal{Z}|$, it holds that there exists an element $z \in \mathcal{Z} \setminus \mathcal{X}$ such that $\mathcal{X} \cup \{z\} \in \mathcal{I}$.*

We next motivate Definition 31 by presenting three matroid examples —uniform, partition, and transversal matroid— that appear in applications in control, robotics, and optimization.

Uniform matroid, and applications. A matroid $(\mathcal{V}, \mathcal{I})$ is a *uniform matroid* if for a positive integer α it holds $\mathcal{I} \equiv \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, |\mathcal{A}| \leq \alpha\}$. Thus, the uniform matroid treats all elements in $\mathcal{V}$ *uniformly* (that is, as being the same), by only limiting their number in each set that is feasible in $\mathcal{I}$.

Applications of the uniform matroid in control, robotics, and optimization, arise when one cannot use an arbitrary number of system elements, e.g., actuators, sensors, or data, to achieve a desired system performance; for example, such sparse element-selection scenarios are necessitated in resource constrained environments of, e.g., limited battery, communication bandwidth, or data processing time [220]. In more detail, applications of sparse, uniform selection in control, robotics, and optimization include the following:

- (*Control*) Actuator and sensor placement, e.g., for system controllability with minimal control effort [4, 53], and for optimal smoothing or Kalman filtering [10, 58];
- (*Robotics*) Sparse visual-cue selection, e.g., for agile autonomous robot navigation [220];
- (*Optimization*) Sparse recovery and column subset selection, e.g., for experiment design [221, 222, 223].

Partition matroid, and applications. A matroid $(\mathcal{V}, \mathcal{I})$ is a *partition matroid* if for a positive integer n, disjoint sets $\mathcal{V}_1, \ldots, \mathcal{V}_n$, and positive integers $\alpha_1, \ldots, \alpha_n$, it holds $\mathcal{V} \equiv \mathcal{V}_1 \cup \cdots \cup \mathcal{V}_n$ and $\mathcal{I} \equiv \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, |\mathcal{A} \cap \mathcal{V}_i| \leq \alpha_i, \text{ for all } i = 1, \ldots, n\}$. Hence, the partition matroid goes beyond the uniform matroid by allowing for *heterogeneity* in the elements included in each set that is feasible in $\mathcal{I}$. We give two interpretations of the disjoint sets $\mathcal{V}_1, \ldots, \mathcal{V}_n$: the first interpretation considers that $\mathcal{V}_1, \ldots, \mathcal{V}_n$ correspond to the available elements across n different *types* (buckets) of elements, and correspondingly, the positive integers $\alpha_1, \ldots, \alpha_n$ constrain uniformly the number of elements one can use from each type $1, \ldots, n$ towards a system design goal; the second interpretation considers that $\mathcal{V}_1, \ldots, \mathcal{V}_n$ correspond to the available elements across n different *times*, and correspondingly, the positive integers $\alpha_1, \ldots, \alpha_n$ constrain uniformly the number of elements that one can use at each time $1, \ldots, n$.

Applications of the partition matroid in control, robotics, and optimization include all the aforementioned applications in scenarios where heterogeneity in the element-selection enhances the system performance; for example, to guarantee voltage control in power grids, one needs to (possibly) actuate different types of actuators [25], and to guarantee active target tracking, one needs to activate different sensors at each time step [5]. Additional applications of the partition matroid in control and robotics include the following:

- (*Control*) Synchronization in complex dynamical networks, e.g., for missions of motion coordination [9];
- (*Robotics*) Robot motion planning, e.g., for multi-target tracking with mobile robots [6];
- (*Optimization*) Sparse approximation and feature selection, e.g., for sparse dictionary selection [224, 225, 226].

Transversal matroid, and applications. A matroid $(\mathcal{V}, \mathcal{I})$ is a *transversal matroid* if for a positive integer n, and a collection of subsets $\mathcal{S}_1, \ldots, \mathcal{S}_n$ of $\mathcal{V}$, it holds I is the collection of all partial transversals of $(\mathcal{S}_1, \ldots, \mathcal{S}_n)$ —a partial transversal is defined as follows: for a finite set $\mathcal{V}$, a positive integer n, and a collection of subsets $\mathcal{S}_1, \ldots, \mathcal{S}_n$ of $\mathcal{V}$, a *partial transversal* of $(\mathcal{S}_1, \ldots, \mathcal{S}_n)$ is a subset $\mathcal{P}$ of $\mathcal{V}$ such that there exist a one-to-one map $\phi : \mathcal{P} \mapsto \{1, \ldots, n\}$ so that for all $p \in \mathcal{P}$ it holds $p \in \mathcal{S}_{\phi(p)}$; i.e., each element in $\mathcal{P}$ intersects with one —and only one— set among the sets $\mathcal{S}_1, \ldots, \mathcal{S}_n$.

An application of the transversal matroid in control is that of actuation selection for optimal control performance subject to structural controllability constraints [218].

Additional examples. Other matroid constraints in control, robotics, and optimization are found in the following papers:

- (*Control*) [1], for the stabilization of power grids;
- (*Robotics*) [11], for task allocation in multi-robot systems;
- (*Optimization*) [219], for general task assignments.

Given the aforementioned matroid-constrained application-examples, we now define the main problem in this chapter.

Problem 4. (Resilient Non-Submodular Maximization over Matroid Constraints) *Consider the problem parameters:*

- *a matroid* $(\mathcal{V},\mathcal{I})$*;*
- *an either uniform or partition matroid* $(\mathcal{V},\mathcal{I}')$*;*
- *a non-decreasing set function* $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ *such that (without loss of generality) it holds* $f(\emptyset) = 0$*, and for any set* $\mathcal{A} \subseteq \mathcal{V}$*, it also holds* $f(\mathcal{A}) \geq 0$*.*

The problem of resilient non-submodular maximization over matroid constraints *is to maximize the function* f *by selecting a set* $\mathcal{A} \subseteq \mathcal{V}$ *such that* $\mathcal{A} \in \mathcal{I}$*, and accounting for any worst-case set removal* $\mathcal{B} \subseteq \mathcal{A}$ *from* $\mathcal{A}$ *such that* $\mathcal{B} \in \mathcal{I}'$*. Formally:*[2]

$$\max_{\mathcal{A}\subseteq\mathcal{V},\, \mathcal{A}\in\mathcal{I}} \quad \min_{\mathcal{B}\subseteq\mathcal{A},\, \mathcal{B}\in\mathcal{I}'} \quad f(\mathcal{A}\setminus\mathcal{B}).$$

As we mentioned in this chapter's Introduction, Problem 4 may be interpreted as a 2-stage perfect information sequential game between two players [26, Chapter 4], namely, a "maximization" player, and a "minimization" player, where the "maximization" player plays first by selecting the set $\mathcal{A}$, and, then, *the "minimization" player observes* $\mathcal{A}$, and plays second by selecting a worst-case set removal $\mathcal{B}$ from $\mathcal{A}$.

In sum, Problem 4 aims to guard all the aforementioned applications [1, 4, 5, 6, 9, 10, 11, 25, 53, 58, 218, 219, 220, 221, 222, 223, 224, 225, 226] in control, robotics, and optimization against attacks, failures, or deletions, by proposing their resilient re-formulation, since all involve the maximization of non-decreasing functions subject to matroid constrains.

Lastly, we discuss the resilient re-formulation of two among the aforementioned applications [1, 4, 5, 6, 9, 10, 11, 25, 53, 58, 218, 219, 220, 221, 222, 223, 224, 225, 226]:

Actuator placement for minimal control effort [4, 53]: Given a dynamical system, the design objective is to select a few actuators to place in the system to achieve controllability with minimal control effort [53]. In particular, the actuator-selection framework is as follows: given a set $\mathcal{V}$ of available actuators to choose from, then, up to α actuators can be placed in the system. In more detail, the aforementioned actuator-selection problem can be captured by a uniform matroid $(\mathcal{V},\mathcal{I})$ where $\mathcal{I} \triangleq \{\mathcal{A} : \mathcal{A} \in \mathcal{V}, |\mathcal{A}| \leq \alpha\}$. However, in the case of a failure-prone environment where up to β actuators may fail, then a resilient re-formulation of the aforementioned problem formulation is necessary: Problem 4 suggests that such a resilient re-formulation can be achieved by modelling any set of β actuator-failures in $\mathcal{A}$ by a set $\mathcal{B}$ in the uniform matroid on $\mathcal{A}$ where $\mathcal{B} \subseteq \mathcal{A}$ and $|\mathcal{B}| \leq \beta$.

Multi-target coverage with mobile robots [6]: A number of adversarial targets are deployed in the environment, and a team of mobile robots $\mathcal{R}$ is tasked to cover them. To this end, at each

[2] Given a matroid $(\mathcal{V},\mathcal{I}')$, and any subset $\mathcal{A} \subseteq \mathcal{V}$, then, the $(\mathcal{A}, \{\mathcal{B} : \mathcal{B} \subseteq \mathcal{A}, \mathcal{B} \in \mathcal{I}'\})$ is also a matroid [30, Section 39.3].

time step the robots in $\mathcal{R}$ need to jointly choose their motion. In particular, the movement-selection framework is as follows: given a finite set of possible moves $\mathcal{M}_i$ for each robot $i \in \mathcal{R}$, then, at each time step each robot selects a move to make so that the team $\mathcal{R}$ covers collectively as many targets as possible. In more detail, since each robot in $\mathcal{R}$ can make only one move per time, if we denote by $\mathcal{A}$ the set of moves to be made by each robot in $\mathcal{R}$, then the aforementioned movement-selection problem can be captured by a partition matroid $(\mathcal{V}, \mathcal{I})$ such that $\mathcal{V} = \cup_{i \in \mathcal{R}} \mathcal{M}_i$ and $\mathcal{I} = \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, |\mathcal{M}_i \cap \mathcal{A}| \leq 1, \text{ for all } i \in \mathcal{R}\}$ [6]. However, in the case of an adversarial scenario where the targets can attack up to β robots, then a resilient re-formulation of the aforementioned problem formulation is necessary: Problem 4 suggests that such a resilient re-formulation can be achieved by modelling any set of β attacks to the robots in $\mathcal{R}$ by a set $\mathcal{B}$ in the uniform matroid on $\mathcal{S}$ where $\mathcal{B} \subseteq \mathcal{S}$ and $|\mathcal{B}| \leq \beta$.

9.3. Algorithm for Problem 4

We present the first scalable algorithm for Problem 4. The pseudo-code of the algorithm is described in Algorithm 19.

9.3.1. Intuition behind Algorithm 19

The goal of Problem 4 is to ensure a maximal value for an objective function f through a single maximization step, despite compromises to the solution of the maximization step. In particular, Problem 4 aims to select a set $\mathcal{A}$ towards a maximal value of f, despite that $\mathcal{A}$ is later compromised by a worst-case set removal $\mathcal{B}$, resulting to f being finally evaluated at the set $\mathcal{A} \setminus \mathcal{B}$ instead of the set $\mathcal{A}$. In this context, Algorithm 19 aims to fulfil the goal of Problem 4 by constructing the set $\mathcal{A}$ as the union of two sets, namely, the $\mathcal{A}_1$ and $\mathcal{A}_2$ (line 16 of Algorithm 19), whose role we describe in more detail below:

Set $\mathcal{A}_1$ approximates worst-case set removal from $\mathcal{A}$: Algorithm 19 aims with the set $\mathcal{A}_1$ to capture a worst-case set-removal of elements —per the matroid $(\mathcal{V}, \mathcal{I}')$— from the elements Algorithm 19 is going to select in the set $\mathcal{A}$; equivalently, the set $\mathcal{A}_1$ is aimed to act as a "bait" to an attacker that selects to remove the *best* set of elements from $\mathcal{A}$ per the matroid $(\mathcal{V}, \mathcal{I}')$ (*best* with respect to the elements' contribution towards the goal of Problem 4). However, the problem of selecting the *best* elements in $\mathcal{V}$ per a matroid constraint is a combinatorial and, in general, intractable problem [13]. For this reason, Algorithm 19 aims to *approximate* the best set of elements in $\mathcal{I}'$, by letting $\mathcal{A}_1$ be the set of elements with the largest marginal contributions to the value of the objective function f (lines 2-8 of Algorithm 19). In addition, since per Problem 4 the set $\mathcal{A}$ needs to be in the matroid $(\mathcal{V}, \mathcal{I})$, Algorithm 19 constructs $\mathcal{A}_1$ so that not only it is $\mathcal{A}_1 \in \mathcal{I}'$, as we described before, but so that it also is $\mathcal{A}_1 \in \mathcal{I}$ (lines 4-6 of Algorithm 19).

Set $\mathcal{A}_2$ is such that the set $\mathcal{A}_1 \cup \mathcal{A}_2$ approximates optimal solution to Problem 4: Assuming that $\mathcal{A}_1$ is the set that is going to be removed from Algorithm 19's set selection $\mathcal{A}$, Algorithm 19 needs to select a set of elements $\mathcal{A}_2$ to complete the construction of $\mathcal{A}$ so that $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ is in the matroid $(\mathcal{V}, \mathcal{I})$, per Problem 4. In particular, for $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ to be an optimal solution to Problem 4 (assuming the removal of $\mathcal{A}_1$ from $\mathcal{A}$), Algorithm 19 needs to select $\mathcal{A}_2$ as a *best* set of elements from $\mathcal{V} \setminus \mathcal{A}_1$ subject to the constraint that $\mathcal{A}_1 \cup \mathcal{A}_2$ is in $(\mathcal{V}, \mathcal{I})$ (lines 11-13 of Algorithm 19). Nevertheless, the problem of selecting

Algorithm 19 Scalable algorithm for Problem 4.

Input: Per Problem 4, Algorithm 19 receives the inputs:

- a matroid $(\mathcal{V}, \mathcal{I})$;
- an either uniform or partition matroid $(\mathcal{V}, \mathcal{I}')$;
- a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that it is $f(\emptyset) = 0$, and for any set $\mathcal{A} \subseteq \mathcal{V}$, it also is $f(\mathcal{A}) \geq 0$.

Output: Set $\mathcal{A}$.

1: $\mathcal{A}_1 \leftarrow \emptyset$; $\mathcal{R}_1 \leftarrow \emptyset$; $\mathcal{A}_2 \leftarrow \emptyset$; $\mathcal{R}_2 \leftarrow \emptyset$;
2: **while** $\mathcal{R}_1 \neq \mathcal{V}$ **do**
3: $x \in \arg\max_{y \in \mathcal{V} \setminus \mathcal{R}_1} f(y)$;
4: **if** $\mathcal{A}_1 \cup \{x\} \in \mathcal{I}$ and $\mathcal{A}_1 \cup \{x\} \in \mathcal{I}'$ **then**
5: $\mathcal{A}_1 \leftarrow \mathcal{A}_1 \cup \{x\}$;
6: **end if**
7: $\mathcal{R}_1 \leftarrow \mathcal{R}_1 \cup \{x\}$;
8: **end while**
9: **while** $\mathcal{R}_2 \neq \mathcal{V} \setminus \mathcal{A}_1$ **do**
10: $x \in \arg\max_{y \in \mathcal{V} \setminus (\mathcal{A}_1 \cup \mathcal{R}_2)} f(\mathcal{A}_2 \cup \{y\})$;
11: **if** $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \{x\} \in \mathcal{I}$ **then**
12: $\mathcal{A}_2 \leftarrow \mathcal{A}_2 \cup \{x\}$;
13: **end if**
14: $\mathcal{R}_2 \leftarrow \mathcal{R}_2 \cup \{x\}$;
15: **end while**
16: $\mathcal{A} \leftarrow \mathcal{A}_1 \cup \mathcal{A}_2$;

a *best* set of elements subject to such a constraint is a combinatorial and, in general, intractable problem [13]. Hence, Algorithm 19 aims to *approximate* such a best set, using the greedy procedure in the lines 9-15 of Algorithm 19.

Overall, Algorithm 19 constructs the sets $\mathcal{A}_1$ and $\mathcal{A}_2$ to approximate with their union $\mathcal{A}$ an optimal solution to Problem 4.

We next describe the steps in Algorithm 19 in more detail.

9.3.2. Description of steps in Algorithm 19

Algorithm 19 executes four steps:

Initialization (line 1 of Algorithm 19): Algorithm 19 defines four auxiliary sets, namely, the $\mathcal{A}_1$, $\mathcal{R}_1$, $\mathcal{A}_2$, and $\mathcal{R}_2$, and initializes each of them with the empty set (line 1 of Algorithm 19). *The purpose of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ is to construct the set $\mathcal{A}$, which is the set Algorithm 19 selects as a solution to Problem 4; in particular, the union of $\mathcal{A}_1$ and of $\mathcal{A}_2$ constructs $\mathcal{A}$ by the end of Algorithm 19 (line 16 of Algorithm 19). *The purpose of* $\mathcal{R}_1$ *and of* $\mathcal{R}_2$ is to support the construction of $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively; in particular, during the construction of $\mathcal{A}_1$, Algorithm 19 stores in $\mathcal{R}_1$ the elements of $\mathcal{V}$ that have either been included already or cannot be included in $\mathcal{A}_1$ (line 7 of Algorithm 19), and that way, Algorithm 19 keeps track of which

elements remain to be checked whether they could be added in $\mathcal{A}_1$ (line 5 of Algorithm 19). Similarly, during the construction of $\mathcal{A}_2$, Algorithm 19 stores in $\mathcal{R}_2$ the elements of $\mathcal{V} \setminus \mathcal{A}_1$ that have either been included already or cannot be included in $\mathcal{A}_2$ (line 14 of Algorithm 19), and that way, Algorithm 19 keeps track of which elements remain to be checked whether they could be added in $\mathcal{A}_2$ (line 12 of Algorithm 19).

Construction of set $\mathcal{A}_1$ (lines 2-8 of Algorithm 19): Algorithm 19 constructs the set $\mathcal{A}_1$ sequentially —by adding one element at a time from $\mathcal{V}$ to $\mathcal{A}_1$, over a sequence of multiple time-steps— such that $\mathcal{A}_1$ is contained in both the matroid $(\mathcal{V}, \mathcal{I})$ and the matroid $(\mathcal{V}, \mathcal{I}')$ (line 4 of Algorithm 19), and such that each element $v \in \mathcal{V}$ that is chosen to be added in $\mathcal{A}_1$ achieves the highest marginal value of $f(v)$ among all the elements in $\mathcal{V}$ that have not been yet added in $\mathcal{A}_1$ and can be added in $\mathcal{A}_1$ (line 5 of Algorithm 19).

Construction of set $\mathcal{A}_2$ (lines 9-15 of Algorithm 19): Algorithm 19 constructs the set $\mathcal{A}_2$ sequentially, by picking greedily elements from the set $\mathcal{V}_t \setminus \mathcal{A}_1$ such that $\mathcal{A}_1 \cup \mathcal{A}_2$ is contained in the matroid $(\mathcal{V}, \mathcal{I})$. Specifically, the greedy procedure in Algorithm 19's "while loop" (lines 9-15 of Algorithm 19) selects an element $y \in \mathcal{V} \setminus (\mathcal{A}_1 \cup \mathcal{R}_2)$ to add in $\mathcal{A}_2$ only if y maximizes the value of $f(\mathcal{A}_2 \cup \{y\})$, where the set $\mathcal{R}_2$ stores the elements that either have already been added to $\mathcal{A}_2$ or have been considered to be added to $\mathcal{A}_2$ but they were not because the resultant set $\mathcal{A}_1 \cup \mathcal{A}_2$ would not be in the matroid $(\mathcal{V}, \mathcal{I})$.

Construction of set $\mathcal{A}$ (line 16 of Algorithm 19): Algorithm 19 constructs the set $\mathcal{A}$ as the union of the previously constructed sets $\mathcal{A}_1$ and $\mathcal{A}_2$ (lines 16 of Algorithm 19).

In sum, Algorithm 19 proposes a set $\mathcal{A}$ as solution to Problem 4, and in particular, Algorithm 19 constructs the set $\mathcal{A}$ so it can withstand any compromising set removal from it.

9.4. Performance Guarantees for Algorithm 19

We quantify Algorithm 19's performance, by bounding its running time, and its approximation performance. To this end, we use the following two notions of curvature for set functions, as well as, a notion of rank for a matroid.

9.4.1. Curvature and total curvature of non-decreasing functions

We present the notions of *curvature* and of *total curvature* for non-decreasing set functions. We start by describing the notions of *modularity* and *submodularity* for set functions.

Definition 32 (Modularity)**.** *Consider any finite set $\mathcal{V}$. The set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is modular if and only if for any set $\mathcal{A} \subseteq \mathcal{V}$, it holds $g(\mathcal{A}) = \sum_{v \in \mathcal{A}} g(v)$.*

In words, a set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is modular if through g all elements in $\mathcal{V}$ cannot substitute each other; in particular, Definition 32 of modularity implies that for any set $\mathcal{A} \subseteq \mathcal{V}$, and for any element $v \in \mathcal{V} \setminus \mathcal{A}$, it holds $g(\{v\} \cup \mathcal{A}) - g(\mathcal{A}) = g(v)$.

Definition 33 (Submodularity [70, Proposition 2.1])**.** *Consider any finite set $\mathcal{V}$. Then, the set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* submodular *if and only if for any sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, and any element $v \in \mathcal{V}$, it holds $g(\mathcal{A} \cup \{v\}) - g(\mathcal{A}) \geq g(\mathcal{B} \cup \{v\}) - g(\mathcal{B})$.*

Definition 33 implies that a set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is submodular if and only if it satisfies

a diminishing returns property where for any set $\mathcal{A} \subseteq \mathcal{V}$, and for any element $v \in \mathcal{V}$, the marginal gain $g(\mathcal{A} \cup \{v\}) - g(\mathcal{A})$ is non-increasing. In contrast to modularity, submodularity implies that the elements in $\mathcal{V}$ *can* substitute each other, since Definition 33 of submodularity implies the inequality $g(\{v\} \cup \mathcal{A}) - g(\mathcal{A}) \leq g(v)$; that is, in the presence of the set $\mathcal{A}$, the element v may lose part of its contribution to the value of $g(\{x\} \cup \mathcal{A})$.

Definition 34. (Curvature of monotone submodular functions [33]) *Consider a finite set $\mathcal{V}$, and a non-decreasing submodular set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that (without loss of generality) for any element $v \in \mathcal{V}$, it is $g(v) \neq 0$. Then, the curvature of g is defined as follows:*

$$\kappa_g \triangleq 1 - \min_{v \in \mathcal{V}} \frac{g(\mathcal{V}) - g(\mathcal{V} \setminus \{v\})}{g(v)}. \tag{9.3}$$

Definition 34 of curvature implies that for any non-decreasing submodular set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$, it holds $0 \leq \kappa_g \leq 1$. In particular, the value of κ_g measures how far g is from modularity, as we explain next: if $\kappa_g = 0$, then for all elements $v \in \mathcal{V}$, it holds $g(\mathcal{V}) - g(\mathcal{V} \setminus \{v\}) = g(v)$, that is, g is modular. In contrast, if $\kappa_g = 1$, then there exist an element $v \in \mathcal{V}$ such that $g(\mathcal{V}) = g(\mathcal{V} \setminus \{v\})$, that is, in the presence of $\mathcal{V} \setminus \{v\}$, v loses all its contribution to the value of $g(\mathcal{V})$.

Definition 35. (Total curvature of non-decreasing functions [15, Section 8]) *Consider a finite set $\mathcal{V}$, and a monotone set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$. Them, the total curvature of g is defined as follows:*

$$c_g \triangleq 1 - \min_{v \in \mathcal{V}} \min_{\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}} \frac{g(\{v\} \cup \mathcal{A}) - g(\mathcal{A})}{g(\{v\} \cup \mathcal{B}) - g(\mathcal{B})}. \tag{9.4}$$

Definition 35 of total curvature implies that for any non-decreasing set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$, it holds $0 \leq c_g \leq 1$. To connect the notion of total curvature with that of curvature, we note that when the function g is non-decreasing and submodular, then the two notions coincide, i.e., it holds $c_g = \kappa_g$; the reason is that if g is non-decreasing and submodular, then the inner minimum in eq. (9.4) is attained for $\mathcal{A} = \mathcal{B} \setminus \{v\}$ and $\mathcal{B} = \emptyset$. In addition, to connect the above notion of total curvature with the notion of modularity, we note that if $c_g = 0$, then g is modular, since eq. (9.4) implies that for any elements $v \in \mathcal{V}$, and for any sets $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}$, it holds:

$$(1 - c_g)\,[g(\{v\} \cup \mathcal{B}) - g(\mathcal{B})] \leq g(\{v\} \cup \mathcal{A}) - g(\mathcal{A}), \tag{9.5}$$

which for $c_g = 0$ implies the modularity of g. Finally, to connect the above notion of total curvature with the notion of monotonicity, we mention that if $c_g = 1$, then eq. (9.5) implies that g is merely non-decreasing (as it is already assumed by the Definition 35 of total curvature).

9.4.2. Rank of a matroid

We present a notion of rank for a matroid.

Definition 36 (Rank of a matroid [30, Section 39.1])**.** *Consider a matroid $(\mathcal{V}, \mathcal{I})$. Then, the rank of $(\mathcal{V}, \mathcal{I})$ is the number equal to the cardinality of the set $\mathcal{X} \in \mathcal{I}$ with the maximum cardinality among all sets in $\mathcal{I}$.*

For example, per the discussions in Section 9.2, for a uniform matroid $(\mathcal{V},\mathcal{I})$ of the form $\mathcal{I} \equiv \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, |\mathcal{A}| \leq \alpha\}$, the rank is equal to α; and for a partition matroid $(\mathcal{V},\mathcal{I})$ of the form $\mathcal{V} \equiv \mathcal{V}_1 \cup \cdots \cup \mathcal{V}_n$ and $\mathcal{I} \equiv \{\mathcal{A} : \mathcal{A} \subseteq \mathcal{V}, |\mathcal{A} \cap \mathcal{V}_i| \leq \alpha_i, \text{ for all } i = 1, \ldots, n\}$, the rank is equal to $\alpha_1 + \ldots + \alpha_n$.

9.4.3. Performance analysis for Algorithm 19

We quantify Algorithm 19's approximation performance, as well as, its running time per maximization step in Problem 4.

Theorem 22 (Performance of Algorithm 19)**.** *Consider an instance of Problem 4, the notation therein, the notation in Algorithm 19, and the definitions:*

- *let the number $f^\star$ be the (optimal) value to Problem 4;*
- *given a set $\mathcal{A}$ as solution to Problem 4, let $\mathcal{B}^\star(\mathcal{A})$ be an optimal (worst-case) set removal from $\mathcal{A}$, per Problem 4, that is: $\mathcal{B}^\star(\mathcal{A}) \in \arg \min_{\mathcal{B} \subseteq \mathcal{A}, \mathcal{B} \in \mathcal{I}'(\mathcal{A})} f(\mathcal{A} \setminus \mathcal{B})$;*
- *let the numbers α and β be such that α is the rank of the matroid $(\mathcal{V},\mathcal{I})$; and β is the rank of the matroid $(\mathcal{V},\mathcal{I}')$;*
- *define $h(\alpha,\beta) \triangleq \max[1/(1+\alpha), 1/(\alpha-\beta)]$.*[3]

The performance of Algorithm 19 is bounded as follows:

*leftmirgin=** (Approximation performance) *Algorithm 19 returns a set $\mathcal{A}$ such that $\mathcal{A} \subseteq \mathcal{V}$, $\mathcal{A} \in \mathcal{I}$, and:*

- *if f is* non-decreasing *and* submodular*, and:*
  - *if $(\mathcal{V},\mathcal{I})$ is a* uniform *matroid, then:*

$$\frac{f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A}))}{f^\star} \geq \frac{\max\left[1-\kappa_f, h(\alpha,\beta)\right]}{\kappa_f}(1-e^{-\kappa_f}); \tag{9.6}$$

  - *if $(\mathcal{V},\mathcal{I})$ is* any *matroid, then:*

$$\frac{f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A}))}{f^\star} \geq \frac{\max\left[1-\kappa_f, h(\alpha,\beta)\right]}{1+\kappa_f}; \tag{9.7}$$

- *if f is* non-decreasing*, then:*

$$\frac{f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A}))}{f^\star} \geq (1-c_f)^3. \tag{9.8}$$

*leftmiirgiin=** (Running time) *Algorithm 19 constructs the set $\mathcal{A}$ as a solutions to Problem 4 with $O(|\mathcal{V}|^2)$ evaluations of f.*

Provable approximation performance. Theorem 22 implies on the approximation per-

[3] A plot of $h(\alpha,\beta)$ is found in Fig. 10.

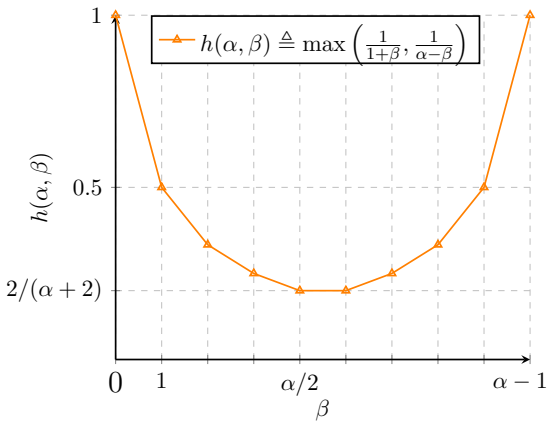


Figure 10: Given a natural number α, plot of $h(\alpha, \beta)$ versus β. Given a finite α, then $h(\alpha, \beta)$ is always non-zero, with minimum value $2/(\alpha + 2)$, and maximum value 1.

formance of Algorithm 19:

Near-optimality: Both for any monotone submodular objective functions f, and for any merely monotone objective functions f with total curvature $c_f < 1$, Algorithm 19 guarantees a value for Problem 4 finitely close to the optimal. In particular, per ineq. (9.6) and ineq. (9.7) (*case of submodular functions*), the approximation factor of Algorithm 19 is bounded by $\frac{h_f(\alpha,\beta)}{\kappa_f}(1 - e^{-\kappa_f})$ and $\frac{h_f(\alpha,\beta)}{1+\kappa_f}$, respectively, which for any finite number α are both non-zero (see also Fig. 10); in addition, per ineq. (9.6) and ineq. (9.7), the approximation factor of Algorithm 19 is also bounded by $\frac{1-\kappa_f}{\kappa_f}(1 - e^{-\kappa_f})$ and $\frac{1-\kappa_f}{1+\kappa_f}$, respectively, which are also non-zero for any monotone submodular function f with $\kappa_f < 1$ (see also Fig. 11). Similarly, per ineq. (9.8) (*case of monotone functions*), the approximation factor of Algorithm 19 is bounded by $(1 - c_f)^3$, which is non-zero for any monotone function f with $c_f < 1$ —notably, although it is known for the problem of (non-resilient) set function maximization that the approximation bound $(1-c_f)$ is tight [15, Theorem 8.6], the tightness of the bound $(1 - c_f)^3$ in ineq. (9.8) for Problem 4 is an open problem.

We discuss classes of functions f with curvatures $\kappa_f < 1$ or $c_f < 1$, along with relevant applications, in the remark below.

Remark 16. (Classes of functions f with $\kappa_f < 1$ or $c_f < 1$, and applications) Classes of functions f with $\kappa_f < 1$ *are the concave over modular functions [31, Section 2.1], and the* $\log\det$ *of positive-definite matrices [227, 228].* Classes of functions f with $c_f < 1$ *are support selection functions [223], and estimation error metrics such as the average minimum square error of the Kalman filter [193, Theorem 4].*

The aforementioned classes of functions f with $\kappa_f < 1$ or $c_f < 1$ appear in applications of control, robotics, and optimization, such as actuator and sensor placement [4, 10, 53, 58], sparse approximation and feature selection [225, 226], and sparse recovery and column subset selection [221, 222]; as a result, Problem 4 enables critical applications such as resilient actuator placement for minimal control effort, resilient multi-robot navigation with minimal sensing and communication, and resilient experiment design; see, for example, [229].

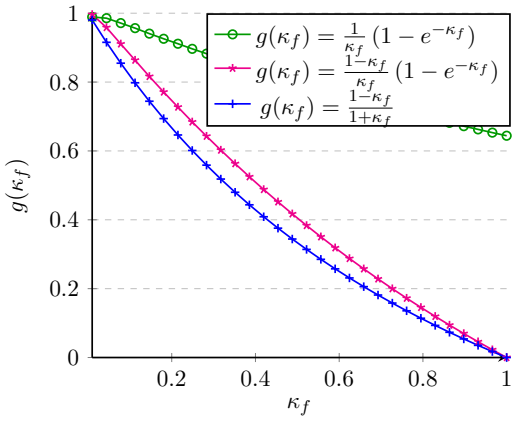


Figure 11: Plot of $g(\kappa_f)$ versus curvature κ_f of a monotone submodular function f. By definition, the curvature κ_f of a monotone submodular function f takes values between 0 and 1. $g(\kappa_f)$ increases from 0 to 1 as κ_f decreases from 1 to 0.

Approximation performance for low curvature: For both monotone submodular and merely monotone functions f, when the curvature κ_f and the total curvature c_f, respectively, tend to zero, Algorithm 19 becomes exact, since for $\kappa_f \to 0$ and $c_f \to 0$ the terms $\frac{1-\kappa_f}{\kappa_f}(1 - e^{-\kappa_f})$, $\frac{1-\kappa_f}{1+\kappa_f}$, and $(1 - c_f)^3$ in ineqs. (9.6)-(9.8) respectively, tend to 1. Overall, Algorithm 19's curvature-dependent approximation bounds make a first step towards separating the classes of monotone submodular and merely monotone functions into functions for which Problem 4 can be approximated well (low curvature functions), and functions for which it cannot (high curvature functions).

A machine learning problem where Algorithm 19 guarantees an approximation performance close to 100% the optimal is that of Gaussian process regression for processes with RBF kernels [114, 230]; this problem emerges in applications of sensor deployment and scheduling for temperature monitoring. The reason that in this class of regression problems Algorithm 19 performs almost optimally is that the involved objective function is the entropy of the selected sensor measurements, which for Gaussian processes with RBF kernels has curvature value close to zero [228, Theorem 5].

Approximation performance for no failures, deletions, or attacks: Both for monotone submodular functions f, and for merely monotone functions f, when the number of set removals is zero, —i.e., when $\mathcal{I}' = \emptyset$ in Problem 4, which implies $\beta = 0$ in Theorem 22,— Algorithm 19's approximation performance is the same as that of the state-of-the-art algorithms for (non-resilient) set function maximization. In particular, *for monotone submodular functions*, scalable algorithms for (non-resilient) set function maximization have approximation performance at least $\frac{1}{\kappa_f}(1 - e^{-\kappa_f})$ the optimal for any uniform matroid constraint [33, Theorem 5.4], and $\frac{1}{1+\kappa_f}$ the optimal for any matroid constraint [33, Theorem 2.3]; at the same time, per Theorem 22, when $\beta = 0$, then Algorithm 19 also has approximation performance at least $\frac{1}{\kappa_f}(1 - e^{-\kappa_f})$ the optimal for any uniform matroid constraint, and $\frac{1}{1+\kappa_f}$ the optimal for any matroid constraint, since for $\beta = 0$ it is $h(\alpha, \beta) = 1$ in ineq. (9.6) and ineq. (9.7).

Finally, *for monotone functions f*, and for $\mathcal{I}' = \emptyset$, Algorithm 19 is the same as the algorithm proposed in [12, Section 2] for (non-resilient) set function maximization, whose performance is optimal [15, Theorem 8.6].

Minimal running time. Theorem 22 implies that Algorithm 19, even though it goes beyond the objective of (non-resilient) set function optimization, by accounting for attacks, deletions, and failures, it has the same order of running time as state-of-the-art algorithms for (non-resilient) set function optimization. In particular, such algorithms for (non-resilient) set function optimization [12, 15, 70] terminate with $O(|\mathcal{V}|^2)$ evaluations of the function f, and Algorithm 19 also terminates with $O(|\mathcal{V}|^2)$ evaluations of the function f.

Summary of theoretical results. In sum, Algorithm 19 is the first algorithm for the problem of resilient maximization over matroid constraints (Problem 4), and it enjoys:

- *system-wide resiliency*: Algorithm 19 is valid for any number of denial-of-service attacks, deletions, and failures;
- *minimal running time*: Algorithm 19 terminates with the same running time as state-of-the-art algorithms for (non-resilient) matroid-constrained optimization;
- *provable approximation performance*: Algorithm 19 ensures for all monotone objective functions f that are either submodular, or merely non-decreasing with total curvature $c_f < 1$, a solution finitely close to the optimal.

Overall, Algorithm 19 makes the first step to ensure the success of critical applications in control, robotics, and optimization [1, 4, 5, 6, 9, 10, 11, 25, 53, 58, 218, 219, 220, 221, 222, 223, 224, 225, 226], despite compromising worst-case attacks, failures, or deletions, and with minimal running time.

9.5. Numerical Experiments on Control-Aware Sensor Selection

In this section, we demonstrate a near-optimal performance of Algorithm 19 in numerical experiments. In particular, we consider a control-aware sensor selection scenario, namely, *sensing-constrained robot navigation*, where the robot's localization for navigation is supported by both sensors on-board to the robot, and sensors deployed in the environment.[4] Specifically, we consider an unmanned aerial vehicle (UAV) which has the objective to land but it has limited battery and measurement-processing power to utilize to this end; as a result, it needs to activate only a subset of the available sensors to localize itself and to enable that way the generation of a control input for landing; specifically, we consider that the UAV generates its control input via an LQG controller, given the measurements from the activated sensor set [123].

In more detail, herein we present a Monte Carlo analysis of the above sensing-constrained robot navigation scenario for instances where sensor failures are present, and observe that Algorithm 19 results to a near-optimal sensor selection; that is, the resulting navigation

[4] The scenario of sensing-constrained robot navigation with on-board sensors is introduced and motivated in [193, Section V]; see also [128] for the case of autonomous robot navigation with deployed sensors in the environment.

performance of the UAV matches the optimal in all tested instances where the optimal sensor selection could be computed via a brute-force algorithm.

Simulation setup. We consider an UAV that moves in a 3D space, starting from a randomly selected initial location. The objective of the UAV is to land at position [0, 0, 0] with zero velocity. The UAV is modelled as a double-integrator with state $x_t = [p_t \; v_t]^\top \in \mathbb{R}^6$ at each time $t = 1, 2, \ldots$ (p_t is the 3D position of the UAV, and v_t is its velocity), and can control its own acceleration $u_t \in \mathbb{R}3$; the process noise is chosen as $W_t = \mathbf{I}_6$. The UAV may support its localization by utilizing 2 on-board sensors and 12 deployed sensors on the ground. The on-board sensors are one GPS receiver, measuring the UAV position p_t with a covariance $2 \cdot \mathbf{I}_3$, and one altimeter, measuring only the last component of p_t (altitude) with standard deviation 0.5m. The ground sensors vary with each Monte Carlo run, and are generated randomly; we consider them to provide linear measurements of the UAV's state. Among the aforementioned 14 available sensors to the UAV, we assume that the UAV can use only α of them.

In particular, the UAV chooses the α sensors to activate so to minimize an LQG cost of the form:

$$\sum_{t=1}^{T} [x_t^\top Q x_t + u_t^\top R u_t], \tag{9.9}$$

per the problem formulation in [193, Section II], where the cost matrix Q penalizes the deviation of the state vector from the zero state (since the UAV's objective is to land at position [0, 0, 0] with zero velocity), and the cost matrix R penalizes the control input vector; specifically, in the simulation setup herein we consider $Q = \text{diag}\left([1e^{-3}, \; 1e^{-3}, \; 10, \; 1e^{-3}, \; 1e^{-3}, \; 10]\right)$ and $R = \mathbf{I}_3$. Note that the structure of Q reflects the fact that during landing we are particularly interested in controlling the vertical direction and the vertical velocity (entries with larger weight in Q), while we are less interested in controlling accurately the horizontal position and velocity (assuming a sufficiently large landing site). In [193, Section III] it is proven that the UAV selects an optimal sensor set $\mathcal{S}$, and enables the generation of an optimal LQG control input with cost matrices Q and R, if it selects $\mathcal{S}$ by minimizing an objective function of the form:

$$\sum_{t=1}^{T} \text{trace}[M_t \Sigma_{t|t}(\mathcal{S})], \tag{9.10}$$

where M_t is a positive semi-definite matrix that depends on the LQG cost matrices Q and R, as well as, on the UAV's system dynamics; and $\Sigma_{t|t}(\mathcal{S})$ is the error covariance of the Kalman filter given the sensor set selection $\mathcal{S}$.

Compared algorithms. We compare four algorithms; all algorithms only differ in how they select the sensors used. The first algorithm is the optimal sensor selection algorithm, denoted as `optimal`, which attains the minimum of the cost function in eq. (9.10); this brute-force approach is viable since the number of available sensors is small. The second approach is a random sensor selection, denoted as `random`*. The third approach, denoted as `logdet`, selects sensors to greedily minimize the cost function in eq. (9.10), *ignoring the possibility of sensor failures*, per the problem formulation in eq. (9.1). The fourth approach

uses Algorithm 19 to solve the resilient re-formulation of eq. (9.10) per Problem 4, and is denoted as `s-LQG`. From each of the selected sensor sets, by each of the above four algorithms respectively, we consider an optimal sensor removal, which we compute via brute-force.

Results. We next present our simulation results averaged over 20 Monte Carlo runs of the above simulation setup, where we vary the number of sensor selections α from 2 up to 12 with step 1, and the number β of sensors failures from 1 to 10 with step 3, and where we randomize the sensor matrices of the 12 ground sensors. In particular, the results of our numerical analysis are reported in Fig. 12. In more detail, Fig. 12 shows the attained LQG cost for all the combinations of α and β values where $\beta \leq \alpha$ (for $\beta > \alpha$ the LQG cost is considered $+\infty$, since $\beta > \alpha$ implies that all α selected sensors fail). The following observations from Fig. 12 are due:

- *Near-optimality of the* `s-LQG` *Algorithm 19*: Algorithm 19 —blue colour in Fig. 12— performs close to the optimal algorithm `optimal`—green colour in Fig. 12. In particular, across all but two scenarios in Fig. 12, Algorithm 19 achieves an approximation performance at least 97% the optimal, and 90% the optimal in the two scenarios in Fig. 12-(a) where α is 3 or 4, and β is 1.

- *Performance of the* `logdet` *algorithm*: The `logdet`algorithm —red colour in Fig. 12— performs poorly as the number β of sensor failures increases, which is expected, given that the `logdet`algorithm minimizes the cost function in eq. (9.10) ignoring the possibility of sensor failures. Notably, for some of the cases, the `logdet`performs worse or equally poor as the `random`*: for example, see Fig. 12-(c) for $\alpha \geq 9$, and Fig. 12-(d).

- *Performance of the* `random`* *algorithm*: Expectedly, the performance of also the `random`* algorithm —black colour in Fig. 12— is poor across all scenarios in Fig. 12.

Overall, in the above numerical experiments, Algorithm 19 demonstrates a close-to-optimal approximation performance, and the necessity for a resilient re-formulation of the optimization problem in eq. (9.1), e.g., per Problem 4, is exemplified.

9.6. Concluding Remarks & Future Work

We made the first step to ensure the success of critical missions in control, robotics, and optimization that involve the design of systems subject to complex optimization constraints of heterogeneity and global-interdependency —called matroid constraints— against worst-case denial-of-service attacks, failures, or deletions. In particular, we provided the first algorithm for Problem 4, which, with minimal running time, guarantees a close-to-optimal performance against system-wide attacks, failures and deletions. To quantify the algorithm's approximation performance, we exploited a notion of curvature for monotone (not necessarily submodular) set functions, and contributed a first step towards characterizing the curvature's effect on the approximability of resilient *matroid-constrained* maximization. Our curvature-dependent characterizations complement the current knowledge on the curvature's effect on the approximability of simpler problems, such as of non-matroid-constrained resilient maximization [35, 56, 231], and of non-resilient maximization [31, 32, 33]. Finally, we supported our theoretical analyses with numerical experiments.

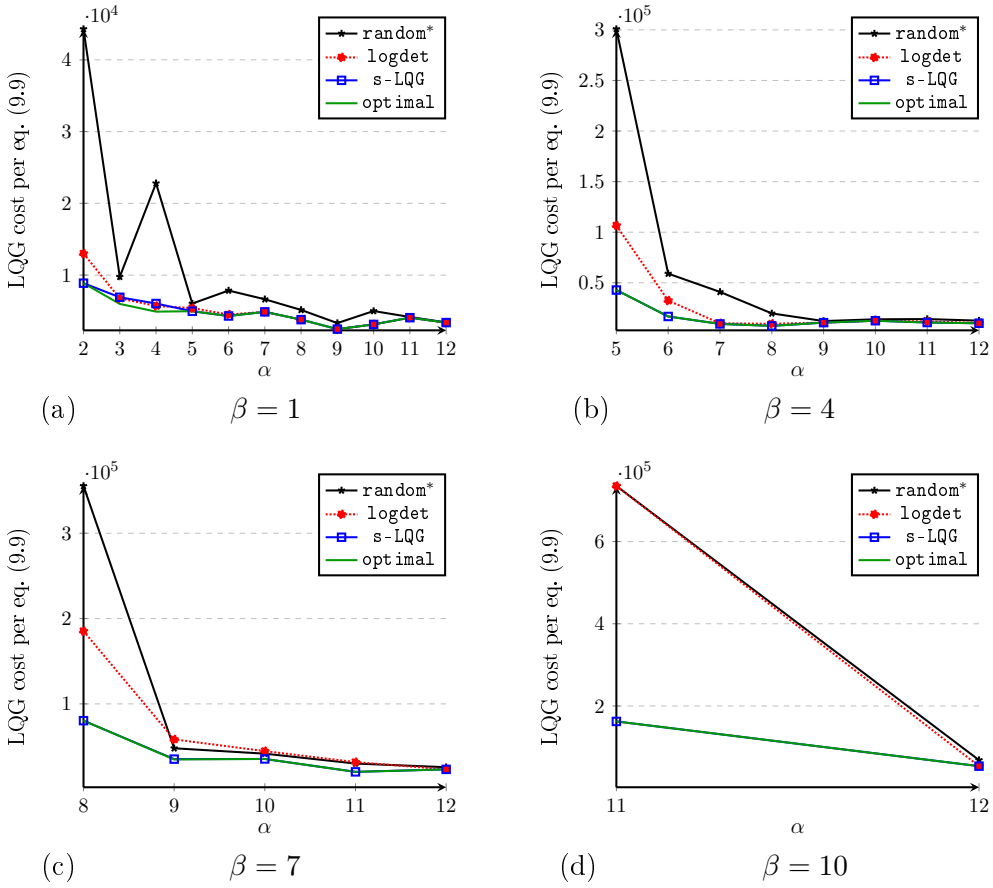


Figure 12: LQG cost for increasing number of sensor selections α (from 2 up to 12 with step 1), and for 4 values of β (number of sensor failures among the α selected sensors); in particular, the value of β varies across the sub-figures as follows: $\beta = 1$ in sub-figure (a); $\beta = 4$ in sub-figure (b); $\beta = 7$ in sub-figure (c); and $\beta = 10$ in sub-figure (d).

This chapter opens several avenues for future research, both in theory and in applications. Future work in theory includes the extension of our results to sequential (multi-step) maximization, per the recent developments in [231], to enable applications of sensor scheduling and of path planning in online optimization that *adapts* against persistent attacks and failures [6, 182]. Future work in applications includes the experimental testing of the proposed algorithm in applications of motion-planning for multi-target covering with mobile vehicles [6], to enable resiliency in critical scenarios of surveillance.

9.7. Appendix: Proof of Results

9.7.1. Notation

In the appendices below we use the following notation: given a finite ground set $\mathcal{V}$, and a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$, then, for any sets $\mathcal{X} \subseteq \mathcal{V}$ and $\mathcal{X}' \subseteq \mathcal{V}$:

$$f(\mathcal{X}|\mathcal{X}') \triangleq f(\mathcal{X} \cup \mathcal{X}') - f(\mathcal{X}').$$

Moreover, let the set $\mathcal{A}^\star$ denote an (optimal) solution to Problem 4; formally:

$$\mathcal{A}^\star \in \arg \max_{\mathcal{A} \subseteq \mathcal{V}, \mathcal{A} \in \mathcal{I}} \min_{\mathcal{B} \subseteq \mathcal{A}, \mathcal{B} \in \mathcal{I}'(\mathcal{A})} f(\mathcal{A} \setminus \mathcal{B}).$$

9.7.2. Preliminary lemmas

We list lemmas that support the proof of Theorem 22.[5]

Lemma 29. *Consider any finite ground set $\mathcal{V}$, a non-decreasing submodular function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$, and non-empty sets $\mathcal{Y}, \mathcal{P} \subseteq \mathcal{V}$ such that for all elements $y \in \mathcal{Y}$, and all elements $p \in \mathcal{P}$, it is $f(y) \geq f(p)$. Then:*

$$f(\mathcal{P}|\mathcal{Y}) \leq |\mathcal{P}| f(\mathcal{Y}).$$

Proof of Lemma 29 Consider any element $y \in \mathcal{Y}$; then:

$$\begin{aligned} f(\mathcal{P}|\mathcal{Y}) &= f(\mathcal{P} \cup \mathcal{Y}) - f(\mathcal{Y}) && (9.11)\\ &\leq f(\mathcal{P}) + f(\mathcal{Y}) - f(\mathcal{Y}) && (9.12)\\ &= f(\mathcal{P}) \\ &\leq \sum_{p \in \mathcal{P}} f(p) && (9.13)\\ &\leq |\mathcal{P}| \max_{p \in \mathcal{P}} f(p) \\ &\leq |\mathcal{P}| f(y) && (9.14)\\ &\leq |\mathcal{P}| f(\mathcal{Y}), && (9.15) \end{aligned}$$

where eqs. (9.11)-(9.15) hold for the following reasons: eq. (9.11) holds since for any sets $\mathcal{X} \subseteq \mathcal{V}$ and $\mathcal{Y} \subseteq \mathcal{V}$, it is $f(\mathcal{X}|\mathcal{Y}) = f(\mathcal{X} \cup \mathcal{Y}) - f(\mathcal{Y})$; ineq. (9.12) holds since f is submodular and, as a result, the submodularity Definition 33 implies that for any set $\mathcal{A} \subseteq \mathcal{V}$ and $\mathcal{A}' \subseteq \mathcal{V}$, it is $f(\mathcal{A} \cup \mathcal{A}') \leq f(\mathcal{A}) + f(\mathcal{A}')$ [70, Proposition 2.1]; ineq. (9.13) holds for the same reason as ineq. (9.12); ineq. (9.14) holds since for all elements $y \in \mathcal{Y}$, and for all elements $p \in \mathcal{P}$, it is $f(y) \geq f(p)$; finally, ineq. (9.15) holds since f is monotone, and since $y \in \mathcal{Y}$. ■

Lemma 30. *Consider a finite ground set $\mathcal{V}$, and a non-decreasing submodular set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any $\mathcal{A} \subseteq \mathcal{V}$, it holds:*

$$f(\mathcal{A}) \geq (1 - \kappa_f) \sum_{a \in \mathcal{A}} f(a).$$

Proof of Lemma 30 Let $\mathcal{A} = \{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$. We prove Lemma 30 by proving the following two inequalities:

$$f(\mathcal{A}) \geq \sum_{i=1}^{|\mathcal{A}|} f(a_i | \mathcal{V} \setminus \{a_i\}), \tag{9.16}$$

$$\sum_{i=1}^{|\mathcal{A}|} f(a_i | \mathcal{V} \setminus \{a_i\}) \geq (1 - \kappa_f) \sum_{i=1}^{|\mathcal{A}|} f(a_i). \tag{9.17}$$

[5] The proof of Lemmas 29-33 and of Corollary 10 is also found in [56] and [231].

We begin with the proof of ineq. (9.16):

$$f(\mathcal{A}) = f(\mathcal{A}|\emptyset) \tag{9.18}$$
$$\geq f(\mathcal{A}|\mathcal{V} \setminus \mathcal{A}) \tag{9.19}$$
$$= \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V} \setminus \{a_i, a_{i+1}, \ldots, a_{|\mathcal{A}|}\}) \tag{9.20}$$
$$\geq \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V} \setminus \{a_i\}), \tag{9.21}$$

where ineqs. (9.19)-(9.21) hold for the following reasons: ineq. (9.19) is implied by eq. (9.18) because f is submodular and $\emptyset \subseteq \mathcal{V} \setminus \mathcal{A}$; eq. (9.20) holds since for any sets $\mathcal{X} \subseteq \mathcal{V}$ and $\mathcal{Y} \subseteq \mathcal{V}$ it is $f(\mathcal{X}|\mathcal{Y}) = f(\mathcal{X} \cup \mathcal{Y}) - f(\mathcal{Y})$, and since $\{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ denotes the set $\mathcal{A}$; and ineq. (9.21) holds since f is submodular, and since $\mathcal{V} \setminus \{a_i, a_{i+1}, \ldots, a_\mu\} \subseteq \mathcal{V} \setminus \{a_i\}$. These observations complete the proof of ineq. (9.16).

We now prove ineq. (9.17) using the Definition 34 of κ_f, as follows: since $\kappa_f = 1 - \min_{v \in \mathcal{V}} \frac{f(v|\mathcal{V}\setminus\{v\})}{f(v)}$, it is implied that for all elements $v \in \mathcal{V}$ it is $f(v|\mathcal{V} \setminus \{v\}) \geq (1 - \kappa_f) f(v)$. Therefore, by adding the latter inequality across all elements $a \in \mathcal{A}$, we complete the proof of ineq. (9.17). ■

Lemma 31. *Consider a finite ground set $\mathcal{V}$, and a monotone set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any sets $\mathcal{A} \subseteq \mathcal{V}$ and $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, it holds:*

$$f(\mathcal{A} \cup \mathcal{B}) \geq (1 - c_f) \left(f(\mathcal{A}) + f(\mathcal{B})\right).$$

Proof of Lemma 31 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i|\mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}). \tag{9.22}$$

The definition of total curvature in Definition 35 implies:

$$f(b_i|\mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}) \geq$$
$$(1 - c_f) f(b_i|\{b_1, b_2, \ldots, b_{i-1}\}). \tag{9.23}$$

The proof is completed by substituting ineq. (9.23) in eq. (9.22) and then by taking into account that it holds $f(\mathcal{A}) \geq (1 - c_f) f(\mathcal{A})$, since $0 \leq c_f \leq 1$. ■

Lemma 32. *Consider a finite ground set $\mathcal{V}m$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, it holds:*

$$f(\mathcal{A} \cup \mathcal{B}) \geq (1 - c_f) \left(f(\mathcal{A}) + \sum_{b \in \mathcal{B}} f(b)\right).$$

Proof of Lemma 32 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i | \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}). \tag{9.24}$$

In addition, Definition 35 of total curvature implies:

$$f(b_i | \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}) \geq (1 - c_f) f(b_i | \emptyset) = (1 - c_f) f(b_i), \tag{9.25}$$

where the latter equation holds since $f(\emptyset) = 0$. The proof is completed by substituting (9.25) in (9.24) and then taking into account that $f(\mathcal{A}) \geq (1 - c_f) f(\mathcal{A})$ since $0 \leq c_f \leq 1$. ■

Lemma 33. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \setminus \mathcal{B} \neq \emptyset$, it holds:*

$$f(\mathcal{A}) + (1 - c_f) f(\mathcal{B}) \geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}) + f(\mathcal{A} \cap \mathcal{B}).$$

Proof of Lemma 33 Let $\mathcal{A} \setminus \mathcal{B} = \{i_1, i_2, \ldots, i_r\}$, where $r = |\mathcal{A} - \mathcal{B}|$. From Definition 35 of total curvature c_f, for any $i = 1, 2, \ldots, r$, it is $f(i_j | \mathcal{A} \cap \mathcal{B} \cup \{i_1, i_2, \ldots, i_{j-1}\}) \geq (1 - c_f) f(i_j | \mathcal{B} \cup \{i_1, i_2, \ldots, i_{j-1}\})$. Summing these r inequalities:

$$f(\mathcal{A}) - f(\mathcal{A} \cap \mathcal{B}) \geq (1 - c_f) \left(f(\mathcal{A} \cup \mathcal{B}) - f(\mathcal{B}) \right),$$

which implies the lemma. ■

Corollary 10. *Consider a finite ground set $\mathcal{V}m$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, it holds:*

$$f(\mathcal{A}) + \sum_{b \in \mathcal{B}} f(b) \geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}).$$

Proof of Corollary 10 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$.

$$\begin{aligned}
f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i) &\geq (1 - c_f) f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i)) \qquad (9.26) \\
&\geq (1 - c_f) f(\mathcal{A} \cup \{b_1\}) + \sum_{i=2}^{|\mathcal{B}|} f(b_i) \\
&\geq (1 - c_f) f(\mathcal{A} \cup \{b_1, b_2\}) + \sum_{i=3}^{|\mathcal{B}|} f(b_i) \\
&\vdots \\
&\geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}),
\end{aligned}$$

where (9.26) holds since $0 \leq c_f \leq 1$, and the rest due to Lemma 33, since $\mathcal{A} \cap \mathcal{B} = \emptyset$ implies $\mathcal{A} \setminus \{b_1\} \neq \emptyset$, $\mathcal{A} \cup \{b_1\} \setminus \{b_2\} \neq \emptyset$, ..., $\mathcal{A} \cup \{b_1, b_2, \ldots, b_{|\mathcal{B}|-1}\} \setminus \{b_{|\mathcal{B}|}\} \neq \emptyset$. ■

Lemma 34. *Recall the notation in Algorithm 19, and consider the sets $\mathcal{A}_1$ and $\mathcal{A}_2$ constructed by Algorithm 19's lines 2-8 and lines 9-15, respectively. Then, for all elements $v \in \mathcal{A}_1$ and all elements $v' \in \mathcal{A}_2$, it holds $f(v) \geq f(v')$.*

Proof of Lemma 34 Let $v_1, \ldots, v_{|\mathcal{A}_1|}$ be the elements in $\mathcal{A}_1$, —i.e., $\mathcal{A}_1 \equiv \{v_1, \ldots, v_{|\mathcal{A}_1|}\}$,— and be such that for each $i = 1, \ldots, |\mathcal{A}_1|$ the element v_i is the i-th element added in $\mathcal{A}_1$ per Algorithm 19's lines 2-8; similarly, let $v'_1, \ldots, v'_{|\mathcal{A}_2|}$ be the elements in $\mathcal{A}_2$, —i.e., $\mathcal{A}_2 \equiv \{v'_1, \ldots, v'_{|\mathcal{A}_2|}\}$,— and be such that for each $i = 1, \ldots, |\mathcal{A}_2|$ the element v'_i is the i-th element added in $\mathcal{A}_2$ per Algorithm 19's lines 9-15.

We prove Lemma 34 by the method of contradiction. In particular, assume that there exists an index $i \in \{1, \ldots, |\mathcal{A}_1|\}$ and an $j \in \{1, \ldots, |\mathcal{A}_2|\}$ such that $f(v'_j) > f(v_i)$, and, in particular, assume that i, j are the smallest indexes such that $f(v'_j) > f(v_i)$. Since Algorithm 19 constructs $\mathcal{A}_1$ and $\mathcal{A}_2$ such that $\mathcal{A}_1 \cup \mathcal{A}_2 \in \mathcal{I}$, and since it also is that $(\mathcal{V}, \mathcal{I})$ is a matroid and $\{v_1, \ldots, v_{i-1}, v'_j\} \subseteq \mathcal{A}_1 \cup \mathcal{A}_2$, we have that $\{v_1, \ldots, v_{i-1}, v'_j\} \in \mathcal{I}$. In addition, we have that $\{v_1, \ldots, v_{i-1}, v'_j\} \in \mathcal{I}'$, since $\mathcal{I}'$ is either a uniform or a partition matroid and, as a result, if $\{v_1, \ldots, v_{i-1}, v_i\} \in \mathcal{I}'$ then it also is $\{v_1, \ldots, v_{i-1}, v\} \in \mathcal{I}'$ for any $v \in \mathcal{V} \setminus \{v_1, \ldots, v_{i-1}\}$. Overall, $\{v_1, \ldots, v_{i-1}, v'_j\} \in \mathcal{I}, \mathcal{I}'$. Now, consider the "while loop" in Algorithm 19's lines 2-8 at the beginning of its i-th iteration, that is, when Algorithm 19 has chosen only the elements $\{v_1, \ldots, v_{i-1}\}$ among the elements in $\mathcal{A}_1$. Then, per Algorithm 19's lines 3-5, the next element v that is added in $\{v_1, \ldots, v_{i-1}\}$ is the one that achieves the highest value of $f(v')$ among all elements in $v' \in \mathcal{V} \setminus \{v_1, \ldots, v_{i-1}\}$, and which satisfies $\{v_1, \ldots, v_{i-1}, v\} \in \mathcal{I}, \mathcal{I}'$. Therefore, the next element v that is added in $\{v_1, \ldots, v_{i-1}\}$ cannot be v_i, since $f(v'_j) > f(v_i)$ and $\{v_1, \ldots, v_{i-1}, v'_j\} \in \mathcal{I}, \mathcal{I}'$. ■

Lemma 35. *Consider a matroid $(\mathcal{V}, \mathcal{I})$, and a set $\mathcal{Y} \subseteq \mathcal{V}$ such that $\mathcal{Y} \in \mathcal{I}$. Moreover, define the following collection of subsets of $\mathcal{V} \setminus \mathcal{Y}$: $\mathcal{I}' \triangleq \{\mathcal{X} : \mathcal{X} \subseteq \mathcal{V} \setminus \mathcal{Y}, \mathcal{X} \cup \mathcal{Y} \in \mathcal{I}\}$. Then, $(\mathcal{V} \setminus \mathcal{Y}, \mathcal{I}')$ is a matroid.*

Proof of Lemma 35 We validate that $(\mathcal{V} \setminus \mathcal{Y}, \mathcal{I}')$ satisfies the conditions in Definition 31 of a matroid. In particular:

- to validate the first condition in Definition 31, assume a set $\mathcal{X} \subseteq \mathcal{V} \setminus \mathcal{Y}$ such that $\mathcal{X} \in \mathcal{I}'$; moreover, assume a set $\mathcal{Z} \subseteq \mathcal{X}$; we need to show that $\mathcal{Z} \in \mathcal{I}'$. To this end, observe that the definition of $\mathcal{I}'$ implies $\mathcal{X} \cup \mathcal{Y} \in \mathcal{I}$, since we assumed $\mathcal{X} \in \mathcal{I}'$. In addition, the assumption $\mathcal{Z} \subseteq \mathcal{X}$ implies $\mathcal{Z} \cup \mathcal{Y} \subseteq \mathcal{X} \cup \mathcal{Y}$, and, as a result, $\mathcal{Z} \cup \mathcal{Y} \in \mathcal{I}$, since $(\mathcal{V}, \mathcal{I})$ is a matroid. Overall, $\mathcal{Z} \subseteq \mathcal{V} \setminus \mathcal{Y}$ (since $\mathcal{Z} \subseteq \mathcal{X}$, by assumption, and $\mathcal{X} \subseteq \mathcal{V} \setminus \mathcal{Y}$) and $\mathcal{Z} \cup \mathcal{Y} \in \mathcal{I}$; hence, $\mathcal{Z} \in \mathcal{I}'$, by the definition of $\mathcal{I}'$, and now the first condition in Definition 31 is validated;

- to validate the second condition in Definition 31, assume sets $\mathcal{X}, \mathcal{Z} \in \mathcal{V} \setminus \mathcal{Y}$ such that $\mathcal{X}, \mathcal{Z} \in \mathcal{I}'$ and $|\mathcal{X}| < |\mathcal{Z}|$; we need to show that there exists an element $z \in \mathcal{Z} \setminus \mathcal{X}$ such that $\mathcal{X} \cup \{z\} \in \mathcal{I}'$. To this end, observe that since $\mathcal{X}, \mathcal{Z} \in \mathcal{I}'$, the definition of $\mathcal{I}'$ implies that $\mathcal{X} \cup \mathcal{Y}, \mathcal{Z} \cup \mathcal{Y} \in \mathcal{I}$. Moreover, since $|\mathcal{X}| < |\mathcal{Z}|$, it also is $|\mathcal{X} \cup \mathcal{Y}| < |\mathcal{Z} \cup \mathcal{Y}|$. Therefore, since $(\mathcal{V}, \mathcal{I})$ is a matroid, there exists an element $z \in (\mathcal{Z} \cup \mathcal{Y}) \setminus (\mathcal{X} \cup \mathcal{Y}) = \mathcal{Z} \setminus \mathcal{X}$ such that $(\mathcal{X} \cup \mathcal{Y}) \cup \{z\} \in \mathcal{I}$; as a result, $\mathcal{X} \cup \{z\} \in \mathcal{I}'$, by the definition of $\mathcal{I}'$. In sum, $z \in \mathcal{Z} \setminus \mathcal{X}$ and $\mathcal{X} \cup \{z\} \in \mathcal{I}'$, and the second condition in Definition 31 is validated too.

■

Lemma 36. *Recall the notation in Algorithm 19, and consider the sets $\mathcal{A}_1$ and $\mathcal{A}_2$ constructed by Algorithm 19's lines 2-8 and lines 9-15, respectively. Then, for the set $\mathcal{A}_2$, it*

holds:

- *if the function f is non-decreasing submodular and:*
  - *if $(\mathcal{V},\mathcal{I})$ is a uniform matroid, then:*

$$f(\mathcal{A}_2) \geq \frac{1}{\kappa_f}(1-e^{-\kappa_f}) \max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X}). \tag{9.27}$$

  - *if $(\mathcal{V},\mathcal{I})$ is a matroid, then:*

$$f(\mathcal{A}_2) \geq \frac{1}{1+\kappa_f} \max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X}). \tag{9.28}$$

- *if the function f is non-decreasing, then:*

$$f(\mathcal{A}_2) \geq (1-c_f) \max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X}). \tag{9.29}$$

Proof of Lemma 36 We first prove ineq. (9.29), then ineq. (9.28), and, finally, ineq. (9.27). In particular, Algorithm 19 constructs the set $\mathcal{A}_2$ greedily, by replicating the steps of the greedy algorithm introduced [12, Section 2], to solve the following optimization problem:

$$\max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X}); \tag{9.30}$$

let in the latter problem $\mathcal{I}' \triangleq \{\mathcal{X} \subseteq \mathcal{V}\setminus\mathcal{A}_1, \mathcal{X}\cup\mathcal{A}_1 \in \mathcal{I}\}$. Lemma 35 implies that $(\mathcal{A}_1,\mathcal{I}')$ is a matroid, and, as a result, the previous optimization problem is a matroid-constrained set function maximization problem. Now, to prove ineq. (9.29), ineq. (9.28), and ineq. (9.27), we make the following observations, respectively: when the function f is merely non-decreasing, then [15, Theorem 8.1] implies that the greedy algorithm introduced in [12, Section 2] returns for the optimization problem in eq. (9.30) a solution $\mathcal{S}$ such that $f(\mathcal{S}) \geq (1-c_f)\max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X})$; this proves ineq. (9.29). Similarly, when the function f is non-decreasing and submodular, then [33, Theorem 2.3] implies that the greedy algorithm introduced in [12, Section 2] returns for the optimization problem in eq. (9.30) a solution $\mathcal{S}$ such that $f(\mathcal{S}) \geq 1/(1+\kappa_f)\max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X})$; this proves ineq. (9.28). Finally, when the objective function f is non-decreasing submodular, and when $\mathcal{I}$ is a uniform matroid, then [33, Theorem 5.4] implies that the greedy algorithm introduced in [12, Section 2] returns for the optimization problem in eq. (9.30) a solution $\mathcal{S}$ such that $f(\mathcal{S}) \geq 1/\kappa_f(1-e^{-\kappa_f})\max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{A}_1,\mathcal{X}\cup\mathcal{A}_1\in\mathcal{I}} f(\mathcal{X})$; this proves ineq. (9.27), and concludes the proof of the lemma. ■

Lemma 37. *Recall the notation in Theorem 22 and Appendix 9.7.1. Also, consider a uniform or partition matroid $(\mathcal{V},\mathcal{I}')$. Then, for any set $\mathcal{Y}\subseteq\mathcal{V}$ such that $\mathcal{Y}\in\mathcal{I}$ and $\mathcal{Y}\in\mathcal{I}'$, it holds:*

$$\max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{Y},\mathcal{X}\cup\mathcal{Y}\in\mathcal{I}} f(\mathcal{X}) \geq f(\mathcal{A}^\star\setminus\mathcal{B}^\star(\mathcal{A}^\star)). \tag{9.31}$$

Proof of Lemma 37 We start from the left-hand-side of ineq. (9.31), and make the following observations:

$$\begin{aligned}\max_{\mathcal{X}\subseteq\mathcal{V}\setminus\mathcal{Y},\mathcal{X}\cup\mathcal{Y}\in\mathcal{I}} \; f(\mathcal{X}) &\geq \min_{\bar{\mathcal{Y}}\subseteq\mathcal{V},\bar{\mathcal{Y}}\in\mathcal{I},\mathcal{I}'} \; \max_{\mathcal{X}\subseteq\mathcal{V}\setminus\bar{\mathcal{Y}},\mathcal{X}\cup\bar{\mathcal{Y}}\in\mathcal{I}} \; f(\mathcal{X}) \\ &= \min_{\bar{\mathcal{Y}}\subseteq\mathcal{V},\bar{\mathcal{Y}}\in\mathcal{I},\mathcal{I}'} \; \max_{\bar{\mathcal{A}}\subseteq\mathcal{V},\bar{\mathcal{A}}\in\mathcal{I}} \; f(\bar{\mathcal{A}}\setminus\bar{\mathcal{Y}}) \\ &\triangleq h.\end{aligned}$$

We next complete the proof of Lemma 37 by proving that $h \geq f(\mathcal{A}^\star \setminus \mathcal{B}^\star(\mathcal{A}^\star))$. To this end, observe that for any set $\mathcal{A} \subseteq \mathcal{V}$ such that $\mathcal{A} \in \mathcal{I}$, and for any set $\mathcal{Y} \subseteq \mathcal{V}$ such that $\mathcal{Y} \in \mathcal{I}$ and $\mathcal{Y} \in \mathcal{I}'$, it holds:

$$\max_{\bar{\mathcal{A}}\subseteq\mathcal{V},\bar{\mathcal{A}}\in\mathcal{I}} \; f(\bar{\mathcal{A}}\setminus\mathcal{Y}) \geq f(\mathcal{A}\setminus\mathcal{Y}),$$

which implies the following observations:

$$\begin{aligned}h &\geq \min_{\bar{\mathcal{Y}}\subseteq\mathcal{V},\bar{\mathcal{Y}}\in\mathcal{I},\mathcal{I}'} \; f(\mathcal{A}\setminus\bar{\mathcal{Y}}) \\ &\geq \min_{\bar{\mathcal{Y}}\subseteq\mathcal{V},\bar{\mathcal{Y}}\in\mathcal{I}'} \; f(\mathcal{A}\setminus\bar{\mathcal{Y}}) \\ &= \min_{\bar{\mathcal{Y}}\subseteq\mathcal{A},\bar{\mathcal{Y}}\in\mathcal{I}'} \; f(\mathcal{A}\setminus\bar{\mathcal{Y}}),\end{aligned}$$

and, as a result, it holds:

$$\begin{aligned}h &\geq \max_{\bar{\mathcal{A}}\subseteq\mathcal{V},\bar{\mathcal{A}}\in\mathcal{I}} \; \min_{\bar{\mathcal{Y}}\subseteq\mathcal{A},\bar{\mathcal{Y}}\in\mathcal{I}'} \; f(\bar{\mathcal{A}}\setminus\bar{\mathcal{Y}}) \\ &= f(\mathcal{A}^\star \setminus \mathcal{B}^\star(\mathcal{A}^\star)).\end{aligned}$$

■

9.7.3. Proof of Theorem 22

We first prove Theorem 22's part 1 (approximation performance), and then, Theorem 22's part 2 (running time).

Proof of Theorem 22's part 1 (approximation performance)

We first prove ineq. (9.8), and, then, ineq. (9.7) and ineq. (9.6).

To the above ends, we use the following notation (along with the notation in Algorithm 19, Theorem 22, and Appendix 9.7.1):

- let $\mathcal{A}_1^+ \triangleq \mathcal{A}_1 \setminus \mathcal{B}^\star(\mathcal{A})$, i.e., $\mathcal{A}_1^+$ is the set of remaining elements in the set $\mathcal{A}_1$ after the removal from $\mathcal{A}_1$ of the elements in the optimal (worst-case) removal $\mathcal{B}^\star(\mathcal{A})$;
- let $\mathcal{A}_2^+ \triangleq \mathcal{A}_2 \setminus \mathcal{B}^\star(\mathcal{A})$, i.e., $\mathcal{A}_2^+$ is the set of remaining elements in the set $\mathcal{A}_2$ after the removal from $\mathcal{A}_2$ of the elements in the optimal (worst-case) removal $\mathcal{B}^\star(\mathcal{A})$.

Proof of ineq. (9.8) Consider that the objective function f is non-decreasing and such that (without loss of generality) f is non-negative and $f(\emptyset) = 0$. Then, the proof of ineq. (9.8)

follows by making the following observations:

$$
\begin{aligned}
& f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) \\
& \quad = f(\mathcal{A}_1^+ \cup \mathcal{A}_2^+) && (9.32)\\
& \quad \geq (1 - c_f) \sum_{v \in \mathcal{A}_1^+ \cup \mathcal{A}_2^+} f(v) && (9.33)\\
& \quad \geq (1 - c_f) \sum_{v \in \mathcal{A}_2} f(v) && (9.34)\\
& \quad \geq (1 - c_f)^2 f(\mathcal{A}_2) && (9.35)\\
& \quad \geq (1 - c_f)^3 \max_{\mathcal{X} \subseteq \mathcal{V} \setminus \mathcal{A}_1, \mathcal{X} \cup \mathcal{A}_1 \in \mathcal{I}} f(\mathcal{X}) && (9.36)\\
& \quad \geq (1 - c_f)^3 f(\mathcal{A}^\star \setminus \mathcal{B}^\star(\mathcal{A}^\star)), && (9.37)
\end{aligned}
$$

where eqs. (9.32)-(9.37) hold for the following reasons: eq. (9.32) follows from the definitions of the sets $\mathcal{A}_1^+$ and $\mathcal{A}_2^+$; ineq. (9.33) follows from ineq. (9.32) due to Lemma 32; ineq. (9.34) follows from ineq. (9.33) due to Lemma 34, which implies that for any element $v \in \mathcal{A}_1^+$ and any element $v' \in \mathcal{A}_2^+$ it is $f(v) \geq f(v')$ —note that due to the definitions of $\mathcal{A}_1^+$ and $\mathcal{A}_2^+$ it is $|\mathcal{A}_1^+| = |\mathcal{A}_2 \setminus \mathcal{A}_2^+|$, that is, the number of non-removed elements in $\mathcal{A}_1$ is equal to the number of removed elements in $\mathcal{A}_2$,— and the fact $\mathcal{A}_2 = (\mathcal{A}_2 \setminus \mathcal{A}_2^+) \cup \mathcal{A}_2^+$; ineq. (9.35) follows from ineq. (9.34) due to Corollary 10; ineq. (9.36) follows from ineq. (9.35) due to Lemma 36's ineq. (9.29); finally, ineq. (9.37) follows from ineq. (9.36) due to Lemma 37. ■

In what follows, we first prove ineq. (9.7), and then ineq. (9.6): we first prove the part $\frac{1-\kappa_f}{1+\kappa_f}$ and $\frac{1-\kappa_f}{\kappa_f}(1 - e^{-\kappa_f})$ of ineq. (9.7) and of ineq. (9.6), respectively, and then, the part $\frac{h_f(\alpha,\beta)}{1+\kappa_f}$ and $\frac{h_f(\alpha,\beta)}{\kappa_f}(1 - e^{-\kappa_f})$ of ineq. (9.7) and of ineq. (9.6), respectively.

Proof of part $(1 - \kappa_f)/(1 + \kappa_f)$ of ineq. (9.7) Consider that the objective function f is non-decreasing submodular and such that (without loss of generality) f is non-negative and $f(\emptyset) = 0$. To prove the part $(1 - \kappa_f)/(1 + \kappa_f)$ of ineq. (9.7) we follow similar observations

to the ones we followed in the proof of ineq. (9.8); in particular:

$$
\begin{aligned}
& f(\mathcal{A} \setminus \mathcal{B}^{\star}(\mathcal{A})) \\
& = f(\mathcal{A}_1^{+} \cup \mathcal{A}_2^{+}) && (9.38)\\
& \geq (1-\kappa_f) \sum_{v \in \mathcal{A}_1^{+} \cup \mathcal{A}_2^{+}} f(v) && (9.39)\\
& \geq (1-\kappa_f) \sum_{v \in \mathcal{A}_2} f(v) && (9.40)\\
& \geq (1-\kappa_f) f(\mathcal{A}_2) && (9.41)\\
& \geq \frac{1-\kappa_f}{1+\kappa_f} \max_{\mathcal{X} \subseteq \mathcal{V} \setminus \mathcal{A}_1, \mathcal{X} \cup \mathcal{A}_1 \in \mathcal{I}} \quad f(\mathcal{X}) && (9.42)\\
& \geq \frac{1-\kappa_f}{1+\kappa_f} f(\mathcal{A}^{\star} \setminus \mathcal{B}^{\star}(\mathcal{A}^{\star})), && (9.43)
\end{aligned}
$$

where eqs. (9.38)-(9.43) hold for the following reasons: eq. (9.38) follows from the definitions of the sets $\mathcal{A}_1^{+}$ and $\mathcal{A}_2^{+}$; ineq. (9.39) follows from ineq. (9.38) due to Lemma 30; ineq. (9.40) follows from ineq. (9.39) due to Lemma 34, which implies that for any element $v \in \mathcal{A}_1^{+}$ and any element $v' \in \mathcal{A}_2^{+}$ it is $f(v) \geq f(v')$ —note that due to the definitions of the sets $\mathcal{A}_1^{+}$ and $\mathcal{A}_2^{+}$ it is $|\mathcal{A}_1^{+}| = |\mathcal{A}_2 \setminus \mathcal{A}_2^{+}|$, that is, the number of non-removed elements in $\mathcal{A}_1$ is equal to the number of removed elements in $\mathcal{A}_2$,— and because $\mathcal{A}_2 = (\mathcal{A}_2 \setminus \mathcal{A}_2^{+}) \cup \mathcal{A}_2^{+}$; ineq. (9.41) follows from ineq. (9.40) because the set function f is submodular, and as a result, the submodularity Definition 33 implies that for any sets $\mathcal{S} \subseteq \mathcal{V}$ and $\mathcal{S}' \subseteq \mathcal{V}$, it is $f(\mathcal{S}) + f(\mathcal{S}') \geq f(\mathcal{S} \cup \mathcal{S}')$ [70, Proposition 2.1]; ineq. (9.42) follows from ineq. (9.41) due to Lemma 36's ineq. (9.28); finally, ineq. (9.43) follows from ineq. (9.42) due to Lemma 37. ■

Proof of part $(1-\kappa_f)/\kappa_f(1-e^{-\kappa_f})$ of ineq. (9.6) Consider that the objective function f is non-decreasing submodular and such that (without loss of generality) f is non-negative and $f(\emptyset) = 0$. Moreover, consider that the pair $(\mathcal{V}, \mathcal{I})$ is a uniform matroid. To prove the part $(1-\kappa_f)/\kappa_f(1-e^{-\kappa_f})$ of ineq. (9.6) we follow similar steps to the ones we followed in the proof of ineq. (9.7) via the ineqs. (9.38)-(9.43). We explain next where these steps differ: if instead of using Lemma 36's ineq. (9.28) to get ineq. (9.42) from ineq. (9.41), we use Lemma 36's ineq. (9.27), and afterwards apply Lemma 37, then, we derive ineq. (9.6). ■

Proof of parts $h_f(\alpha,\beta)/(1+\kappa_f)$ and $h_f(\alpha,\beta)/\kappa_f(1-e^{-\kappa_f})$ of ineq. (9.7) and ineq. (9.6), respectively We complete the proof by first proving that:

$$
f(\mathcal{A} \setminus \mathcal{B}^{\star}(\mathcal{A})) \geq \frac{1}{1+\beta} f(\mathcal{A}_2), \tag{9.44}
$$

and, then, proving that:

$$
f(\mathcal{A} \setminus \mathcal{B}^{\star}(\mathcal{A})) \geq \frac{1}{\alpha-\beta} f(\mathcal{A}_2). \tag{9.45}
$$

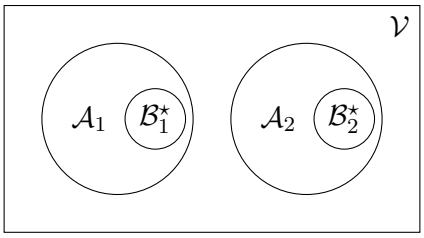


Figure 13: Venn diagram, where the sets $\mathcal{A}_1, \mathcal{A}_2, \mathcal{B}_1^\star, \mathcal{B}_2^\star$ are as follows: per Algorithm 19, $\mathcal{A}_1$ and $\mathcal{A}_2$ are such that $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$. Due to their construction, it holds $\mathcal{A}_1 \cap \mathcal{A}_2 = \emptyset$. Next, $\mathcal{B}_1^\star$ and $\mathcal{B}_2^\star$ are such that $\mathcal{B}_1^\star = \mathcal{B}^\star(\mathcal{A}) \cap \mathcal{A}_1$, and $\mathcal{B}_2^\star = \mathcal{B}^\star(\mathcal{A}) \cap \mathcal{A}_2$; therefore, $\mathcal{B}_1^\star \cap \mathcal{B}_2^\star = \emptyset$ and $\mathcal{B}^\star(\mathcal{A}) = (\mathcal{B}_1^\star \cup \mathcal{B}_2^\star)$.

The combination of ineq. (9.44) and ineq. (9.45) proves the part $\frac{h_f(\alpha,\beta)}{1+\kappa_f}$ and $\frac{h_f(\alpha,\beta)}{\kappa_f}(1-e^{-\kappa_f})$ of ineq. (9.7) and of ineq. (9.6), respectively, after also applying Lemma 36's ineq. (9.28) and ineq. (9.27), respectively, and then Lemma 37.

To prove ineq. (9.44), we follow the steps of the proof of [56, Theorem 1], and use the notation introduced in Fig. 13, along with the following notation:

$$\eta = \frac{f(\mathcal{B}_2^\star | \mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A}))}{f(\mathcal{A}_2)}. \tag{9.46}$$

In particular, to prove ineq. (9.44) we focus on the worst-case where $\mathcal{B}_2^\star \neq \emptyset$; the reason is that if we assume otherwise, i.e., if we assume $\mathcal{B}_2^\star = \emptyset$, then $f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) = f(\mathcal{A}_2)$, which is a tighter inequality to ineq. (9.44). Hence, considering $\mathcal{B}_2^\star \neq \emptyset$, we prove ineq. (9.44) by first observing that:

$$f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) \geq \max\{f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})), f(\mathcal{A}_1^+)\}, \tag{9.47}$$

and then proving the following three inequalities:

$$f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) \geq (1-\eta) f(\mathcal{A}_2), \tag{9.48}$$

$$f(\mathcal{A}_1^+) \geq \eta \frac{1}{\beta} f(\mathcal{A}_2), \tag{9.49}$$

$$\max\{(1-\eta), \eta\frac{1}{\beta}\} \geq \frac{1}{\beta+1}. \tag{9.50}$$

Specifically, if we substitute ineqs. (9.48)-(9.50) to ineq. (9.47), and take into account that $f(\mathcal{A}_2) \geq 0$, then:

$$f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) \geq \frac{1}{\beta+1} f(\mathcal{A}_2),$$

which implies ineq. (9.44).

We complete the proof of ineq. (9.47) by proving $0 \leq \eta \leq 1$, and ineqs. (9.48)-(9.50), respectively.

Proof of ineq. $0 \leq \eta \leq 1$ We first prove that $\eta \geq 0$, and then, that $\eta \leq 1$: it holds $\eta \geq 0$, since by definition $\eta = f(\mathcal{B}_2^\star|\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A}))/f(\mathcal{A}_2)$, and since f is non-negative; and it holds $\eta \leq 1$, since $f(\mathcal{A}_2) \geq f(\mathcal{B}_2^\star)$, due to monotonicity of f and that $\mathcal{B}_2^\star \subseteq \mathcal{A}_2$, and since $f(\mathcal{B}_2^\star) \geq f(\mathcal{B}_2^\star|\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A}))$, due to submodularity of f and that $\emptyset \subseteq \mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})$.

Proof of ineq. (9.48) We complete the proof of ineq. (9.48) in two steps. First, it can be verified that:

$$\begin{aligned} f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) &= f(\mathcal{A}_2) - \\ f(\mathcal{B}_2^\star|\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) &+ f(\mathcal{A}_1|\mathcal{A}_2) - f(\mathcal{B}_1^\star|\mathcal{A} \setminus \mathcal{B}_1^\star), \end{aligned} \tag{9.51}$$

since for any $\mathcal{X} \subseteq \mathcal{V}$ and $\mathcal{Y} \subseteq \mathcal{V}$, it holds $f(\mathcal{X}|\mathcal{Y}) = f(\mathcal{X} \cup \mathcal{Y}) - f(\mathcal{Y})$. Second, eq. (9.51) implies ineq. (9.48), since $f(\mathcal{B}_2^\star|\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) = \eta f(\mathcal{A}_2)$, and $f(\mathcal{A}_1|\mathcal{A}_2) - f(\mathcal{B}_1^\star|\mathcal{A} \setminus \mathcal{B}_1^\star) \geq 0$. The latter is true due to the following two observations: $f(\mathcal{A}_1|\mathcal{A}_2) \geq f(\mathcal{B}_1^\star|\mathcal{A}_2)$, since f is monotone and $\mathcal{B}_1^\star \subseteq \mathcal{A}_1$; and $f(\mathcal{B}_1^\star|\mathcal{A}_2) \geq f(\mathcal{B}_1^\star|\mathcal{A} \setminus \mathcal{B}_1^\star)$, since f is submodular and $\mathcal{A}_2 \subseteq \mathcal{A} \setminus \mathcal{B}_1^\star$ (see also Fig. 13).

Proof of ineq. (9.49) Since it is $\mathcal{B}_2^\star \neq \emptyset$ (and as a result, it also is $\mathcal{A}_1^+ \neq \emptyset$), and since for all elements $a \in \mathcal{A}_1^+$ and all elements $b \in \mathcal{B}_2^\star$ it is $f(a) \geq f(b)$, from Lemma 29 we have:

$$\begin{aligned} f(\mathcal{B}_2^\star|\mathcal{A}_1^+) &\leq |\mathcal{B}_2^\star| f(\mathcal{A}_1^+) \\ &\leq \beta f(\mathcal{A}_1^+), \end{aligned} \tag{9.52}$$

since $|\mathcal{B}_2^\star| \leq \beta$. Overall,

$$f(\mathcal{A}_1^+) \geq \frac{1}{\beta} f(\mathcal{B}_2^\star|\mathcal{A}_1^+) \tag{9.53}$$

$$\geq \frac{1}{\beta} f(\mathcal{B}_2^\star|\mathcal{A}_1^+ \cup \mathcal{A}_2^+) \tag{9.54}$$

$$= \frac{1}{\beta} f(\mathcal{B}_2^\star|\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) \tag{9.55}$$

$$= \eta \frac{1}{\beta} f(\mathcal{A}_2), \tag{9.56}$$

where ineqs. (9.53)-(9.56) hold for the following reasons: ineq. (9.53) follows from ineq. (9.52); ineq. (9.54) holds since f is submodular and $\mathcal{A}_1^+ \subseteq \mathcal{A}_1^+ \cup \mathcal{A}_2^+$; eq. (9.55) holds due to the definitions of the sets $\mathcal{A}_1^+$, $\mathcal{A}_2^+$ and $\mathcal{B}^\star(\mathcal{A})$; finally, eq. (9.56) holds due to the definition of η.

Proof of ineq. (9.50) Let $b = 1/\beta$. We complete the proof first for the case where $(1-\eta) \geq \eta b$, and then for the case $(1-\eta) < \eta b$: when $(1-\eta) \geq \eta b$, $\max\{(1-\eta), \eta b\} = 1-\eta$ and $\eta \leq 1/(1+b)$; due to the latter, $1 - \eta \geq b/(1+b) = 1/(\beta+1)$ and, as a result, (9.50) holds. Finally, when $(1-\eta) < \eta b$, $\max\{(1-\eta), \eta b\} = \eta b$ and $\eta > 1/(1+b)$; due to the latter, $\eta b > b/(1+b)$ and, as a result, (9.50) holds.

We completed the proof of $0 \leq \eta \leq 1$, and of ineqs. (9.48)-(9.50). Thus, we also completed the proof of ineq. (9.44).

To prove ineq. (9.45), we consider the following mutually exclusive and collectively exhaustive cases:

- consider $\mathcal{B}_2^\star = \emptyset$, i.e., all elements in $\mathcal{A}_1$ are removed, and as result, none of the elements in $\mathcal{A}_2$ is removed. Then, $f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) = f(\mathcal{A}_2)$, and ineq. (9.45) holds.

- Consider $\mathcal{B}_2^\star \neq \emptyset$, i.e., at least one of the elements in $\mathcal{A}_1$ is *not* removed; call any of these elements s. Then:
$$f(\mathcal{A} \setminus \mathcal{B}^\star(\mathcal{A})) \geq f(s), \tag{9.57}$$
since f is non-decreasing. In addition:
$$f(\mathcal{A}_2) \leq \sum_{v \in \mathcal{A}_2} f(v) \leq (\alpha - \beta) f(s), \tag{9.58}$$
where the first inequality holds since f is submodular [70, Proposition 2.1], and the second holds due to Lemma 34 and the fact that $\mathcal{A}_2$ is constructed by Algorithm 19 such that $\mathcal{A}_1 \cup \mathcal{A}_2 \subseteq \mathcal{V}$ and $\mathcal{A}_1 \cup \mathcal{A}_2 \in \mathcal{I}$, where $|\mathcal{A}_1| = \beta$ (since $\mathcal{A}_1$ is constructed by Algorithm 19 such that $\mathcal{A}_1 \subseteq \mathcal{V}$ and $\mathcal{A}_1 \in \mathcal{I}'$, where $(\mathcal{V}, \mathcal{I}')$ is a matroid with rank β) and $(\mathcal{V}, \mathcal{I})$ is a matroid that has rank α; the combination of ineq. (9.57) and ineq. (9.58) implies ineq. (9.45).

Overall, the proof of ineq. (9.45) is complete. ■

Proof of Theorem 22's part 2 (running time)

We complete the proof in two steps, where we denote the time for each evaluation of the objective function f as τ_f. In particular, we first compute the running time of lines 2-8 and, then, of lines 9-15: lines 2-8 need at most $|\mathcal{V}|[|\mathcal{V}|\tau_f + |\mathcal{V}|\log(|\mathcal{V}|) + |\mathcal{V}| + O(\log(|\mathcal{V}|))]$ time, since they are repeated at most $|\mathcal{V}|$ times, and at each repetition line 3 asks for at most $|\mathcal{V}|$ evaluations of f, and for their sorting, which takes $|\mathcal{V}|\log(|\mathcal{V}|) + |\mathcal{V}| + O(\log(|\mathcal{V}|))$ time, using, e.g., the merge sort algorithm. Similarly, lines 9-15 need $|\mathcal{V}|[|\mathcal{V}|\tau_f + |\mathcal{V}|\log(|\mathcal{V}|) + |\mathcal{V}| + O(\log(|\mathcal{V}|))]$. Overall, Algorithm 19 runs in $2|\mathcal{V}|[|\mathcal{V}|\tau_f + |\mathcal{V}|\log(|\mathcal{V}|) + |\mathcal{V}| + O(\log(|\mathcal{V}|))] = O(|\mathcal{V}|^2\tau_f)$ time. ■

CHAPTER 10 : Resilient (Non-)Submodular Sequential Maximization

Applications in machine learning, optimization, and control require the sequential selection of a few system elements, such as sensors, data, or actuators, to optimize the system performance across multiple time steps. However, in failure-prone and adversarial environments, sensors get attacked, data get deleted, and actuators fail. Thence, traditional sequential design paradigms become insufficient and, in contrast, *resilient* sequential designs that adapt against system-wide attacks, deletions, or failures become important. In general, resilient sequential design problems are computationally hard. Also, even though they often involve objective functions that are monotone and (possibly) submodular, no scalable approximation algorithms are known for their solution. In this chapter, we provide the first scalable algorithm, that achieves the following characteristics: *system-wide resiliency*, i.e., the algorithm is valid for any number of denial-of-service attacks, deletions, or failures; *adaptiveness*, i.e., at each time step, the algorithm selects system elements based on the history of inflicted attacks, deletions, or failures; and *provable approximation performance*, i.e., the algorithm guarantees for monotone objective functions a solution close to the optimal. We quantify the algorithm's approximation performance using a notion of curvature for monotone (not necessarily submodular) set functions. Finally, we support our theoretical analyses with simulated experiments, by considering a control-aware sensor scheduling scenario, namely, *sensing-constrained robot navigation.*[1]

10.1. Introduction

Problems in machine learning, optimization, and control [6, 50, 54, 193, 221, 222, 225, 226, 232] require the design of systems in applications such as:

- *Car-congestion prediction*: Given a flood of driving data, collected from the drivers' smart-phones, which *few* drivers' data should we process at *each time of the day* to enable the accurate prediction of car traffic? [232]

- *Adversarial-target tracking*: At a flying robot, that uses on-board sensors to navigate itself, which *few* sensors should we activate *at each time step* both to maximize the robot's battery life, and to ensure its ability to track targets moving in a cluttered environment? [193]

- *Hazardous environmental-monitoring*: In a team of mobile robots, which *few* robots should we choose *at each time step* as actuators (leaders) to guarantee the team's capability to monitor the radiation around a nuclear reactor despite intro-robot communication noise? [50]

In particular, all the aforementioned applications [6, 50, 54, 193, 221, 222, 225, 226, 232] motivate the *sequential* selection of *a few* system elements, such as sensors, data, or actuators, to optimize the system performance across multiple time steps, subject to a resource constraint, such as limited battery for sensor activation. More formally, each of the above

[1] This chapter is based on the paper by Tzoumas et al. [231].

applications motivate the solution to a sequential optimization problem of the form:

$$\begin{aligned} &\max_{\mathcal{A}_1 \subseteq \mathcal{V}_1} \cdots \max_{\mathcal{A}_T \subseteq \mathcal{V}_T} f(\mathcal{A}_1, \ldots, \mathcal{A}_T), \\ &\text{such that:} \\ &|\mathcal{A}_t| = \alpha_t, \text{ for all } t = 1, \ldots, T, \end{aligned} \tag{10.1}$$

where T represents the number of design steps in time; the objective function f is monotone and (possibly) submodular —submodularity is a diminishing returns property;— and the cardinality bound α_t captures a resource constraint at time t. The problem in eq. (10.1) is combinatorial, and, specifically, it is NP-hard [13]; notwithstanding, several approximation algorithms have been proposed for its solution, such as the greedy [12].

But in all the above critical applications, sensors can get cyber-attacked [24]; data can get deleted [36]; and actuators can fail [23]. Hence, in such failure-prone and adversarial scenarios, *resilient* sequential designs that *adapt* against denial-of-service attacks, deletions, or failures become important.

In this chapter, we formalize for the first time a problem of *resilient monotone sequential maximization*, that goes beyond the traditional objective of the problem in eq. (10.1), and guards adaptively against real-time attacks, deletions, or failures. In particular, we introduce the following resilient re-formulation of the problem in eq. (10.1):

$$\begin{aligned} &\max_{\mathcal{A}_1 \subseteq \mathcal{V}_1} \min_{\mathcal{B}_1 \subseteq \mathcal{A}_1} \cdots \max_{\mathcal{A}_T \subseteq \mathcal{V}_T} \min_{\mathcal{B}_T \subseteq \mathcal{A}_T} f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_T \setminus \mathcal{B}_T), \\ &\text{such that:} \\ &|\mathcal{A}_t| = \alpha_t \text{ and } |\mathcal{B}_t| \leq \beta_t, \text{ for all } t = 1, \ldots, T, \end{aligned} \tag{10.2}$$

where the number β_t represents the number of possible attacks, deletions, or failures —in general, it is $\beta_t \leq \alpha_t$. Overall, the problem in eq. (10.2) maximizes the function f despite real-time *worst-case* failures that compromise the consecutive maximization steps in eq. (10.1). Therefore, the problem formulation in eq. (10.2) is suitable in scenarios where there is no prior on the removal mechanism, as well as, in scenarios where protection against worst-case failures is essential, such as in expensive experiment designs, or missions of adversarial-target tracking.

In more detail, the problem in eq. (10.2) may be interpreted as a T-stage perfect information sequential game between two players [26, Chapter 4], namely, a "maximization" player (designer), and a "minimization" player (attacker), who play sequentially, *both observing all past actions of all players*, and with the designer starting the game. That is, at each time $t = 1, \ldots, T$, both the designer and the attacker *adapt* their set selections to the history of all the players' selections so far, and, in particular, the attacker adapts its selection also to the current (t-th) selection of the designer (since at each step t, the attacker plays after it observes the selection of the 'designer).

In sum, the problem in eq. (10.2) goes beyond traditional (non-resilient) optimization [31, 32, 33, 39, 40] by proposing *resilient* optimization; beyond single-step resilient optimization [34, 35, 56] by proposing *multi-step* (sequential) resilient optimization; beyond memo-

ryless resilient optimization [41] by proposing *adaptive* resilient optimization; and beyond protection against non-adversarial attacks [36, 37] by proposing protection against *worst-case* attacks. Hence, the problem in eq. (10.2) aims to protect the system performance over extended periods of time against real-time denial-of-service attacks or failures, which is vital in critical applications, such as multi-target surveillance with teams of mobile robots [6].

Contributions. In this chapter, we make the contributions:

- (*Problem definition*) We formalize the problem of *resilient monotone sequential maximization* against denial-of-service removals, per eq. (10.2). This is the first work to formalize, address, and motivate this problem.

- (*Solution*) We develop the first algorithm for the problem of resilient monotone sequential maximization in eq. (10.2), and prove it has the following properties:

  - *system-wide resiliency*: the algorithm is valid for any number of removals;

  - *adaptiveness*: the algorithm adapts the solution to each of the maximization steps in eq. (10.2) to the history of realized (inflicted) removals;

  - *minimal running time*: the algorithm terminates with the same running time as state-of-the-art algorithms for (non-resilient) set function optimization, per eq. (10.1);

  - *provable approximation performance*: the algorithm ensures for all $T \geq 1$, and for objective functions f that are monotone and (possibly) submodular —as it holds true in all aforementioned applications [6, 50, 54, 193, 221, 222, 225, 226, 232],— a solution finitely close to the optimal.

    To quantify the algorithm's approximation performance, we used a notion of curvature for monotone (not necessarily submodular) set functions.

- (*Simulations*) We conduct simulations in a variety of sensor scheduling scenarios for autonomous robot navigation, varying the number of sensor failures. Our simulations validate the benefits of our approach.

Overall, the proposed algorithm in this chapter enables the resilient reformulation and solution of all the aforementioned applications [6, 50, 54, 193, 221, 222, 225, 226, 232] against worst-case attacks, deletions, or failures, over multiple design steps, and with provable approximation guarantees.

Notation. Calligraphic fonts denote sets (e.g., $\mathcal{A}$). Given a set $\mathcal{A}$, then $2^{\mathcal{A}}$ denotes the power set of $\mathcal{A}$; in addition, $|\mathcal{A}|$ denotes $\mathcal{A}$'s cardinality; given also a set $\mathcal{B}$, then $\mathcal{A} \setminus \mathcal{B}$ denotes the set of elements in $\mathcal{A}$ that are not in $\mathcal{B}$. Given a ground set $\mathcal{V}$, a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$, and an element $x \in \mathcal{V}$, the $f(x)$ denotes $f(\{x\})$, and the $f(\mathcal{A}, \mathcal{B})$ denotes $f(\mathcal{A} \cup \mathcal{B})$.

10.2. Resilient Monotone Sequential Maximization

We formally define resilient monotone sequential maximization. We start with the basic definition of monotonicity.

Definition 37 (Monotonicity)**.** *Consider any finite ground set $\mathcal{V}$. The set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is non-decreasing if and only if for any sets $\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{V}$, it holds $f(\mathcal{A}) \leq f(\mathcal{A}')$.*

We define next the main problem in this chapter.

Problem 5. (Resilient Monotone Sequential Maximization) *Consider the parameters: an integer T; finite ground sets $\mathcal{V}_1, \ldots, \mathcal{V}_T$; a non-decreasing set function $f : 2^{\mathcal{V}_1} \times \cdots \times 2^{\mathcal{V}_T} \mapsto \mathbb{R}$ such that, without loss of generality, it holds $f(\emptyset) = 0$ and f is non-negative; finally, integers α_t and β_t such that $0 \leq \beta_t \leq \alpha_t \leq |\mathcal{V}_t|$, for all $t = 1, 2, \ldots, T$.*

The problem of resilient monotone sequential maximization *is to maximize the objective function f through a sequence of T maximization steps, despite compromises to the solutions of each of the maximization steps; in particular, at each maximization step $t = 1, \ldots, T$ a set $\mathcal{A}_t \subseteq \mathcal{V}_t$ of cardinality α_t is selected, and is compromised by a worst-case set removal $\mathcal{B}_t$ of cardinality β_t. Formally:*

$$\begin{aligned} \max_{\mathcal{A}_1 \subseteq \mathcal{V}_1} \min_{\mathcal{B}_1 \subseteq \mathcal{A}_1} \cdots \max_{\mathcal{A}_T \subseteq \mathcal{V}_T} \min_{\mathcal{B}_T \subseteq \mathcal{A}_T} & f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_T \setminus \mathcal{B}_T), \\ & \textit{such that:} \\ & |\mathcal{A}_t| = \alpha_t \textit{ and } |\mathcal{B}_t| \leq \beta_t, \textit{ for all } t = 1, \ldots, T. \end{aligned} \tag{10.3}$$

As we mentioned in this chapter's Introduction, Problem 20 may be interpreted as a T-stage perfect information sequential game between two players [26, Chapter 4], a "maximization" player, and a "minimization" player, who play sequentially, both observing all past actions of all players, and with the "maximization" player starting the game. In the following paragraphs, we describe this game in more detail:

- *1st round of the game in Problem 5*: the "maximization" player selects the set $\mathcal{A}_1$; then, the "minimization" player observes $\mathcal{A}_1$, and selects the set $\mathcal{B}_1$ against $\mathcal{A}_1$;
- *2nd round of the game in Problem 5*: the "maximization" player, who already knows $\mathcal{A}_1$, observes $\mathcal{B}_1$, and selects the set $\mathcal{A}_2$, given $\mathcal{A}_1$ and $\mathcal{B}_1$; then, the "minimization" player, who already knows $\mathcal{A}_1$ and $\mathcal{B}_1$, observes $\mathcal{A}_2$, and selects the set $\mathcal{B}_2$ against $\mathcal{A}_2$, given $\mathcal{A}_1$ and $\mathcal{B}_1$.

$$\vdots$$

- *T-th round of the game in Problem 5*: the "maximization" player, who already knows the history of selections $\mathcal{A}_1, \ldots, \mathcal{A}_{T-1}$, as well as, removals $\mathcal{B}_1, \ldots, \mathcal{B}_{T-1}$, selects the set $\mathcal{A}_T$, given $\mathcal{A}_1, \ldots, \mathcal{A}_{T-1}$ and $\mathcal{B}_1, \ldots, \mathcal{B}_{T-1}$; then, the "minimization" player, who also already knows the history of selections $\mathcal{A}_1, \ldots, \mathcal{A}_{T-1}$, as well as, removals $\mathcal{B}_1, \ldots, \mathcal{B}_{T-1}$, observes $\mathcal{A}_T$, and selects the set $\mathcal{B}_T$ against $\mathcal{A}_T$, given $\mathcal{A}_1, \ldots, \mathcal{A}_{T-1}$ and $\mathcal{B}_1, \ldots, \mathcal{B}_{T-1}$.

10.3. Adaptive Algorithm for Problem 5

We present the first algorithm for Problem 5, show it is adaptive, and, finally, describe the intuition behind it. The pseudo-code of the algorithm is described in Algorithm 20.

10.3.1. Intuition behind Algorithm 20

The goal of Problem 5 is to ensure a maximal value for an objective function f through a sequence of T maximization steps, despite compromises to the solutions of each of the maximization steps. In particular, at each maximization step $t = 1, \ldots, T$, Problem 5 aims to select a set $\mathcal{A}_t$ towards a maximal value of f, despite that each $\mathcal{A}_t$ is compromised by a worst-case set removal $\mathcal{B}_t$ from $\mathcal{A}_t$, resulting to f being finally evaluated at the sequence of sets $\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_T \setminus \mathcal{B}_T$ instead of the sequence of sets $\mathcal{A}_1, \ldots, \mathcal{A}_T$. In this context, Algorithm 20 aims to fulfil the goal of Problem 5 by constructing each set $\mathcal{A}_t$ as the union of the sets $\mathcal{S}_{t,1}$, and $\mathcal{S}_{t,2}$ (line 9 of Algorithm 20), whose role we describe in more detail below:

Set $\mathcal{S}_{t,1}$ approximates worst-case set removal from $\mathcal{A}_t$: Algorithm 20 aims with the set $\mathcal{S}_{t,1}$ to capture the worst-case removal of β_t elements among the α_t elements that Algorithm 20 is going to select in $\mathcal{A}_t$; equivalently, the set $\mathcal{S}_{t,1}$ is aimed to act as a "bait" to an attacker that selects to remove the *best* β_t elements from $\mathcal{A}_t$ (*best* with respect to the elements' contribution towards the goal of Problem 5). However, the problem of selecting the *best* β_t elements in $\mathcal{V}_t$ is a combinatorial and, in general, intractable problem [13]. For this reason, Algorithm 20 aims to *approximate* the best β_t elements in $\mathcal{V}_t$, by letting $\mathcal{S}_{t,1}$ be the set of β_t elements with the largest marginal contributions in the value of the objective function f (lines 3-4 of Algorithm 20).

Set $\mathcal{S}_{t,2}$ is such that $\mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$ approximates optimal set solution to t-th maximization step of Problem 5: Assuming that $\mathcal{S}_{t,1}$ is the set of β_t elements that are going to be removed from Algorithm 20's set selection $\mathcal{A}_t$, Algorithm 20 needs next to select a set $\mathcal{S}_{t,2}$ of $\alpha_t - \beta_t$ elements to complete the construction of $\mathcal{A}_t$, since it is $|\mathcal{A}_t| = \alpha_t$ per Problem 5. In particular, for $\mathcal{A}_t = \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$ to be an optimal solution to t-th maximization step of Problem 5 (assuming the removal of $\mathcal{S}_{t,1}$ from $\mathcal{A}_t$), Algorithm 20 needs to select $\mathcal{S}_{t,2}$ as a *best* set of $\alpha_t - \beta_t$ elements from $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$. Nevertheless, the problem of selecting a *best* set of $\alpha_t - \beta_t$ elements from $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$ is a combinatorial and, in general, intractable problem [13]. As a result, Algorithm 20 aims to *approximate* such a best set, using the greedy procedure in the lines 5-8 of Algorithm 20.

Overall, Algorithm 20 constructs the sets $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$ to approximate an optimal solution $\mathcal{A}_t$ to the t-th maximization step of Problem 5 with their union (line 9 of Algorithm 20).

We describe next the steps in Algorithm 20 in more detail.

10.3.2. Description of steps in Algorithm 20

Algorithm 20 executes four steps for each $t = 1, \ldots, T$, where T is the number of maximization steps in Problem 5:

Initialization (line 2 of Algorithm 20): Algorithm 20 defines two auxiliary sets, namely, the $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$, and initializes each of them with the empty set (line 2 of Algorithm 20).

Algorithm 20 Adaptive algorithm for Problem 5.

Input: Per Problem 5, Algorithm 20 receives two input types:

- (*Off-line*) Integer T; finite ground sets $\mathcal{V}_1, \ldots, \mathcal{V}_T$; set function $f : 2^{\mathcal{V}_1} \times \cdots \times 2^{\mathcal{V}_T} \mapsto \mathbb{R}$ such that f is non-decreasing, non-negative, and $f(\emptyset) = 0$; integers α_t and β_t such that $0 \leq \beta_t \leq \alpha_t \leq |\mathcal{V}_t|$, for all $t = 1, \ldots, T$.
- (*On-line*) At each step $t = 2, 3, \ldots, T$: realized set removal $\mathcal{B}_{t-1}$ from Algorithm 20's set selection $\mathcal{A}_{t-1}$.

Output: At each step $t = 1, 2, \ldots, T$, set $\mathcal{A}_t$.

1: **for all** $t = 1, \ldots, T$ **do**
2: $\quad \mathcal{S}_{t,1} \leftarrow \emptyset$; $\quad \mathcal{S}_{t,2} \leftarrow \emptyset$;
3: $\quad$ Sort elements in $\mathcal{V}_t$ such that $\mathcal{V}_t \equiv \{v_{t,1}, \ldots, v_{t,|\mathcal{V}_t|}\}$ and $f(v_{t,1}) \geq \ldots \geq f(v_{t,|\mathcal{V}_t|})$;
4: $\quad \mathcal{S}_{t,1} \leftarrow \{v_{t,1}, \ldots, v_{t,\beta}\}$;
5: $\quad$ **while** $|\mathcal{S}_{t,2}| < \alpha_t - \beta_t$ **do**
6: $\quad\quad x \in \arg\max_{y \in \mathcal{V}_t \setminus (\mathcal{S}_{t,1} \cup \mathcal{S}_{t,2})} f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus \mathcal{B}_{t-1}, \mathcal{S}_{t,2} \cup \{y\})$;
7: $\quad\quad \mathcal{S}_{t,2} \leftarrow \{x\} \cup \mathcal{S}_{t,2}$;
8: $\quad$ **end while**
9: $\quad \mathcal{A}_t \leftarrow \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$;
10: **end for**

The purpose of $\mathcal{S}_{t,1}$ and of $\mathcal{S}_{t,2}$ is to construct the set $\mathcal{A}_t$, which is the set Algorithm 20 selects as a solution to Problem 5's t-th maximization step; in particular, the union of $\mathcal{S}_{t,1}$ and of $\mathcal{S}_{t,2}$ constructs $\mathcal{A}_t$ at the end of the t-th execution of the algorithm's "for loop" (lines 1-10 of Algorithm 20).

Construction of set $\mathcal{S}_{t,1}$ (lines 3-4 of Algorithm 20): Algorithm 20 constructs the set $\mathcal{S}_{t,1}$ such that $\mathcal{S}_{t,1}$ contains β_t elements from the ground set $\mathcal{V}_t$ and, for any element $s \in \mathcal{S}_{t,1}$ and any element $s' \notin \mathcal{S}_{t,1}$, the marginal value of $f(s)$ is at least that of $f(s')$; that is, among all elements in $\mathcal{V}_t$, the set $\mathcal{S}_{t,1}$ contains a collection of β_t elements that correspond to the highest marginal values of f. In detail, Algorithm 20 constructs $\mathcal{S}_{t,1}$ by first sorting and indexing all elements in $\mathcal{V}_t$ such that $\mathcal{V}_t = \{v_{t,1}, \ldots, v_{t,|\mathcal{V}_t|}\}$ and $f(v_{t,1}) \geq \ldots \geq f(v_{t,|\mathcal{V}_t|})$ (line 3 of Algorithm 20), and, then, by including in $\mathcal{S}_{t,1}$ the fist β_t elements among the $\{v_{t,1}, \ldots, v_{t,|\mathcal{V}_t|}\}$ (line 4 of Algorithm 20).

Construction of set $\mathcal{S}_{t,2}$ (lines 5-8 of Algorithm 20): Algorithm 20 constructs the set $\mathcal{S}_{t,2}$ by picking greedily $\alpha_t - \beta_t$ elements from the set $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$, and by accounting for the effect that the history of set selections and removals $(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus \mathcal{B}_{t-1})$ has on the objective function f of Problem 5. Specifically, the greedy procedure in Algorithm 20's "while loop" (lines 5-8 of Algorithm 20) selects an element $y \in \mathcal{V}_t \setminus (\mathcal{S}_{t,1} \cup \mathcal{S}_{t,2})$ to add in $\mathcal{S}_{t,2}$ only if y maximizes the value of $f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus \mathcal{B}_{t-1}, \mathcal{S}_{t,2} \cup \{y\})$.

Construction of set $\mathcal{A}_t$ (line 9 of Algorithm 20): Algorithm 20 proposes the set $\mathcal{A}_t$ as a solution to Problem 5's t-th maximization step. To this end, Algorithm 20 constructs $\mathcal{A}_t$ as the union of the previously constructed sets $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$.

In sum, Algorithm 20 enables an adaptive solution of Problem 5: for each $t = 1, 2, \ldots,$

Algorithm 20 constructs a solution set $\mathcal{A}_t$ to the t-th maximization step of Problem 5 based on both the history of selected solutions up to step $t-1$, namely, the sets $\mathcal{A}_1, \ldots, \mathcal{A}_{t-1}$, and the corresponding history of set removals from $\mathcal{A}_1, \ldots, \mathcal{A}_{t-1}$, namely, the $\mathcal{B}_1, \ldots, \mathcal{B}_{t-1}$.

10.4. Performance Guarantees for Algorithm 20

We quantify Algorithm 20's performance, by bounding its running time, and its approximation performance. To this end, we use the following two notions of curvature for set functions.

10.4.1. Curvature and total curvature of non-decreasing functions

We present the notions of *curvature* and of *total curvature* for non-decreasing set functions. We start by describing the notions of *modularity* and *submodularity* for set functions.

Definition 38 (Modularity)**.** *Consider any finite set $\mathcal{V}$. The set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is modular if and only if for any set $\mathcal{A} \subseteq \mathcal{V}$, it holds $g(\mathcal{A}) = \sum_{v \in \mathcal{A}} g(v)$.*

In words, a set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is modular if through g all elements in $\mathcal{V}$ cannot substitute each other; in particular, Definition 38 of modularity implies that for any set $\mathcal{A} \subseteq \mathcal{V}$, and for any element $v \in \mathcal{V} \setminus \mathcal{A}$, it holds $g(\{v\} \cup \mathcal{A}) - g(\mathcal{A}) = g(v)$.

Definition 39 (Submodularity [70, Proposition 2.1])**.** *Consider any finite set $\mathcal{V}$. The set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* submodular *if and only if for any sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, and any element $v \in \mathcal{V}$, it holds $g(\mathcal{A} \cup \{v\}) - g(\mathcal{A}) \geq g(\mathcal{B} \cup \{v\}) - g(\mathcal{B})$.*

Definition 39 implies that a set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is submodular if and only if it satisfies a diminishing returns property where for any set $\mathcal{A} \subseteq \mathcal{V}$, and for any element $v \in \mathcal{V}$, the marginal gain $g(\mathcal{A} \cup \{v\}) - g(\mathcal{A})$ is non-increasing. In contrast to modularity, submodularity implies that the elements in $\mathcal{V}$ *can* substitute each other, since Definition 39 of submodularity implies the inequality $g(\{v\} \cup \mathcal{A}) - g(\mathcal{A}) \leq g(v)$; that is, in the presence of the set $\mathcal{A}$, the element v may lose part of its contribution to the value of $g(\{x\} \cup \mathcal{A})$.

Definition 40. (Curvature of monotone submodular functions [33]) *Consider a finite set $\mathcal{V}$, and a non-decreasing submodular set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that (without loss of generality) for any element $v \in \mathcal{V}$, it is $g(v) \neq 0$. The curvature of g is defined as follows:*

$$\kappa_g \triangleq 1 - \min_{v \in \mathcal{V}} \frac{g(\mathcal{V}) - g(\mathcal{V} \setminus \{v\})}{g(v)}. \tag{10.4}$$

Definition 40 of curvature implies that for any non-decreasing submodular set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$, it holds $0 \leq \kappa_g \leq 1$. In particular, the value of κ_g measures how far g is from modularity, as we explain next: if $\kappa_g = 0$, then for all elements $v \in \mathcal{V}$, it holds $g(\mathcal{V}) - g(\mathcal{V} \setminus \{v\}) = g(v)$, that is, g is modular. In contrast, if $\kappa_g = 1$, then there exist an element $v \in \mathcal{V}$ such that $g(\mathcal{V}) = g(\mathcal{V} \setminus \{v\})$, that is, in the presence of $\mathcal{V} \setminus \{v\}$, v loses all its contribution to the value of $g(\mathcal{V})$.

Definition 41. (Total curvature of non-decreasing functions [15, Section 8]) *Consider a finite set $\mathcal{V}$, and a monotone set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$. The total curvature of g is*

defined as follows:

$$c_g \triangleq 1 - \min_{v \in \mathcal{V}} \min_{\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}} \frac{g(\{v\} \cup \mathcal{A}) - g(\mathcal{A})}{g(\{v\} \cup \mathcal{B}) - g(\mathcal{B})}. \tag{10.5}$$

Definition 41 of total curvature implies that for any non-decreasing set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$, it holds $0 \leq c_g \leq 1$. To connect the notion of total curvature with that of curvature, we note that when the function g is non-decreasing and submodular, then the two notions coincide, i.e., it holds $c_g = \kappa_g$; the reason is that if g is non-decreasing and submodular, then the inner minimum in eq. (10.5) is attained for $\mathcal{A} = \mathcal{B} \setminus \{v\}$ and $\mathcal{B} = \emptyset$. In addition, to connect the above notion of total curvature with the notion of modularity, we note that if $c_g = 0$, then g is modular, since eq. (10.5) implies that for any elements $v \in \mathcal{V}$, and for any sets $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}$, it holds:

$$(1 - c_g) \left[g(\{v\} \cup \mathcal{B}) - g(\mathcal{B}) \right] \leq g(\{v\} \cup \mathcal{A}) - g(\mathcal{A}), \tag{10.6}$$

which for $c_g = 0$ implies the modularity of g. Finally, to connect the above notion of total curvature with the notion of monotonicity, we mention that if $c_g = 1$, then eq. (10.6) implies that g is merely non-decreasing (as it is already assumed by the Definition 41 of total curvature).

Definition 42 (Approximate submodularity)**.** *Consider a finite set $\mathcal{V}$, and a non-decreasing set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$, whose total curvature c_g is such that $c_g < 1$. Then, we say that g is* approximately submodular*.*

10.4.2. Performance analysis for Algorithm 20

We quantify Algorithm 20's approximation performance, as well as, its running time per maximization step in Problem 5.

Theorem 23 (Performance of Algorithm 20)**.** *Consider an instance of Problem 5, the notation therein, the notation in Algorithm 20, and the definitions:*

- *let the number $f^\star$ be the (optimal) value to Problem 5;*
- *given sets $\mathcal{A}_{1:T} \triangleq (\mathcal{A}_1, \ldots, \mathcal{A}_T)$ as solutions to the maximization steps in Problem 5, let $\mathcal{B}^\star(\mathcal{A}_{1:T})$ be the collection of optimal (worst-case) set removals from each of the $\mathcal{A}_t$, where $t = 1, \ldots, T$, per Problem 5, i.e.:*

$$\mathcal{B}^\star(\mathcal{A}_{1:T}) \in \arg \min_{\mathcal{B}_1 \subseteq \mathcal{A}_1, |\mathcal{B}_1| \leq \beta_1} \cdots \min_{\mathcal{B}_T \subseteq \mathcal{A}_T, |\mathcal{B}_T| \leq \beta_T} f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_T \setminus \mathcal{B}_T);$$

The performance of Algorithm 20 is bounded as follows:

*leftmirgin=** (Approximation performance) *Algorithm 20 returns the sequence of sets $\mathcal{A}_{1:T} \triangleq (\mathcal{A}_1, \ldots, \mathcal{A}_T)$ such that, for all $t = 1, \ldots, T$, it holds $\mathcal{A}_t \subseteq \mathcal{V}_t$, $|\mathcal{A}_t| \leq \alpha_t$, and:*

- *if the objective function f is non-decreasing and submodular, then:*

$$\frac{f(\mathcal{A}_{1:T} \setminus \mathcal{B}^\star(\mathcal{A}_{1:T}))}{f^\star} \geq (1-\kappa_f)^4, \tag{10.7}$$

  where κ_f is the curvature of f (Definition 40).

- *if the objective function f is non-decreasing, then:*

$$\frac{f(\mathcal{A}_{1:T} \setminus \mathcal{B}^\star(\mathcal{A}_{1:T}))}{f^\star} \geq (1-c_f)^5, \tag{10.8}$$

  where c_f is the total curvature of f (Definition 41).

*leftmiirgiin=** (Running time) *Algorithm 20 constructs each set $\mathcal{A}_t$, for each $t = 1, \ldots, T$, to solve the t-th maximization step of Problem 5, with $O(|\mathcal{V}_t|(\alpha_t - \beta_t))$ evaluations of f.*

Provable approximation performance. Theorem 23 implies on the approximation performance of Algorithm 20:

Near-optimality: Both for monotone submodular objective functions f with curvature $\kappa_f < 1$, and for merely monotone objective functions f with total curvature $c_f < 1$, Algorithm 20 guarantees a value for Problem 5 finitely close to the optimal. In particular, per ineq. (10.7) (case of submodular objective functions), the approximation factor of Algorithm 20 is bounded by $(1-\kappa_f)^4$, which is non-zero for any monotone submodular function f with $\kappa_f < 1$; similarly, per ineq. (10.8) (case of approximately-submodular functions), the approximation factor of Algorithm 20 is bounded by $(1-c_f)^5$, which is non-zero for any monotone function f with $c_f < 1$ —notably, although it is known for the problem of (non-resilient) set function maximization that the approximation bound $(1-c_f)$ is tight [15, Theorem 8.6], the tightness of the bound $(1-c_f)^5$ in ineq. (10.8) for Problem 5 is an open problem.

We discuss classes of functions f with curvatures $\kappa_f < 1$ or $c_f < 1$, along with relevant applications, in the remark below.

Remark 17. (Classes of functions f with $\kappa_f < 1$ or $c_f < 1$, and applications) Classes of functions f with $\kappa_f < 1$ *are the concave over modular functions [31, Section 2.1], and the* $\log\det$ *of positive-definite matrices [227, 228].* Classes of functions f with $c_f < 1$ *are support selection functions [223], and estimation error metrics such as the average minimum square error of the Kalman filter [193, Theorem 4]*

The aforementioned classes of functions f with $\kappa_f < 1$ or $c_f < 1$ appear in applications of facility location, machine learning, and control, such as sparse approximation and feature selection [225, 226], sparse recovery and column subset selection [221, 222], and actuator and sensor scheduling [54, 193]; as a result, Problem 5 enables applications such as resilient experiment design, resilient actuator scheduling for minimal control effort, and resilient multi-robot navigation with minimal sensing and communication.

Approximation performance for low curvature: For both monotone submodular and merely

monotone objective functions f, when the curvature κ_f and the total curvature c_f, respectively, tend to zero, Algorithm 20 becomes exact since for $\kappa_f \to 0$ and $c_f \to 0$ the terms $(1-\kappa_f)^4$ and $(1-c_f)^5$ in ineq. (10.7) and ineq. (10.8) tend to 1. Overall, Algorithm 20's curvature-dependent approximation bounds make a first step towards separating the classes of monotone submodular and merely monotone functions into functions for which Problem 5 can be approximated well (low curvature functions), and functions for which it cannot (high curvature functions).

A machine learning problem where Algorithm 20 guarantees an approximation performance close to 100% the optimal is that of Gaussian process regression for processes with RBF kernels [114, 230]; this problem emerges in applications of sensor deployment and scheduling for temperature monitoring. The reason that in this class of regression problems Algorithm 20 performs almost optimally is that the involved objective function is the entropy of the selected sensor measurements, which for Gaussian processes with RBF kernels has curvature value close to zero [228, Theorem 5].

Approximation performance for no failures or attacks: Both for monotone submodular objective functions f, and for merely monotone objective functions f, when the number of attacks, deletions, and failures is zero ($\beta_t = 0$, for all $t = 1, \ldots, T$), Algorithm 20's approximation performance is the same as that of the state-of-the-art algorithms for (non-resilient) set function maximization. In particular, when for all $t = 1, \ldots, T$ it is $\beta_t = 0$, then Algorithm 20 is the same as the local greedy algorithm, proposed in [12, Section 4] for (non-resilient) set function maximization, whose approximation performance is optimal [15, Theorem 8.6].

Minimal running time. Theorem 23 implies that Algorithm 20, even though it goes beyond the objective of (non-resilient) multi-step set function optimization, by accounting for attacks, deletions, and failures, it has the same order of running time as state-of-the-art algorithms for (non-resilient) multi-step set function optimization. In particular, such algorithms for (non-resilient) multi-step set function optimization [12, Section 4] [15, Section 8] terminate with $O(|\mathcal{V}_t|(\alpha_t - \beta_t))$ evaluations of the objective function f per maximization step $t = 1, \ldots, T$, and Algorithm 20 also terminates with $O(|\mathcal{V}_t|(\alpha_t - \beta_t))$ evaluations of the objective function f per maximization step $t = 1, \ldots, T$.

Summary of theoretical results. In sum, Algorithm 20 is the first algorithm for Problem 5, and it enjoys:

- *system-wide resiliency*: Algorithm 20 is valid for any number of denial-of-service attacks, deletions, and failures;
- *adaptiveness*: Algorithm 20 adapts the solution to each of the maximization steps in Problem 5 to the history of inflicted denial-of-service attacks and failures;
- *minimal running time*: Algorithm 20 terminates with the same running time as state-of-the-art algorithms for (non-resilient) multi-step submodular function optimization;
- *provable approximation performance*: Algorithm 20 ensures for all monotone objective functions f that are either submodular or approximately submodular ($c_f < 1$), and

for all $T \geq 1$, a solution finitely close to the optimal.

Notably, Algorithm 20 is also the first to guarantee for any number of failures, and for monotone approximately-submodular functions f, a provable approximation performance for the one-step version of Problem 5 where $T = 1$.

10.5. Numerical Experiments

In this section, we demonstrate a near-optimal performance of Algorithm 20 via numerical experiments. In particular, we consider a control-aware sensor scheduling scenario, namely, *sensing-constrained robot navigation*.[2] According to this scenario, an unmanned aerial vehicle (UAV), which has limited remaining battery and measurement-processing power, has the objective to land, and to this end, it schedules to activate at each time step only a subset of its on-board sensors, so to localize itself and enable the generation of a control input for landing; specifically, at each time step, the UAV generates its control input by implementing an LQG-optimal controller, given the measurements collected by the activated sensors up to the current time step [123, 193].

In more detail, in the following paragraphs we present a Monte Carlo analysis for an instance of the aforementioned sensing-constrained robot navigation scenario, in the presence of worst-case sensor failures, and observe that Algorithm 20 results to a near-optimal sensor selection schedule: in particular, the resulting navigation performance of the UAV matches the optimal in all tested instances for which the optimal selection could be computed via a brute-force approach.

Simulation setup. We adopt the same instance of the sensing-constrained robot navigation scenario adopted in [193, Section V.B]. Specifically, a UAV moves in a 3D space, starting from a randomly selected initial location. The objective of the UAV is to land at position [0, 0, 0] with zero velocity. The UAV is modelled as a double-integrator with state $x_t = [p_t \; v_t]^\top \in \mathbb{R}^6$ at each time $t = 1, 2, \ldots$ (p_t is the 3D position of the UAV, and v_t is its velocity), and can control its own acceleration $u_t \in \mathbb{R}^3$; the process noise is chosen as $W_t = \mathbf{I}_6$. The UAV is equipped with multiple sensors, as follows: it has two on-board GPS receivers, measuring the UAV position p_t with a covariance $2 \cdot \mathbf{I}_3$, and an altimeter, measuring only the last component of p_t (altitude) with standard deviation 0.5m. Moreover, the UAV can use a stereo camera to measure the relative position of 10 landmarks on the ground; we assume the location of each landmark to be known only approximately, and we associate to each landmark an uncertainty covariance, which is randomly generated at the beginning of each run. The UAV has limited on-board resource-constraints, hence it can only activate a subset of sensors (possibly different at each time step). For instance, the resource-constraints may be due to the power consumption of the GPS and the altimeter, or due to computational constraints that prevent to run object-detection algorithms to detect all landmarks on the ground.

Among the aforementioned 13 possible sensor measurements available to the UAV at each time step, we assume that the UAV can use only $\alpha = 11$ of them. In particular, the

[2]The scenario of sensing-constrained robot navigation is introduced in [193, Section V.B], yet in the absence of sensor failures.

UAV chooses the sensors to activate at each time step so to minimize an LQG cost with cost matrices Q (which penalizes the state vector) and R (which penalizes the control input vector), per the problem formulation in [193, Section II]; specifically, in this simulation setup we set $Q = \text{diag}\left([1e^{-3},\ 1e^{-3},\ 10,\ 1e^{-3},\ 1e^{-3},\ 10]\right)$ and $R = \mathbf{I}_3$. Note that the structure of Q (which penalizes the magnitude of the state vector) reflects the fact that during landing we are particularly interested in controlling the vertical direction and the vertical velocity (entries with larger weight in Q), while we are less interested in controlling accurately the horizontal position and velocity (assuming a sufficiently large landing site). Given a time horizon T for landing, in [193] it is proven that the UAV selects an optimal sensor schedule and generates an optimal LQG control input with cost matrices Q and R if it selects the sensors set $\mathcal{S}_t$ to activate at each time $t = 1, \ldots, T$ by minimizing an objective function of the form:

$$\sum_{t=1}^{T} \text{trace}[M_t \Sigma_{t|t}(\mathcal{S}_1, \ldots, \mathcal{S}_t)], \tag{10.9}$$

where M_t is a positive semi-definite matrix that depends on the LQG cost matrices Q and R, as well as, on the UAV's model dynamics; and $\Sigma_{t|t}(\mathcal{S}_1, \ldots, \mathcal{S}_t)$ is the error covariance of the Kalman filter given the sensor selections up to time t.

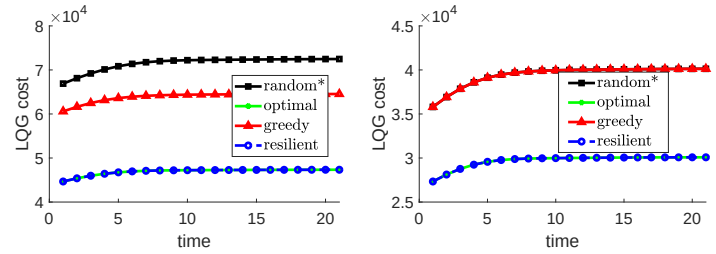
In the remaining paragraphs, we present results averaged over 10 Monte Carlo runs of the above simulation setup, where in each run we randomize the covariances describing the landmark position uncertainty, and where we vary the number β of sensors failures at each time step t: in particular, we consider β to vary among the values $1, 4, 7, 10$.

Compared algorithms. We compare four algorithms. All algorithms only differ in how they select the sensors used. The first algorithm is the optimal sensor selection algorithm, denoted as `optimal`, which attains the minimum of the cost function in eq. (10.9); this brute-force approach is viable since the number of available sensors is small. The second approach is a pseudo-random sensor selection, denoted as `random`*, which selects one of the GPS measurements and a random subset of the lidar measurements; note that we do not consider a fully random selection since in practice this often leads to an unobservable system. The third approach, denoted as `logdet`, selects sensors to greedily minimize the cost function in eq. (10.9), *ignoring the possibility of sensor failures*, per the problem formulation in eq. (9.1). The fourth approach uses Algorithm 20 to solve the resilient reformulation of eq. (10.9), per Problem 5, and is denoted as `s-LQG`.

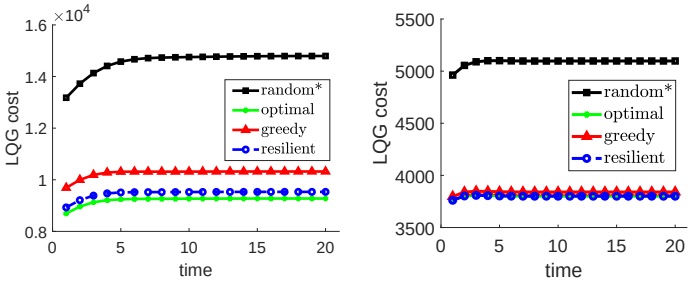
At each time step, from each of the selected sensor sets, selected by any of the above four algorithms, we consider an optimal sensor removal, which we compute via a brute-force.

Results. The results of our numerical analysis are reported in Fig. 14. In particular, Fig. 14 shows the LQG cost for increasing time, for the case where the number of selected sensors at each time step is $\alpha = 11$, while the number of sensor failures β at each time step varies across the values $10, 7, 4, 1$. The following observations are due:

- (*Near-optimality of Algorithm 20*) Algorithm 20 (blue colour in Fig. 14) performs close to the optimal brute-force algorithm (green colour in Fig. 14); in particular, across all scenarios in Fig. 14, Algorithm 20 achieves an approximation performance at least



(a) $\beta = 10$, $\alpha = 11$ (b) $\beta = 7$, $\alpha = 11$



(c) $\beta = 4$, $\alpha = 11$ (d) $\beta = 1$, $\alpha = 11$

Figure 14: LQG cost for increasing time, where across all sub-figures (a)-(d) it is $\alpha = 11$ (number of active sensors per time step). The value of β (number of sensor failures at each time step among the α active sensors) varies across the sub-figures.

97% the optimal.

- (*Performance of greedy algorithm*) The greedy algorithm (red colour in Fig. 14) performs poorly as the number β of sensor failures increases, which was expected, given that this algorithm greedily minimizes the cost function in eq. (10.9) ignoring the possibility of sensor failures.

- (*Performance of random algorithm*) Expectedly, also the performance of the random algorithm (black colour in Fig. 14) is poor across all scenarios in Fig. 14.

Overall, in the above numerical experiments, Algorithm 20 demonstrates a near-optimal approximation performance, and the necessity for the resilient reformulation of the problem in eq. (9.1) per Problem 5 is exemplified.

10.6. Concluding Remarks & Future Work

We made the first step to ensure the success of critical missions in machine learning and control, that involve the optimization of systems across multiple time-steps, against persistent failures or denial-of-service attacks. In particular, we provided the first algorithm for Problem 5, which, with minimal running time, adapts to the history of the inflicted failures and attacks, and guarantees a close-to-optimal performance against system-wide failures and attacks. To quantify the algorithm's approximation performance, we exploited a notion of curvature for monotone (not necessarily submodular) set functions, and contributed a first step towards characterizing the curvature's effect on the approximability of resilient *sequential* maximization. Our curvature-dependent characterizations complement the current knowledge on the curvature's effect on the approximability of simpler problems, such as of

non-sequential resilient maximization [35, 56], and of non-resilient maximization [31, 32, 33]. Finally, we supported our theoretical analyses with simulated experiments.

This chapter opens several avenues for future research, both in theory and in applications. Future work in theory includes the extension of our results to matroid constraints, to enable applications of set coverage and of network design [17, 233]. Future work in applications includes the experimental testing of the proposed algorithm in applications of motion-planning for multi-target covering with mobile vehicles [6], and in applications of control-aware sensor scheduling for multi-agent autonomous navigation [193], to enable resiliency in critical scenarios of surveillance, and of search and rescue.

10.7. Appendix: Proof of Results

10.7.1. Notation

In the appendix we use the following notation to support the proofs in this chapter: given a finite ground set $\mathcal{V}$, and a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$, then, for any sets $\mathcal{X} \subseteq \mathcal{V}$ and $\mathcal{X}' \subseteq \mathcal{V}$:

$$f(\mathcal{X}|\mathcal{X}') \triangleq f(\mathcal{X} \cup \mathcal{X}') - f(\mathcal{X}'). \tag{10.10}$$

Moreover, let the sets $\mathcal{A}^\star_{1:T} = (\mathcal{A}^\star_1, \dots, \mathcal{A}^\star_T)$ denote an (optimal) solution to Problem 5, i.e.:

$$\begin{aligned} &\mathcal{A}^\star_{1:T} \in \\ &\arg \max_{\mathcal{A}_1 \subseteq \mathcal{V}_1} \min_{\mathcal{B}_1 \subseteq \mathcal{A}_1} \cdots \max_{\mathcal{A}_T \subseteq \mathcal{V}_T} \min_{\mathcal{B}_T \subseteq \mathcal{A}_T} f(\mathcal{A}_1 \setminus \mathcal{B}_1, \dots, \mathcal{A}_T \setminus \mathcal{B}_T), \\ &\qquad \text{such that:} \\ &\qquad |\mathcal{A}_t| = \alpha_t \text{ and } |\mathcal{B}_t| \leq \beta_t, \text{ for all } t = 1, \dots, T. \end{aligned} \tag{10.11}$$

10.7.2. Preliminary lemmas

We list lemmas that support the proof of Theorem 23.

Lemma 38. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing submodular set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any $\mathcal{A} \subseteq \mathcal{V}$, it holds:*

$$f(\mathcal{A}) \geq (1 - \kappa_f) \sum_{a \in \mathcal{A}} f(a).$$

Proof of Lemma 38 Let $\mathcal{A} = \{a_1, a_2, \dots, a_{|\mathcal{A}|}\}$. We prove Lemma 38 by proving the following two inequalities:

$$f(\mathcal{A}) \geq \sum_{i=1}^{|\mathcal{A}|} f(a_i | \mathcal{V} \setminus \{a_i\}), \tag{10.12}$$

$$\sum_{i=1}^{|\mathcal{A}|} f(a_i | \mathcal{V} \setminus \{a_i\}) \geq (1 - \kappa_f) \sum_{i=1}^{|\mathcal{A}|} f(a_i). \tag{10.13}$$

We begin with the proof of ineq. (10.12):

$$
\begin{aligned}
f(\mathcal{A}) &= f(\mathcal{A}|\emptyset) && (10.14)\\
&\geq f(\mathcal{A}|\mathcal{V}\setminus\mathcal{A}) && (10.15)\\
&= \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V}\setminus\{a_i, a_{i+1}, \ldots, a_{|\mathcal{A}|}\}) && (10.16)\\
&\geq \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V}\setminus\{a_i\}), && (10.17)
\end{aligned}
$$

where ineqs. (10.15) to (10.17) hold for the following reasons: ineq. (10.15) is implied by eq. (10.14) because f is submodular and $\emptyset \subseteq \mathcal{V}\setminus\mathcal{A}$; eq. (10.16) holds since for any sets $\mathcal{X}\subseteq\mathcal{V}$ and $\mathcal{Y}\subseteq\mathcal{V}$ it is $f(\mathcal{X}|\mathcal{Y}) = f(\mathcal{X}\cup\mathcal{Y}) - f(\mathcal{Y})$, and it also $\{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ denotes the set $\mathcal{A}$; and ineq. (10.17) holds since f is submodular and $\mathcal{V}\setminus\{a_i, a_{i+1}, \ldots, a_\mu\} \subseteq \mathcal{V}\setminus\{a_i\}$. These observations complete the proof of ineq. (10.12).

We now prove ineq. (10.13) using the Definition 40 of κ_f, as follows: since $\kappa_f = 1 - \min_{v\in\mathcal{V}} \frac{f(v|\mathcal{V}\setminus\{v\})}{f(v)}$, it is implied that for all elements $v\in\mathcal{V}$ it is $f(v|\mathcal{V}\setminus\{v\}) \geq (1-\kappa_f)f(v)$. Therefore, adding the latter inequality across all elements $a\in\mathcal{A}$ completes the proof of ineq. (10.13). ■

Lemma 39. *Consider a finite ground set $\mathcal{V}$ and a monotone set function $f: 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any sets $\mathcal{A}\subseteq\mathcal{V}$ and $\mathcal{B}\subseteq\mathcal{V}$ such that $\mathcal{A}\cap\mathcal{B} = \emptyset$, it holds:*

$$f(\mathcal{A}\cup\mathcal{B}) \geq (1 - c_f)\left(f(\mathcal{A}) + f(\mathcal{B})\right).$$

Proof of Lemma 39 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A}\cup\mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i|\mathcal{A}\cup\{b_1, b_2, \ldots, b_{i-1}\}). \tag{10.18}$$

The definition of total curvature in Definition 41 implies:

$$
\begin{aligned}
&f(b_i|\mathcal{A}\cup\{b_1, b_2, \ldots, b_{i-1}\}) \geq\\
&(1-c_f) f(b_i|\{b_1, b_2, \ldots, b_{i-1}\}). && (10.19)
\end{aligned}
$$

The proof is completed by substituting ineq. (10.19) in eq. (10.18) and then by taking into account that it holds $f(\mathcal{A}) \geq (1-c_f)f(\mathcal{A})$, since $0\leq c_f\leq 1$. ■

Lemma 40. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f: 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any set $\mathcal{A}\subseteq\mathcal{V}$ and any set $\mathcal{B}\subseteq\mathcal{V}$ such that $\mathcal{A}\cap\mathcal{B} = \emptyset$, it holds:*

$$f(\mathcal{A}\cup\mathcal{B}) \geq (1 - c_f)\left(f(\mathcal{A}) + \sum_{b\in\mathcal{B}} f(b)\right).$$

Proof of Lemma 40 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i | \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}). \tag{10.20}$$

In addition, Definition 41 of total curvature implies:

$$\begin{aligned} f(b_i | \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}) &\geq (1 - c_f) f(b_i | \emptyset) \\ &= (1 - c_f) f(b_i), \end{aligned} \tag{10.21}$$

where the latter equation holds since $f(\emptyset) = 0$. The proof is completed by substituting (10.21) in (10.20) and then taking into account that $f(\mathcal{A}) \geq (1 - c_f) f(\mathcal{A})$ since $0 \leq c_f \leq 1$. ■

Lemma 41. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then for any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \setminus \mathcal{B} \neq \emptyset$, it holds:*

$$f(\mathcal{A}) + (1 - c_f) f(\mathcal{B}) \geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}) + f(\mathcal{A} \cap \mathcal{B}).$$

Proof of Lemma 41 Let $\mathcal{A} \setminus \mathcal{B} = \{i_1, i_2, \ldots, i_r\}$, where $r = |\mathcal{A} - \mathcal{B}|$. From Definition 41 of total curvature c_f, for any $i = 1, 2, \ldots, r$, it is $f(i_j | \mathcal{A} \cap \mathcal{B} \cup \{i_1, i_2, \ldots, i_{j-1}\}) \geq (1 - c_f) f(i_j | \mathcal{B} \cup \{i_1, i_2, \ldots, i_{j-1}\})$. Summing these r inequalities,

$$f(\mathcal{A}) - f(\mathcal{A} \cap \mathcal{B}) \geq (1 - c_f) \left(f(\mathcal{A} \cup \mathcal{B}) - f(\mathcal{B}) \right),$$

which implies the lemma. ■

Corollary 11. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. Then, for any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, it holds:*

$$f(\mathcal{A}) + \sum_{b \in \mathcal{B}} f(b) \geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}).$$

Proof of Corollary 11 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$.

$$\begin{aligned} f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i) &\geq (1 - c_f) f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i)) \qquad (10.22) \\ &\geq (1 - c_f) f(\mathcal{A} \cup \{b_1\}) + \sum_{i=2}^{|\mathcal{B}|} f(b_i) \\ &\geq (1 - c_f) f(\mathcal{A} \cup \{b_1, b_2\}) + \sum_{i=3}^{|\mathcal{B}|} f(b_i) \\ &\vdots \\ &\geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}), \end{aligned}$$

where ineq. (10.22) holds since $0 \leq c_f \leq 1$, and the rest due to Lemma 41 since $\mathcal{A} \cap \mathcal{B} = \emptyset$ implies $\mathcal{A} \setminus \{b_1\} \neq \emptyset$, $\mathcal{A} \cup \{b_1\} \setminus \{b_2\} \neq \emptyset$, ..., $\mathcal{A} \cup \{b_1, b_2, \ldots, b_{|\mathcal{B}|-1}\} \setminus \{b_{|\mathcal{B}|}\} \neq \emptyset$. ■

Lemma 42. *Recall the notation in Algorithm 20. Given the sets $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ selected by Algorithm 20 (lines 3-4 of Algorithm 20), then, for each $t = 1, \ldots, T$, let the set $\mathcal{O}_t$ be a subset —any subset— of $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$ of cardinality $\alpha_t - \beta_t$. Then, for the sets $\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}$ selected by Algorithm 20 (lines 5-8 of Algorithm 20), it holds:*

$$f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) \geq (1 - c_f)^2 f(\mathcal{O}_1, \ldots, \mathcal{O}_T). \qquad (10.23)$$

Proof of Lemma 42 For all $t = 1, 2, \ldots, T$, let the set $\mathcal{R}_t \triangleq \mathcal{A}_t \setminus \mathcal{B}_t$; namely, $\mathcal{R}_t$ is the set that remains after the optimal (worst-case) removal $\mathcal{B}_t$ from $\mathcal{A}_t$. Furthermore, let the element $s_{t,2}^i \in \mathcal{S}_{t,2}$ denote the i-th element added in $\mathcal{S}_{t,2}$ per the greedy subroutine in lines 5-8 of Algorithm 20; i.e., $\mathcal{S}_{t,2} = \{s_{t,2}^1, \ldots, s_{t,2}^{\alpha_t - \beta_t}\}$. In addition, for all $i = 1, \ldots, \alpha_t - \beta_t$, denote $\mathcal{S}_{t,2}^i \triangleq \{s_{t,2}^1, \ldots, s_{t,2}^i\}$, and also set $\mathcal{S}_{t,2}^0 \triangleq \emptyset$. Next, order the elements in each $\mathcal{O}_t$ so that $\mathcal{O} = \{o_t^1, \ldots, o_t^{\alpha_t - \beta_t}\}$ and so that if o_t^i is also in $\mathcal{S}_{t,2}$, then it holds $o_t^i = s_{t,2}^i$; i.e., order the elements in each $\mathcal{O}_t$ so that the common elements in $\mathcal{O}_t$ and $\mathcal{S}_{t,2}$ appear at the same index. Moreover, for all $i = 1, \ldots, \alpha_t - \beta_t$, denote $\mathcal{O}_t^i \triangleq \{o_t^1, \ldots, o_t^i\}$, and also set $\mathcal{O}_t^0 \triangleq \emptyset$. Finally, let: $\mathcal{O}_{1:t} \triangleq \mathcal{O}_1 \cup \ldots \cup \mathcal{O}_t$; $\mathcal{O}_{1:0} \triangleq \emptyset$; $\mathcal{S}_{1:t,2} \triangleq \mathcal{S}_{1,2} \cup \ldots \cup \mathcal{S}_{t,2}$; and $\mathcal{S}_{1:0,2} \triangleq \emptyset$. Then, it

Algorithm 21 Local greedy algorithm [12, Section 4].

Input: Integer T; finite ground sets $\mathcal{K}_1, \ldots, \mathcal{K}_T$; set function $f : 2^{\mathcal{K}_1} \times \cdots \times 2^{\mathcal{K}_T} \mapsto \mathbb{R}$ such that f is non-decreasing, non-negative, and $f(\emptyset) = 0$; integers $\delta_1, \ldots, \delta_T$ such that $0 \leq \delta_t \leq |\mathcal{K}_t|$, for all $t = 1, \ldots, T$.

Output: At each step $t = 1, 2, \ldots, T$, set $\mathcal{M}_t$.

1: **for all** $t = 1, \ldots, T$ **do**
2: $\quad \mathcal{M}_t \leftarrow \emptyset$;
3: $\quad$ **while** $|\mathcal{M}_t| < \delta_t$ **do**
4: $\quad\quad x \in \arg\max_{y \in \mathcal{K}_t \setminus \mathcal{M}_t} f(\mathcal{S}_1, \ldots, \mathcal{S}_{t-1}, \mathcal{M}_t \cup \{y\})$;
5: $\quad\quad \mathcal{M}_t \leftarrow \{x\} \cup \mathcal{M}_t$;
6: $\quad$ **end while**
7: **end for**

holds:

$$
\begin{aligned}
& f(\mathcal{O}_1, \ldots, \mathcal{O}_T) \\
& = \sum_{t=1}^{T} \sum_{i=1}^{\alpha_t - \beta_t} f(o_t^i | \mathcal{O}_{1:t-1} \cup \mathcal{O}_t^{i-1}) && (10.24) \\
& \leq \frac{1}{1 - c_f} \sum_{t=1}^{T} \sum_{i=1}^{\alpha_t - \beta_t} f(o_t^i | \mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}) && (10.25) \\
& \leq \frac{1}{1 - c_f} \sum_{t=1}^{T} \sum_{i=1}^{\alpha_t - \beta_t} f(s_{t,2}^i | \mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}) && (10.26) \\
& \leq \frac{1}{(1 - c_f)^2} \sum_{t=1}^{T} \sum_{i=1}^{\alpha_t - \beta_t} f(s_{t,2}^i | \mathcal{S}_{1:t-1,2} \cup \mathcal{S}_{t,2}^{i-1}) && (10.27) \\
& = \frac{1}{(1 - c_f)^2} f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}). && (10.28)
\end{aligned}
$$

where the eqs. (10.24)-(10.28) hold for the following reasons: eq. (10.24) holds due the notation introduced in eq. (10.10); ineq. (10.25) holds since Definition 41 of total curvature implies ineq. (10.6), and since the definition of each o_t^i implies that because $o_t^i \notin \mathcal{O}_t^{i-1}$, then it also is $o_t^i \notin \mathcal{S}_{t,2}^{i-1}$, and as a result, because $o_t^i \notin \mathcal{O}_{1:t-1} \cup \mathcal{O}_t^{i-1}$, then it also is $o_t^i \notin \mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}$ (which fact allows the application of ineq. (10.6)); ineq. (10.26) holds since the element $s_{t,2}^i$ is chosen greedily, given $\mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}$; ineq. (10.27) holds for the same reasons as ineq. (10.25); similarly, eq. (10.28) holds for the same reasons as eq. (10.24). ■

Lemma 43. *Recall the notation in Algorithm 20; in particular, consider the sets $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ selected by Algorithm 20 (lines 3-4 of Algorithm 20). Moreover, consider the notation in Algorithm 21,*[3] *and for all $t = 1, 2, \ldots, T$, let in Algorithm 21 be $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $\delta_t = \alpha_t - \beta_t$. Finally, for all $t = 1, 2, \ldots, T$, let the set $\mathcal{P}_t$ be such that $\mathcal{P}_t \subseteq \mathcal{K}_t$, $|\mathcal{P}_t| \leq \delta_t$,*

[3] The local greedy Algorithm 21 is connected to Algorithm 20 as follows: Algorithm 20 reduces to Algorithm 21 if in Problem 5 we assume no removals; equivalently, if in Algorithm 20 we assume that for all $t = 1, \ldots, T$ it is $\mathcal{B}_t = \emptyset$ (no attacks), and correspondingly, that $\beta_t = 0$, which implies $\mathcal{S}_{t,1} = \emptyset$.

and $f(\mathcal{P}_1, \ldots, \mathcal{P}_T)$ is maximal, that is:

$$(\mathcal{P}_1, \ldots, \mathcal{P}_T) \in \arg \max_{\bar{\mathcal{P}}_1 \subseteq \mathcal{K}_1, |\bar{\mathcal{P}}_1| \leq \delta_1} \cdots \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{K}_T, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\bar{\mathcal{P}}_1, \ldots, \bar{\mathcal{P}}_T). \tag{10.29}$$

Then, it holds:

$$f(\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_T) \geq (1 - c_f) f(\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T). \tag{10.30}$$

Proof of Lemma 43 We use similar notation to the one introduced in the proof of Lemma 42. In addition, —again similarly to the proof of Lemma 42,— we order the elements in each $\mathcal{P}_t$ so that $\mathcal{P}_t = \{p_t^1, \ldots, p_T^{\delta_t}\}$ and so that they appear in the same place as in $\mathcal{M}_t$. Moreover, we let the element $m_t^i \in \mathcal{M}_t$ denote the i-th element added in $\mathcal{M}_t$ per the greedy subroutine in lines 3-6 of Algorithm 21. Then, it holds:

$$\begin{aligned}
&f(\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T) \\
&= \sum_{t=1}^{T} \sum_{i=1}^{\delta_t} f(p_t^i | \mathcal{P}_{1:t-1} \cup \mathcal{P}_t^{i-1}) && (10.31) \\
&\leq \frac{1}{1 - c_f} \sum_{t=1}^{T} \sum_{i=1}^{\delta_t} f(p_t^i | \mathcal{M}_{1:t-1} \cup \mathcal{M}_t^{i-1}) && (10.32) \\
&\leq \frac{1}{1 - c_f} \sum_{t=1}^{T} \sum_{i=1}^{\delta_t} f(m_t^i | \mathcal{M}_{1:t-1} \cup \mathcal{M}_t^{i-1}) && (10.33) \\
&= \frac{1}{1 - c_f} f(\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_T). && (10.34)
\end{aligned}$$

where the eqs. (10.31)-(10.34) hold for the following reasons: eq. (10.31) holds due to the notation introduced in eq. (10.10); ineq. (10.32) holds since Definition 41 of total curvature implies ineq. (10.6), and since the definition of each p_t^i implies that because $p_t^i \notin \mathcal{P}_t^{i-1}$, then it also is $p_t^i \notin \mathcal{M}_t^{i-1}$, and as a result, because $p_t^i \notin \mathcal{P}_{1:t-1} \cup \mathcal{P}_t^{i-1}$, then it also is $p_t^i \notin \mathcal{M}_{1:t-1} \cup \mathcal{M}_t^{i-1}$ (which fact allows the application of ineq. (10.6)); ineq. (10.33) holds since the element m_t^i is chosen greedily, given $\mathcal{M}_{1:t-1} \cup \mathcal{M}_t^{i-1}$; eq. (10.34) holds for the same reasons as eq. (10.31). ■

Corollary 12. *Recall the notation in Algorithm 20. In particular, consider the sets $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ selected by Algorithm 20 (lines 3-4 of Algorithm 20), as well as, the sets $\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}$ selected by Algorithm 20 (lines 5-8 of Algorithm 20). Finally, per the notation of Lemma 43, for all $t = 1, 2, \ldots, T$, consider $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $\delta_t = \alpha_t - \beta_t$, and let the set $\mathcal{P}_t$ be such that $\mathcal{P}_t \subseteq \mathcal{K}_t$, $|\mathcal{P}_t| \leq \delta_t$, and $f(\mathcal{P}_1, \ldots, \mathcal{P}_T)$ is maximal, per eq. (10.29). Then, for the sets $\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}$ selected by Algorithm 20 (lines 5-8 of Algorithm 20), it holds:*

$$f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) \geq (1 - c_f)^3 f(\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T). \tag{10.35}$$

Proof of Corollary 12 The proof follows from Lemma 42 and Lemma 43. In particular, let $\mathcal{O}_t = \mathcal{M}_t$ in ineq. (10.23) to get:

$$f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) \geq (1 - c_f)^2 f(\mathcal{M}_1, \ldots, \mathcal{M}_T). \tag{10.36}$$

Using in ineq. (10.36) the ineq. (10.30), the proof is complete. ■

Lemma 44. *Recall the notation in Theorem 23. In addition, per the notation of Corollary 12, for all $t = 1, 2, \ldots, T$, consider $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $\delta_t = \alpha_t - \beta_t$, and let the set $\mathcal{P}_t$ be such that $\mathcal{P}_t \subseteq \mathcal{K}_t$, $|\mathcal{P}_t| \leq \delta_t$, and $f(\mathcal{P}_1, \ldots, \mathcal{P}_T)$ is maximal, per eq. (10.29). Then, it holds:*

$$f(\mathcal{P}_1, \ldots, \mathcal{P}_T) \geq f(\mathcal{A}^\star_{1:T} \setminus \mathcal{B}^\star(\mathcal{A}^\star_{1:T})). \tag{10.37}$$

Proof of Lemma 44 Consider the following notation: since for each $t = 1, \ldots, T$, it is $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$, let:

$$\begin{aligned} &h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}) \triangleq \\ &\max_{\bar{\mathcal{P}}_1 \subseteq \mathcal{V}_1 \setminus \mathcal{S}_{1,1}, |\bar{\mathcal{P}}_1| \leq \delta_1} \cdots \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \mathcal{S}_{T,1}, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\bar{\mathcal{P}}_1, \ldots, \bar{\mathcal{P}}_T). \end{aligned} \tag{10.38}$$

Given the above notation, for any $\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_T$ such that for all $t = 1, \ldots, T$ it is $\hat{\mathcal{P}}_t \subseteq \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $|\hat{\mathcal{P}}_t| \leq \delta_t$, it holds:

$$h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}) \geq f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_T) \Rightarrow \tag{10.39}$$

$$\begin{aligned} &h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}) \geq \\ &\max_{\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \mathcal{S}_{T,1}, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}, \bar{\mathcal{P}}_T) \Rightarrow \\ &\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}, \bar{\mathcal{B}}_T) \geq \\ &\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \bar{\mathcal{B}}_T, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}, \bar{\mathcal{P}}_T). \end{aligned} \tag{10.40}$$

Denote the right-hand-side of ineq. (10.40) by $z(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1})$. Since $\delta_T = \alpha_T - \beta_T$, and for $\bar{\mathcal{P}}_T$ in ineq. (10.40) it is $\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \bar{\mathcal{B}}_T$ and $|\bar{\mathcal{P}}_T| \leq \delta_T$, then it equivalently holds:

$$\begin{aligned} &z(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}) = \\ &\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \end{aligned} \tag{10.41}$$

Let in ineq. (10.41) be $w(\bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T) \triangleq f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T)$. We prove next that it holds:

$$\begin{aligned} &z(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}) \geq \\ &\max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} w(\bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \end{aligned} \tag{10.42}$$

The proof of ineq. (10.42) is as follows: for any $\hat{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\hat{\mathcal{A}}_T| \leq \alpha_T$, and any $\hat{\mathcal{S}}_{T,1} \subseteq$

$\mathcal{V}_T, |\hat{\mathcal{S}}_{T,1}| \leq \beta_T$, it holds:

$$\max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} w(\bar{\mathcal{A}}_T \setminus \hat{\mathcal{S}}_{T,1}) \geq w(\hat{\mathcal{A}}_T \setminus \hat{\mathcal{S}}_{T,1}) \Rightarrow$$
$$\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} w(\bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T) \geq$$
$$\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} w(\hat{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T), \tag{10.43}$$

and now ineq. (10.43) implies ineq. (10.41). Overall, ineq. (10.40) becomes:

$$\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}, \bar{\mathcal{B}}_T) \geq$$
$$\max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \tag{10.44}$$

The left-hand-side of ineq. (10.44) is a function of $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}$; denote it as $h'(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1})$. Similarly, the right-hand-side of ineq. (10.44) is a function of $\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}$; denote it as $f'(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1})$. Given these notations, ineq. (10.44) is equivalently written as:

$$h'(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}) \geq f'(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-1}), \tag{10.45}$$

which has the same form as ineq. (10.39). Therefore, following the same steps as those we used starting from ineq. (10.39) to prove ineq. (10.44), it holds:

$$\min_{\bar{\mathcal{B}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{B}}_{T-1}| \leq \beta_{T-1}} h'(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-2,1}, \bar{\mathcal{B}}_{T-1}) \geq$$
$$\max_{\bar{\mathcal{A}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{A}}_{T-1}| \leq \alpha_{T-1}} \min_{\bar{\mathcal{B}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{B}}_{T-1}| \leq \beta_{T-1}}$$
$$f'(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_{T-2}, \bar{\mathcal{A}}_{T-1} \setminus \bar{\mathcal{B}}_{T-1}), \tag{10.46}$$

which has the same form as ineq. (10.44). Repeating the same steps as those we used starting from ineq. (10.39) to prove ineq. (10.44) for another $T-2$ times, it holds:

$$\min_{\bar{\mathcal{B}}_{1,} \subseteq \mathcal{V}_1, |\bar{\mathcal{B}}_1| \leq \beta_1} \cdots \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\bar{\mathcal{B}}_1, \ldots \bar{\mathcal{B}}_T) \geq$$
$$\max_{\bar{\mathcal{A}}_1 \subseteq \mathcal{V}_1, |\bar{\mathcal{A}}_1| \leq \alpha_1} \min_{\bar{\mathcal{B}}_1 \subseteq \mathcal{V}_1, |\bar{\mathcal{B}}_1| \leq \beta_1} \cdots \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_1 \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T}$$
$$f(\bar{\mathcal{A}}_1 \setminus \bar{\mathcal{B}}_1, \ldots, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T), \tag{10.47}$$

which is implies ineq. (10.37), since the right-hand-side of ineq. (10.47) is equal to the right-hand-side of ineq. (10.37), and —with respect now to the left-hand-side of ineq. (10.47)— it is:

$$f(\mathcal{P}_1, \ldots, \mathcal{P}_T) \geq$$
$$\min_{\bar{\mathcal{B}}_{1,} \subseteq \mathcal{V}_1, |\bar{\mathcal{B}}_1| \leq \beta_1} \cdots \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\bar{\mathcal{B}}_1, \ldots \bar{\mathcal{B}}_T).$$

■

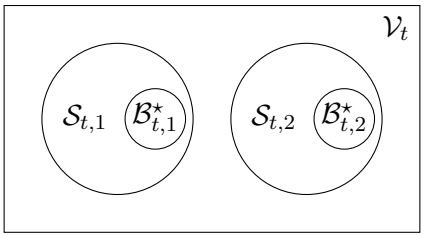


Figure 15: Venn diagram, where the sets $\mathcal{S}_{t,1}, \mathcal{S}_{t,2}, \mathcal{B}^\star_{t,1}, \mathcal{B}^\star_{t,2}$ are as follows: per Algorithm 20, $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$ are such that $\mathcal{A}_t = \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$. In addition, due to their construction, it holds $\mathcal{S}_{t,1} \cap \mathcal{S}_{t,2} = \emptyset$. Next, $\mathcal{B}^\star_{t,1}$ and $\mathcal{B}^\star_{t,2}$ are such that $\mathcal{B}^\star_{t,1} = \mathcal{B}^\star(\mathcal{A}_{1:T}) \cap \mathcal{S}_{t,1}$, and $\mathcal{B}^\star_2 = \mathcal{B}^\star(\mathcal{A}_{1:T}) \cap \mathcal{S}_{t,2}$; therefore, it is $\mathcal{B}^\star_{t,1} \cap \mathcal{B}^\star_{t,2} = \emptyset$ and $\mathcal{B}^\star(\mathcal{A}_{1:T}) = (\mathcal{B}^\star_{1,1} \cup \mathcal{B}^\star_{1,2}) \cup \cdots \cup (\mathcal{B}^\star_{T,1} \cup \mathcal{B}^\star_{T,2})$.

10.7.3. Proof of Theorem 23

We first prove Theorem 23's part 1 (approximation performance), and then, Theorem 23's part 2 (running time).

Proof of Theorem 23's part 1 (approximation performance)

We first prove ineq. (10.8); then, we prove ineq. (10.7).

To the above ends, we use the following notation (along with the notation introduced in Algorithm 20, Theorem 23, and in Appendix 10.7.1): for each $t = 1, \ldots, T$:

- let $\mathcal{S}^+_{t,1} \triangleq \mathcal{S}_{t,1} \setminus \mathcal{B}^\star(\mathcal{A}_{1:T})$, i.e., $\mathcal{S}^+_{t,1}$ is the set of remaining elements in the set $\mathcal{S}_{t,1}$ after the removal from $\mathcal{S}_{t,1}$ of the elements in the optimal (worst-case) removal $\mathcal{B}^\star(\mathcal{A}_{1:T})$;
- let $\mathcal{S}^+_{t,2} \triangleq \mathcal{S}_{t,2} \setminus \mathcal{B}^\star(\mathcal{A}_{1:T})$, i.e., $\mathcal{S}^+_{t,2}$ is the set of remaining elements in the set $\mathcal{S}_{t,2}$ after the removal from $\mathcal{S}_{t,2}$ of the elements in the optimal (worst-case) removal $\mathcal{B}^\star(\mathcal{A}_{1:T})$;
- let the sets $\mathcal{P}_1, \ldots, \mathcal{P}_T$ be a solution to the maximization problem in eq. (10.29) per the conditions in Corollary 12, i.e., for $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $\delta_t = \alpha_t - \beta_t$.

Proof of ineq. (10.8) Consider that the objective function f is non-decreasing and such that (without loss of generality) f is non-negative and $f(\emptyset) = 0$. Then, the proof of

ineq. (10.8) follows by making the following observations:

$$
\begin{aligned}
& f(\mathcal{A}_{1:T} \setminus \mathcal{B}^\star(\mathcal{A}_{1:T})) \\
& \quad = f(\mathcal{S}_{1,1}^+ \cup \mathcal{S}_{1,2}^+, \ldots, \mathcal{S}_{T,1}^+ \cup \mathcal{S}_{T,2}^+) && (10.48) \\
& \quad \geq (1 - c_f) \sum_{t=1}^{T} \sum_{v \in \mathcal{S}_{t,1}^+ \cup \mathcal{S}_{t,2}^+} f(v) && (10.49) \\
& \quad \geq (1 - c_f) \sum_{t=1}^{T} \sum_{v \in \mathcal{S}_{t,2}} f(v) && (10.50) \\
& \quad \geq (1 - c_f)^2 f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) && (10.51) \\
& \quad \geq (1 - c_f)^5 f(\mathcal{P}_1, \ldots, \mathcal{P}_T) && (10.52) \\
& \quad \geq (1 - c_f)^5 f(\mathcal{A}_{1:T}^\star \setminus \mathcal{B}^\star(\mathcal{A}_{1:T}^\star)), && (10.53)
\end{aligned}
$$

where eqs. (10.48) to (10.53) hold for the following reasons: eq. (10.48) follows from the definitions of the sets $\mathcal{S}_{t,1}^+$ and $\mathcal{S}_{t,2}^+$; ineq. (10.49) follows from ineq. (10.48) due to Lemma 40; ineq. (10.50) follows from ineq. (10.49) because for all elements $v \in \mathcal{S}_{t,1}^+$ and all elements $v' \in \mathcal{S}_{t,2} \setminus \mathcal{S}_{t,2}^+$ it is $f(v) \geq f(v')$ (note that due to the definitions of the sets $\mathcal{S}_{t,1}^+$ and $\mathcal{S}_{t,2}^+$ it is $|\mathcal{S}_{t,1}^+| = |\mathcal{S}_{t,2} \setminus \mathcal{S}_{t,2}^+|$, that is, the number of non-removed elements in $\mathcal{S}_{t,1}$ is equal to the number of removed elements in $\mathcal{S}_{t,2}$), and because $\mathcal{S}_{t,2} = (\mathcal{S}_{t,2} \setminus \mathcal{S}_{t,2}^+) \cup \mathcal{S}_{t,2}^+$; ineq. (10.51) follows from ineq. (10.50) due to Corollary 11; ineq. (10.52) follows from ineq. (10.51) due to Corollary 12; finally, ineq. (10.53) follows from ineq. (10.52) due to Lemma 44. The above conclude the proof of ineq. (10.8). ■

Proof of ineq. (10.7) Consider that the objective function f is non-decreasing submodular and such that (without loss of generality) f is non-negative and $f(\emptyset) = 0$. To prove ineq. (10.7) we follow similar observations to the ones we followed in the proof of ineq. (10.8); in particular:

$$
\begin{aligned}
& f(\mathcal{A}_{1:T} \setminus \mathcal{B}^\star(\mathcal{A}_{1:T})) \\
& \quad = f(\mathcal{S}_{1,1}^+ \cup \mathcal{S}_{1,2}^+, \ldots, \mathcal{S}_{T,1}^+ \cup \mathcal{S}_{T,2}^+) && (10.54) \\
& \quad \geq (1 - \kappa_f) \sum_{t=1}^{T} \sum_{v \in \mathcal{S}_{t,1}^+ \cup \mathcal{S}_{t,2}^+} f(v) && (10.55) \\
& \quad \geq (1 - \kappa_f) \sum_{t=1}^{T} \sum_{v \in \mathcal{S}_{t,2}} f(v) && (10.56) \\
& \quad \geq (1 - \kappa_f) f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) && (10.57) \\
& \quad \geq (1 - \kappa_f)^4 f(\mathcal{P}_1, \ldots, \mathcal{P}_T) && (10.58) \\
& \quad \geq (1 - \kappa_f)^4 f(\mathcal{A}_{1:T}^\star \setminus \mathcal{B}^\star(\mathcal{A}_{1:T}^\star)), && (10.59)
\end{aligned}
$$

where eqs. (10.54) to (10.59) hold for the following reasons: eq. (10.54) follows from the definitions of the sets $\mathcal{S}_{t,1}^+$ and $\mathcal{S}_{t,2}^+$; ineq. (10.55) follows from ineq. (10.54) due to Lemma 38;

ineq. (10.56) follows from ineq. (10.55) because for all elements $v \in \mathcal{S}_{t,1}^{+}$ and all elements $v' \in \mathcal{S}_{t,2} \setminus \mathcal{S}_{t,2}^{+}$ it is $f(v) \geq f(v')$ (note that due to the definitions of the sets $\mathcal{S}_{t,1}^{+}$ and $\mathcal{S}_{t,2}^{+}$ it is $|\mathcal{S}_{t,1}^{+}| = |\mathcal{S}_{t,2} \setminus \mathcal{S}_{t,2}^{+}|$, that is, the number of non-removed elements in $\mathcal{S}_{t,1}$ is equal to the number of removed elements in $\mathcal{S}_{t,2}$), and because $\mathcal{S}_{t,2} = (\mathcal{S}_{t,2} \setminus \mathcal{S}_{t,2}^{+}) \cup \mathcal{S}_{t,2}^{+}$; ineq. (10.57) follows from ineq. (10.56) because the set function f is submodular and, as a result, the submodularity Definition 39 implies that for any sets $\mathcal{S} \subseteq \mathcal{V}$ and $\mathcal{S}' \subseteq \mathcal{V}$, it is $f(\mathcal{S}) + f(\mathcal{S}') \geq f(\mathcal{S} \cup \mathcal{S}')$ [70, Proposition 2.1]; ineq. (10.58) follows from ineq. (10.57) due to Corollary 12, along with the fact that since f is monotone submodular it is $c_f = \kappa_f$, per Definition 41 of total curvature; finally, ineq. (10.59) follows from ineq. (10.58) due to Lemma 44. The above conclude the proof of the $(1 - \kappa_f)^4$ part in ineq. (10.7). ■

Proof of Theorem 23's part 2 (running time)

We follow the proof of [56, Part 2 of Theorem 23]. In particular, we complete the proof in two steps, where we denote the time for each evaluation of the objective function f as τ_f: for each $t = 1, \ldots, T$, we first compute the time line 3 of Algorithm 20 needs to be executed, and then the time lines 5-8 of Algorithm 20 need to be executed: line 3 needs $|\mathcal{V}_t|\tau_f + |\mathcal{V}_t| \log(|\mathcal{V}_t|) + |\mathcal{V}_t| + O(\log(|\mathcal{V}_t|))$ time, since it asks for $|\mathcal{V}_t|$ evaluations of f, and their sorting, which takes $|\mathcal{V}_t| \log(|\mathcal{V}_t|) + |\mathcal{V}_t| + O(\log(|\mathcal{V}_t|))$ time, using, e.g., the merge sort algorithm. Lines 5-8 need $(\alpha_t - \beta_t)[|\mathcal{V}_t|\tau_f + |\mathcal{V}_t|]$ time, since the while loop is repeated $\alpha_t - \beta_t$ times, and during each loop at most $|\mathcal{V}_t|$ evaluations of f are needed by line 5, as well as, at most $|\mathcal{V}_t|$ time-steps for a maximal element in line 6 to be found. Overall, Algorithm 20 runs in $(\alpha_t - \beta_t)[|\mathcal{V}_t|\tau_f + |\mathcal{V}_t|] + |\mathcal{V}_t| \log(|\mathcal{V}_t|) + |\mathcal{V}_t| + O(\log(|\mathcal{V}_t|)) = O(|\mathcal{V}_t|(\alpha_t - \beta_t)\tau_f)$ time.

■

Part IV

CONTRIBUTIONS TO RESILIENT SUBMODULAR MAXIMIZATION IN ROBOTICS

CHAPTER 11 : Resilient Active Information Gathering with Mobile Robots

Applications in robotics, such as multi-robot target tracking, involve the execution of information acquisition tasks by teams of mobile robots. However, in failure-prone or adversarial environments, robots get attacked, their communication channels get jammed, and their sensors fail, resulting in the withdrawal of robots from the collective task, and, subsequently, the inability of the remaining active robots to coordinate with each other. As a result, traditional design paradigms become insufficient and, in contrast, *resilient* designs against system-wide *failures and attacks* become important. In general, resilient design problems are hard, and even though they often involve objective functions that are monotone and (possibly) submodular, scalable approximation algorithms for their solution have been hitherto unknown. In this chapter, we provide the first algorithm, enabling the following capabilities: *minimal communication*, i.e., the algorithm is executed by the robots based only on minimal communication between them; *system-wide resiliency*, i.e., the algorithm is valid for any number of denial-of-service attacks and failures; and *provable approximation performance*, i.e., the algorithm ensures for all monotone and (possibly) submodular objective functions a solution that is finitely close to the optimal. We support our theoretical analyses with simulated and real-world experiments, by considering an active information acquisition application scenario, namely, *multi-robot target tracking*.[1]

11.1. Introduction

Advances in robotic miniaturization, perception, and communication [2, 3, 43, 235, 236, 237, 238] envision the deployment of robots to support critical missions such as:

- *Hazardous environmental monitoring*: Deploy a team of mobile robots *to monitor* the radiation flow around a nuclear reactor after an explosion; [43]
- *Adversarial-target tracking*: Deploy a team of agile robots *to track* an adversarial target that moves in a cluttered urban environment, aiming to escape; [3]
- *Search and rescue*: Deploy a team of aerial micro-robots *to localize* people trapped in a burning building; [2]

Each of the above scenarios requires the deployment of a mobile team of robots, where each robot needs to be agile; coordinate its motion with its team in a decentralized way; and navigate itself in unknown, complex, and GPS-denied environments, with the objective of gathering the most information about a process of interest. In particular, the problem of designing the motion of a team of mobile robots to infer the state of a process is known as *active information gathering*.

But in all above mission scenarios the robots operate in failure-prone and adversarial environments, where the robots' can get attacked; their communications channels can get jammed; or their sensors can fail. Therefore, in such failure-prone or adversarial scenarios, *resilient* designs against *worst-case* and *system-wide failures and attacks* become important.

[1] This chapter is based on the paper by B. Schlotfeldt, V. Tzoumas, D. Thakur, and G. J. Pappas [234].

In this chapter we formalize for the first time a problem of *resilient active information gathering* with mobile robots, that goes beyond the traditional objective of (non-resilient) active information gathering, and guards against worst-case failures or attacks that can cause not only the withdrawal of robots from the information gathering task, but also the inability of the remaining robots to jointly optimize their control inputs, due to disruptions to their communication network.

Evidently, resilient active information gathering with mobile robots is a computationally challenging task, since it needs to account for all possible removals of robots from the joint motion-design task, which is a problem of combinatorial complexity. In particular, this computational challenge motivates one of the primary goals in this chapter, namely, to provide a scalable and provably near-optimal *approximation* algorithm for resilient active information gathering with mobile robots.

Related work. Related work on problems of information gathering focuses on the deployment of either static sensors [58, 239, 240] or mobile sensors (mounted on robots) [42, 241, 242, 243, 244, 245, 246, 247, 248, 249] to monitor a target process. Among these works, the line of work [42, 241, 242, 243, 244, 245, 246, 247, 248, 249] is the most relevant to ours, as it considers mobile sensors. In particular, [241, 242, 243, 244, 245] focus on information gathering tasks over non-Gaussian processes, whereas the remaining [42, 246, 247, 248, 249] focus on information gathering tasks over Gaussian processes. The advantage in the latter case is that open-loop robot-motion designs are optimal [42], an observation that led [42, 248, 249] to provide the first scalable, non-myopic robot-motion algorithms for active information gathering, along with sub-optimality guarantees. However, in all of these works, there is no resilience to failures or attacks.

In contrast to robotic control, resilient optimization problems have recently received attention in the literature of set function optimization [56, 231, 250]. However, [56, 231, 250] focus on the resilient selection of a *small subset* of elements in the event of attacks or failures, whereas the information acquisition problem requires the selection of controls for *all* robots over a time horizon. In this chapter, we capitalize on the recent results in [56, 231] and seek to bridge the gap between developments in set function optimization and robotic control design to enable critical missions necessitating resilient active information gathering with mobile robots.

Contributions. We make the following contributions:

- (*Problem definition*) We formalize the problem of *resilient active information gathering with mobile robots* against multi-robot denial-of-service attacks or failures. This is the first work to formalize, address, and demonstrate the importance of this problem.

- (*Solution*) We develop the first algorithm for resilient active information gathering with the following properties:

  - *minimal communication*: it terminates within the same order of communication rounds as state-of-the-art algorithms for (non-resilient) information gathering;

  - *system-wide resiliency*: it is valid for any number of denial-of-service attacks or failures;

  - *provable approximation performance*: for all monotone and (possibly) submodular information gathering objective functions in the active robot set (non-failed robots), it ensures a solution close to the optimal.

- (*Simulations*) We conduct simulations in a variety of multi-robot multi-target tracking scenarios, varying the number of robots, targets, and failures. Our simulations validate the benefits of our approach to achieve resilient robotic control against failures or attacks.

- (*Experiments*) We conduct hardware experiments of multiple quad-rotors tracking static ground targets, to demonstrate visually the necessity for resilient robot motion design against robotic failures or denial-of-service attacks.

Notation. Calligraphic fonts denote sets (e.g., $\mathcal{A}$). Given a set $\mathcal{A}$, then $|\mathcal{A}|$ denotes $\mathcal{A}$'s cardinality; given also a set $\mathcal{B}$, then $\mathcal{A} \setminus \mathcal{B}$ denotes the set of elements in $\mathcal{A}$ that are not in $\mathcal{B}$. Given a random variable v, with mean μ and covariance Σ, then $v \sim \mathsf{N}(\mu, \Sigma)$ denotes that v is a Gaussian random variable.

11.2. Problem Statement

We formalize the problem of resilient active information gathering. To this end, we start with some basic definitions.

11.2.1. Basic definitions

We introduce standard models for the notions *robots*, *target*, *sensors*, and *information objective function* [42].

Robots. Active information gathering utilizes a team of mobile robots to track the evolution of a target process. We denote the set of available robots as $\mathcal{V}$, and model each robot's dynamics as a discrete-time non-linear system:

$$x_{i,t} = f_i(x_{i,t-1}, u_{i,t-1}), \quad i \in \mathcal{V}, \quad t = 1, 2, \ldots, \tag{11.1}$$

where the vector $x_{i,t} \in \mathbb{R}^{n_{x_{i,t}}}$ represents the state of robot i at time t, and the vector $u_{i,t} \in \mathcal{U}_{i,t}$ represents the control input, where $\mathcal{U}_{i,t}$ is a *finite* set of admissible control inputs.

Target. The objective of active information gathering is to track the evolution of a target process. We model the target's evolution as a standard discrete-time (possibly time-varying) linear system with additive process noise:

$$y_t = A_{t-1} y_{t-1} + w_{t-1}, \quad t = 1, 2, \ldots, \tag{11.2}$$

where the vector $y_t \in \mathbb{R}^{n_{y_t}}$ represents the state of the target at time t, the vector $w_{t-1} \in \mathbb{R}^{n_{y_t}}$

represents process noise, and the matrix A_{t-1} has suitable dimension. In addition, we let y_0 be a random variable with covariance $\Sigma_{0|0}$, and w_{t-1} be a random variable with zero mean and covariance W_{t-1} such that w_{t-1} is independent of y_0 and of $w_{t'-1}$, for all $t' \neq t$.

Sensor measurements. We consider the sensor measurements to be linearly dependent on the state of the target,[2] and non-linearly dependent on the robots' state, as follows:

$$z_{i,t} = H_{i,t}(x_{i,t})y_t + v_{i,t}(x_{i,t}), \quad i \in \mathcal{V}, \quad t = 1, 2, \ldots, \tag{11.3}$$

where the vector $z_{i,t} \in \mathbb{R}^{n_{z_{i,t}}}$ is the measurement obtained at time t by the on-board sensor at robot i, the vector $v_{i,t}(x_{i,t}) \in \mathbb{R}^{n_{z_{i,t}}}$ represents measurement noise, and the matrix $H_{i,t}(x_{i,t})$ has suitable dimension. In addition, we let $v_{i,t}(x_{i,t})$ be a random variable with zero mean and covariance $v_{i,t}(x_{i,t})$ such that $v_{i,t}(x_{i,t})$ is independent of y_0, of $w_{t'-1}$, and of $v_{i',t'}$ for all $t' \neq t$, and $i' \neq i$.

Information objective function. The problem of active information gathering requires the team of robots in $\mathcal{V}$ to select their control inputs to maximize the team's tracking capability of a target. To the latter end, we assume the robots to use a Kalman filtering algorithm to track the evolution of the target over an observation time-horizon T. Moreover, we consider the robots' collective tracking capability to be quantified by an information objective function, denoted henceforth by J, that depends *solely* on the Kalman filter's error covariances across all times $t = 1, 2, \ldots, T$. Naturally, the Kalman filter's error covariances depend on the robots' control inputs, as well as on both the target process's initial condition y_0 and the robots' initial conditions $\{x_{i,0} : i \in \mathcal{V}\}$. Overall, given an observation time-horizon T, it is:

$$\begin{aligned} J =& J(u_{1:T}(\mathcal{V})) \triangleq \\ & J[\Sigma_1(u_1(\mathcal{V})), \Sigma_2(u_{1:2}(\mathcal{V})), \ldots, \Sigma_T(u_{1:T}(\mathcal{V}))], \end{aligned} \tag{11.4}$$

where $\Sigma_t(u_{1:t}(\mathcal{V}))$ denotes that Kalman filter's error covariance at time t given the robots' control inputs up to time t, namely, given $u_{1:t}(\mathcal{V}) \triangleq \{u_{i,t'} : \ u_{i,t'} \in \mathcal{U}_{i,t'}, \ i \in \mathcal{V}, \ t' = 1, 2, \ldots, t\}$. Examples of information objective functions of the same form as in eq. (11.4) are the average minimum mean square error $1/T \sum_{t=1}^{T} \mathrm{tr}\,(\Sigma_t)$, the average confidence-ellipsoid volume $1/T \sum_{t=1}^{T} \log\det(\Sigma_t)$ [123, Appendix E], as well as information theoretic objectives such as the mutual information $I(y_t|z_{1:t})$ and conditional entropy $h(y_t|z_{1:t})$ [42], where $z_{1:t} \triangleq \{z_{i,t'} : \ i \in \mathcal{V}, \ \ t' = 1, 2, \ldots, t\}$, i.e., $z_{1:t}$ is the set of measurements collected by all robots' across all times.

11.2.2. Resilient Active Information Gathering

We define next the main problem in this chapter.

Problem 6 (Resilient Active Information Gathering)**.** *Given a time horizon T, consider a set of robots $\mathcal{V}$, with dynamics per eq.* (11.1)*, with sensing capabilities per eq.* (11.3)*,*

[2] This standard modeling consideration is without loss of generality whenever linearization over the current estimate of the target's state is possible.

and with a connected communication network; in addition, consider a target process per eq. (11.2)*; moreover, consider an information gathering objective function J per eq.* (11.4)*; finally, consider a number $\alpha \leq |\mathcal{V}|$. For all robots $i \in \mathcal{V}$, and for all times $t = 1, 2, \ldots, T$, find control inputs $u_{i,t}$ to maximize the objective function J against a* worst-case *failure or attack to the robots in $\mathcal{V}$ that causes the removal α robots from $\mathcal{V}$ at the beginning of time ($t = 0$), as well as the disruption of all communications among the remaining robots in $\mathcal{V}$ across all times ($t = 1, 2, \ldots, T$). Formally:*

$$
\begin{aligned}
&\max_{\substack{u_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1, 2 \ldots, T}} \min_{\mathcal{A} \subseteq \mathcal{V}} \quad J(u_{1:T}(\mathcal{V} \setminus \mathcal{A})) : \\
&\qquad \textit{such that, for all } i \in \mathcal{V}, \quad t = 1, 2, \ldots, T : \\
&\qquad\quad y_t = A_{t-1} y_{t-1} + w_{t-1}, \\
&\qquad\quad x_{i,t} = f_i(x_{i,t-1}, u_{i,t-1}), \\
&\qquad\quad z_{i,t} = H_{i,t}(x_{i,t}) y_{i,t} + v_{i,t}(x_{i,t}), \\
&\qquad\quad u_{i,t} = u_{i,t}(z_{i,1}, z_{i,2}, \ldots, z_{i,t}), \\
&\qquad\quad |\mathcal{A}| \leq \alpha,
\end{aligned}
\tag{11.5}
$$

where for any robot set $\mathcal{R} \subseteq \mathcal{V}$ and any time horizon T, we let $u_{1:T}(\mathcal{R}) \triangleq \{u_{i,t} : \quad u_{i,t} \in \mathcal{U}_{i,t}, \quad i \in \mathcal{R}, \quad t = 1, 2, \ldots, T\}$.

We henceforth denote the problem in eq. (11.5) by:

$$
\mathsf{P}(\mathcal{V}, \alpha), \tag{11.6}
$$

where we stress the dependence of the problem only on the set of robots $\mathcal{V}$, and the maximum number of failures or attacks α. Given an instance of Problem 6, and the notation in eq. (11.6), then the (non-resilient) active information gathering problem is the instance of the problem in eq. (11.5) where $\alpha = 0$, namely, $\mathsf{P}(\mathcal{V}, 0)$. Hence, Problem 6 goes beyond the objective of the active information gathering problem $\mathsf{P}(\mathcal{V}, 0)$, by accounting in the planning process for worst-case failures or attacks that (i) not only may cause the removal of robots from the information gathering task, but also, (ii) they may prevent the remaining robots from jointly re-planning their motion, e.g., due to the caused disruptions to the robots' communication network after the removal of the attacked or failed robots.

11.3. Algorithm for Resilient Active Information gathering

We present the first scalable algorithm for Problem 6, whose pseudo-code is described in Algorithm 22; afterwards, we describe the intuition behind it.

11.3.1. Scalable algorithm for Problem 6

Algorithm 22 is composed of four steps:

Computation of robots' marginal contributions in the absence of attacks (step 1 of Algorithm 22): Each robot $i \in \mathcal{V}$ solves the problem of active information gathering in eq. (11.7),

Algorithm 22 Scalable algorithm for Problem 6.

Input: Time horizon T; set of robots $\mathcal{V}$; dynamics of robots in $\mathcal{V}$, per eq. (11.1); dynamics of target process, per eq. (11.2); sensing capabilities of robots in $\mathcal{V}$, per eq. (11.3); information objective function J, per eq. (11.4); number $\alpha \leq |\mathcal{V}|$, per Problem 6, that represents the maximum number of possible robot removals from $\mathcal{V}$.

Output: Control inputs $u_{i,t}$ for all robots $i \in \mathcal{V}$, and for all times $t = 1, 2, \ldots, T$.

1: Each robot $i \in \mathcal{V}$ computes the value of the (non-resilient) active information gathering problem:

$$\mathsf{P}(\{i\}, 0), \tag{11.7}$$

per the notation in eq. (11.6), and denotes it by q_i.

2: All robots in $\mathcal{V}$ find a subset $\mathcal{L}$ of α robots among them (that is, $\mathcal{L} \subseteq \mathcal{V}$ and $|\mathcal{L}| = \alpha$), such that for all robots $i \in \mathcal{L}$ and all robots $j \in \mathcal{V} \setminus \mathcal{L}$, it is $q_i > q_j$;

3: Each robot in $\mathcal{L}$ adopts the control inputs it computed in Algorithm 22's line 1 by solving the problem in eq. (11.7).

4: The robots in $\mathcal{V} \setminus \mathcal{L}$ compute their control inputs by solving the following active information gathering problem:

$$\mathsf{P}(\mathcal{V} \setminus \mathcal{L}, 0), \tag{11.8}$$

per the notation we introduced in eq. (11.6).

which is an instance of Problem 6 where no other robot participates in the information gathering task, and where no attacks or failures are possible; algorithms to solve such information gathering problems have been proposed in [42, 248, 249]. Overall, each robot $i \in \mathcal{V}$, by solving the problem in eq. (11.7), computes its marginal contribution to the information gathering task in Problem 6 in the absence of any other robot in $\mathcal{V} \setminus \{i\}$, and in the absence of any attacks and failures.

Computation of robot set $\mathcal{L}$ with the α largest marginal contributions in the absence of attacks (step 2 of Algorithm 22): The robots in $\mathcal{V}$ share their marginal contribution to the information gathering task, which they computed in Algorithm 22's step 1, and decide which subset $\mathcal{L}$ of them composes a set of α robots with the α largest marginal contributions; this procedure can be executed with minimal communication (at most $2|\mathcal{V}|$ communication rounds), e.g., by accumulating (through the communication network) to one robot all the marginal contributions $\{q_i : i \in \mathcal{V}\}$, and, then, by letting this robot to select the set $\mathcal{L}$, and to communicate it back to the rest of the robots.

Computation of control inputs of robots in $\mathcal{L}$ (step 3 of Algorithm 22): The robots in the set $\mathcal{L}$, per Algorithm 22's step 2, adopt the control inputs they computed in Algorithm 22's step 1 (e.g., using the algorithm in [42]).

Computation of control inputs of robots in $\mathcal{V} \setminus \mathcal{L}$ (step 4 of Algorithm 22): Given the set of robots $\mathcal{L}$, per Algorithm 22's line 2, the remaining robots in $\mathcal{V} \setminus \mathcal{L}$ jointly solve the problem of active information gathering in eq. (11.8), which is an instance of Problem 6 where the robots in $\mathcal{L}$ do not participate in the information gathering task, and where any attacks or failures are impossible. In particular, the robots in $\mathcal{V} \setminus \mathcal{L}$ can jointly solve the problem

in eq. (11.8) with minimal communication (at most $2|\mathcal{V}|$ communication rounds) using the algorithm *coordinate descent* [249, Section IV].

11.3.2. Intuition behind Algorithm 22

The goal of Problem 6 is to ensure the success of an information gathering task despite failures or attacks that cause the removal of α robots from the task, and, consequently, disruptions to the robot's communication network (due to the robots' previous removal), which prevent the remaining robots from jointly re-planning their motion. In this context, Algorithm 22 aims to fulfill Problem 6's goal first by separating the set of robots $\mathcal{V}$ into two subsets —the set of robots $\mathcal{L}$, and the (remaining) set of robots $\mathcal{V} \setminus \mathcal{L}$ (Algorithm 22's line 1 and line 2),— and second by designing the robots' control inputs in each of the two sets (Algorithm 22's line 3 and line 4). In particular, Algorithm 22 aims with set $\mathcal{L}$ to capture the worst-case attack or failure to α robots among the robots in $\mathcal{V}$; equivalently, the set $\mathcal{L}$ is aimed to act as a "bait" to an attacker that selects the *best* α robots in $\mathcal{V}$ (*best* with respect to the robots' contribution towards attaining the goal of Problem 6). However, the problem of selecting the *best* α robots in $\mathcal{V}$ is a combinatorial problem, and, in general, intractable [13]. Therefore, Algorithm 22 aims to approximate the best α robots in $\mathcal{V}$ by letting the set $\mathcal{L}$ be the set of α robots with the α largest marginal contributions, and, then, it assigns to them the corresponding control inputs (Algorithm 22's line 2 and line 3). Afterwards, given the set $\mathcal{L}$, Algorithm 22 assumes the removal of the robots in $\mathcal{L}$ from $\mathcal{V}$, and coordinates the remaining robots in $\mathcal{V} \setminus \mathcal{L}$ to jointly plan their motion using a decentralized active information gathering algorithm, such as the coordinated descent algorithm proposed in [249, Section IV] (Algorithm 22's line 4).

11.4. Performance Guarantees

We quantify Algorithm 22's performance, by bounding the number of robot communication rounds it requires, as well as, by bounding its approximation performance. To this end, we use the following two notions of curvature for set functions.[3]

11.4.1. Curvature and total curvature of monotone functions

We present the notions of *curvature* and of *total curvature* for non-decreasing set functions. We start with the notions of *monotonicity*, and of *submodularity* for set functions.

Definition 43 (Monotonicity)**.** *Consider any finite set $\mathcal{V}$. The set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* non-decreasing *if and only if for any sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, it holds $g(\mathcal{B}) \geq g(\mathcal{A})$.*

Definition 44 (Submodularity [70, Proposition 2.1])**.** *Consider any finite set $\mathcal{V}$. The set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* submodular *if and only if for any sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, and any element $v \in \mathcal{V}$, it holds $g(\mathcal{A} \cup \{v\}) - g(\mathcal{A}) \geq g(\mathcal{B} \cup \{v\}) - g(\mathcal{B})$.*

In words, a set function g is submodular if and only if it satisfies a diminishing returns property where for any $\mathcal{A} \subseteq \mathcal{V}$ and $v \in \mathcal{V}$, the drop $g(\mathcal{A} \cup \{v\}) - g(\mathcal{A})$ is non-increasing.

Definition 45. (Curvature of monotone submodular functions [33]) *Consider a*

[3] We focus on properties of set functions to quantify Algorithm 22's approximation performance by analyzing the properties of Problem 6's objective function J as a function of the remaining robot set after the removal of a subset of robots from $\mathcal{V}$ (due to failures or attacks).

finite set $\mathcal{V}$ and a non-decreasing submodular set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that (without loss of generality) for any elements $v \in \mathcal{V}$, it is $g(v) \neq 0$. The curvature of g is defined as follows:

$$\kappa_g \triangleq 1 - \min_{v \in \mathcal{V}} \frac{g(\mathcal{V}) - g(\mathcal{V} \setminus \{v\})}{g(v)}. \tag{11.9}$$

Notably, the above notion of curvature implies that for any non-decreasing submodular set function g, it is $0 \leq \kappa_g \leq 1$.

Definition 46. (Total curvature of non-decreasing functions [15, Section 8]) *Consider a finite set $\mathcal{V}$ and a monotone set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$. The total curvature of g is defined as follows:*

$$c_g \triangleq 1 - \min_{v \in \mathcal{V}} \min_{\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}} \frac{g(\{v\} \cup \mathcal{A}) - g(\mathcal{A})}{g(\{v\} \cup \mathcal{B}) - g(\mathcal{B})}. \tag{11.10}$$

The above notion of total curvature implies that for any non-decreasing set function g, it is $0 \leq c_g \leq 1$. Moreover, to connect the notion of total curvature with that of curvature, we note that when a function g is non-decreasing and submodular, then the two notions coincide, i.e., $c_g = \kappa_g$.

11.4.2. Performance analysis for Algorithm 22

We quantify Algorithm 22's approximation performance, as well as, the number of communication rounds it requires.

Theorem 24. (Performance of Algorithm 22) *Consider an instance of Problem 6, and the definitions:*

- *let the number $J^\star$ be the (optimal) value to Problem 6, i.e., it is $J^\star \triangleq \mathsf{P}(\mathcal{V}, \alpha)$;*
- *given any control inputs $u_{1:T}(\mathcal{V})$ for the robots in $\mathcal{V}$, let the set $\mathcal{A}^\star[u_{1:T}(\mathcal{V})]$ be a (worst-case) removal of α robots from $\mathcal{V}$, i.e., $\mathcal{A}^\star[u_{1:T}(\mathcal{V})] \triangleq \arg\min_{\mathcal{A} \subseteq \mathcal{V}} J(u_{1:T}(\mathcal{V} \setminus \mathcal{A}))$;*
- *given any removal of a subset of robots $\mathcal{A}$ from the robot set $\mathcal{V}$ (due to attacks or failures), call the remaining robot set $\mathcal{V} \setminus \mathcal{A}$* active robot set.

Finally, consider the robots in $\mathcal{V}$ to solve optimally the problems in Algorithm 22's step 1 and step 4, using an algorithm that terminates in ρ communication rounds.

1. *(Approximation performance) Algorithm 22 returns control inputs $u_{1:T}(\mathcal{V})$ such that:*
   - *If the objective function J is non-decreasing and submodular in the active robot set, and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then, it is:*
   $$\frac{J\{u_{1:T}[\mathcal{V} \setminus \mathcal{A}^\star(u_{1:T}(\mathcal{V})]\}}{J^\star} \geq \max\left(1 - \kappa_J, \frac{1}{1+\alpha}\right), \tag{11.11}$$
   where κ_J is the curvature of J (Definition 45).
   - *If the objective function J is non-decreasing in the active robot set, and (without*

loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then, it is:

$$\frac{J\{u_{1:T}[\mathcal{V} \setminus \mathcal{A}^\star(u_{1:T}(\mathcal{V})]\}}{J^\star} \geq (1 - c_J)^2, \tag{11.12}$$

where c_J is the total curvature of J (Definition 46).

2. *(Communication rounds) Algorithm 22 terminates in at most $2|\mathcal{V}| + \rho$ communication rounds.*

Theorem 24 implies on Algorithm 22's performance:

Near-optimality: For both monotone submodular and merely monotone information objective functions, Algorithm 22 guarantees a value for Problem 6 which is finitely close to the optimal. For example, per ineq. (11.11), the approximation factor of Algorithm 22 is bounded by $1/(1 + \alpha)$, which, for any finite number of robots $|\mathcal{V}|$, is non-zero.

Approximation difficulty: For both monotone submodular and merely monotone information objective functions, when the curvature κ_J or the total curvature c_J, respectively, tend to zero, Algorithm 22 becomes exact since for $\kappa_J \to 0$ and $c_J \to 0$ the terms $1 - \kappa_J$ and $1 - c_J$ in ineq. (11.11) and ineq. (11.12) tend to 1. Overall, Algorithm 22's curvature-dependent approximation bounds make a first step towards separating the classes of monotone submodular and merely monotone information objective functions into functions for which Problem 6 can be approximated well (low curvature functions), and functions for which it cannot (high curvature functions).

Overall, Theorem 24 quantifies Algorithm 22's approximation performance when the robots in $\mathcal{V}$ solve optimally the problems in Algorithm 22's step 1 and step 4. However, the problems in Algorithm 22's step 1 and step 4 are computationally challenging, and only approximation algorithms are known for their solution, among which the recently proposed *coordinate descent* [249, Section IV]; in particular, coordinate descent has the advantages of being scalable and of having provable approximation performance. We next quantify Algorithm 22's performance when the robots in $\mathcal{V}$ solve the problem in Algorithm 22's step 4 using coordinate descent (we refer the reader to AppendixA for a description of coordinate descent).

Proposition 12. *Consider an instance of Problem 6, and the notation introduced in Theorem 24. Finally, consider the robots in $\mathcal{V}$ to solve the problem in Algorithm 22's step 1 optimally, and the problem in Algorithm 22's step 4 using* coordinate descent *[249, Section IV].*

1. *(Approximation performance) Algorithm 22 returns control inputs $u_{1:T}(\mathcal{V})$ such that:*

   - *If the objective function J is non-decreasing and submodular in the active robot set, and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then, it is:*

   $$\frac{J(u_{1:T}(\mathcal{V}))}{J^\star} \geq \frac{\max\left(1 - \kappa_J, 1/(1 + \alpha)\right)}{2}. \tag{11.13}$$

- *If the objective function J is non-decreasing in the active robot set, and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then, it is:*

$$\frac{J(u_{1:T}(\mathcal{V}))}{J^\star} \geq \frac{(1 - c_J)^3}{2}. \tag{11.14}$$

2. *(Communication rounds) Algorithm 22 terminates in at most $3|\mathcal{V}|$ communication rounds.*

Proposition 12 implies on Algorithm 22's performance:

Approximation performance for low curvature: For both monotone submodular and merely monotone information objective functions, when the curvature κ_J or the total curvature c_J, respectively, tend to zero, Algorithm 22 recovers the same approximation performance as that of the state-of-the-art algorithms for (non-resilient) active information gathering Algorithm 22 calls as subroutines. For example, for submodular information objective functions, the algorithm for active information gathering *coordinate descent* [249, Section IV] has approximation performance at least 1/2 the optimal [249, Theorem 2], and, per Proposition 12, when Algorithm 22 calls as subroutine this algorithm, it has approximation performance at least $(1 - \kappa_J)/2$ the optimal, which tends to 1/2 for $\kappa_J \to 0$.

Approximation performance for no failures or attacks: For submodular information objective functions, and for zero number of failures or attacks ($\alpha = 0$), Algorithm 22's approximation performance becomes the same as that of the state-of-the-art algorithms for (non-resilient) active information gathering Algorithm 22 calls as subroutines. In particular, for submodular information objective functions, the algorithm for active information gathering *coordinate descent* [249, Section IV] has approximation performance at least 1/2 the optimal, and, per Proposition 12, when Algorithm 22 calls as subroutine this algorithm, it has approximation performance at least 1/2 the optimal for $\alpha = 0$, since it is $1/(1 + 0) = 1$ in ineq. (11.13).

Minimal communication: Algorithm 22, even though it goes beyond the objective of (non-resilient) active information gathering, by accounting for attacks or failures, it terminates within the same order of communication rounds as state-of-the-art algorithms for (non-resilient) active information gathering. In particular, the algorithm for active information gathering *coordinate descent* [249, Section IV] terminates in at most $|\mathcal{V}|$ rounds, and, per Proposition 12, when Algorithm 22 calls as a subroutine this algorithm, then it terminates in at most $3|\mathcal{V}|$ rounds; evidently, $|\mathcal{V}|$ and $3|\mathcal{V}|$ have the same order.

Summary of theoretical results. Overall, Algorithm 22 is the first algorithm for Problem 6, and it enjoys the following:

- *minimal communication*: Algorithm 22 terminates within the same order of communication rounds as state-of-the-art algorithms for (non-resilient) information gathering;
- *system-wide resiliency*: Algorithm 22 is valid for any number of denial-of-service attacks and failures;
- *provable approximation performance*: Algorithm 22 ensures for all monotone and (pos-

sibly) submodular objective functions a solution finitely close to the optimal.

11.5. Application: Multi-target tracking with mobile robots

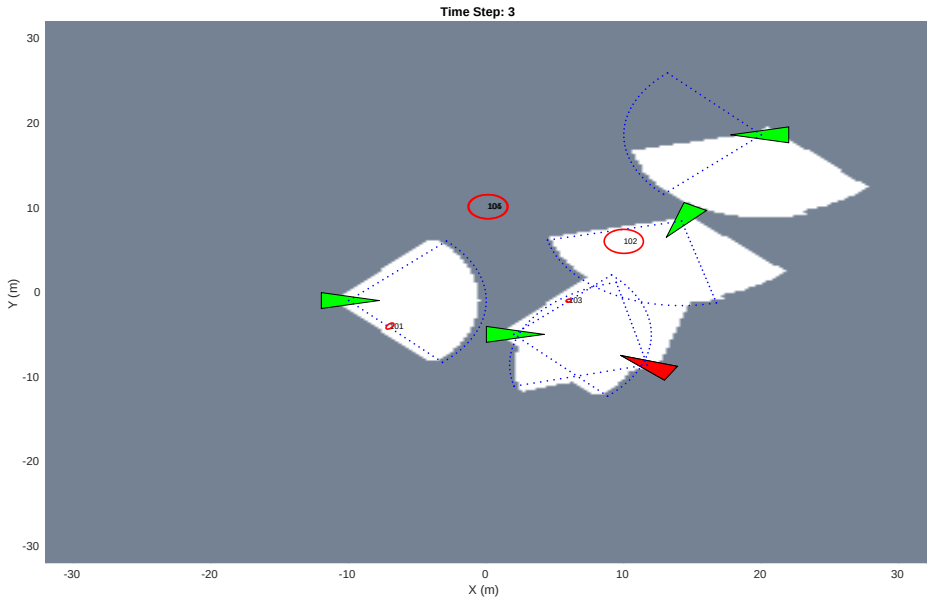


Figure 16: Simulation environment depicting five robots. The jammed robot is indicated in red.

We motivate the importance of Problem 6, as well as, demonstrate the performance of Algorithm 22, by considering an application of active information gathering, namely, *multi-target tracking with mobile robots*. In particular, the application's setting is as follows: a team $\mathcal{V}$ of mobile robots is tasked with tracking the position of M moving targets. In more detail, each robot moves according to unicycle dynamics on $SE(2)$, discretized with a sampling period τ:

$$\begin{pmatrix} x^1_{t+1} \\ x^2_{t+1} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x^1_t \\ x^2_t \\ \theta_t \end{pmatrix} + \begin{pmatrix} \nu \operatorname{sinc}(\frac{\omega\tau}{2}) \cos(\theta_t + \frac{\omega\tau}{2}) \\ \nu \operatorname{sinc}(\frac{\omega\tau}{2}) \sin(\theta_t + \frac{\omega\tau}{2}) \\ \tau\omega \end{pmatrix}. \tag{11.15}$$

The set of admissible controls is given by $\mathcal{U} := \{(\nu, \omega) : \nu \in \{1, 3\}$ m/s, $\omega \in \{0, \pm 1, \pm 3\}$ rad/s$\}$.

The targets move according to double integrator dynamics, corrupted with additive Gaussian noise. For M targets, their state at time t is $y_t = \left[y_{t,1}^\top y_{t,2}^\top, \ldots, y_{t,M}^\top\right]^\top$ where $y_{t,m}$ contains the planar coordinates and velocities of the m-th target, denoted by $(y^1, y^2, \dot{y^1}, \dot{y^2})$. The model is:

$$y_{t+1,m} = A \begin{bmatrix} I_2 & \tau I_2 \\ 0 & I_2 \end{bmatrix} y_{t,m} + w_t, \quad w_t \sim \mathsf{N}\left(0, q \begin{bmatrix} \tau^3/3 I_2 & \tau^2/2 I_2 \\ \tau^2/2 I_2 & \tau I_2 \end{bmatrix}\right).$$

The sensor observation model consists of a range and bearing for each target $m \in \{0, \ldots, M-$

1}:

$$z_{t,m} = h(x_t, y_{t,m}) + v_t, \quad v_t \sim \mathsf{N}\left(0, V(x_t, y_{t,m})\right);$$
$$h(x, y_m) = \begin{bmatrix} r(x, y_m) \\ \alpha(x, y_m) \end{bmatrix} := \begin{bmatrix} \sqrt{(y^1 - x^1)^2 + (y^2 - x^2)^2} \\ \tan^{-1}((y^2 - x^2)(y^1 - x^1)) - \theta \end{bmatrix}.$$

We note that since the sensor observation model is non-linear, we linearize it around the predicted target trajectory $y \neq x$:

$$\nabla_y h(x, y_m) = \frac{1}{r(x, y_m)} \begin{bmatrix} (y^1 - x^1) & (y^2 - x^2) & 0_{1x2} \\ -\sin(\theta + \alpha(x, y_m)) & \cos(\theta + \alpha(x, y_m)) & 0_{1x2} \end{bmatrix}.$$

The observation model for the joint target state can then be expressed as a block diagonal matrix containing the linearized observation models for each target along the diagonal, i.e.,

$$H \triangleq \operatorname{diag}\left(\nabla_{y_1} h(x, y_1), \ldots, \nabla_{y_M} h(x, y_M)\right).$$

The sensor noise covariance grows linearly in range and in bearing, up to σ_r^2, and σ_b^2, where σ_r and σ_b are the standard deviation of the range and the bearing noise, respectively. The model here also includes a limited range and field of view, denoted by the parameters r_{sense} and ψ, respectively.

Finally, as information objective function, in the simulations we use the average log determinant of the covariance matrix[248, 249]. Overall, we solve an instance of Problem 6 with the aforementioned constraints, and the monotone objective function [5]:

$$J \triangleq \frac{1}{T} \sum_{t=1}^{T} \log\det(\Sigma_t),$$

where $\Sigma_{t+1} = \rho_{t+1}^e(\rho_t^p(\Sigma_t), x_{t+1})$ is the Kalman filtering Riccati map [42].[4] We use the subroutines described in [248] and [249] for the step 1 and step 4 of Algorithm 22, respectively.

11.5.1. Simulations on multi-target tracking with mobile robots

We use simulations to evaluate the performance of our Algorithm 22 across different scenarios. In particular, we vary the number of robots, n, the number of targets M, and the number of attacks α. In each of these scenarios we compare the performance of the resilient Algorithm 22 with that of the non-resilient algorithm *coordinate descent* [249, Section IV]. To this end, we consider two information performance measures: the average entropy and average root mean square error (RMSE) per target, averaged over the robots in the team. We describe the parameters of the simulation: the robots and targets in the environment are restricted to move inside a 64x64 meter environment, as in Fig. 16. For the evaluation,

[4] We remark that the problem scenario is dependent on a prior distribution of the target's initial conditions y_0 and $\Sigma_{0|0}$. Notwithstanding, if a prior distribution is unknown, an exploration strategy can be incorporated to find the targets by placing exploration landmarks at the map frontiers [249].

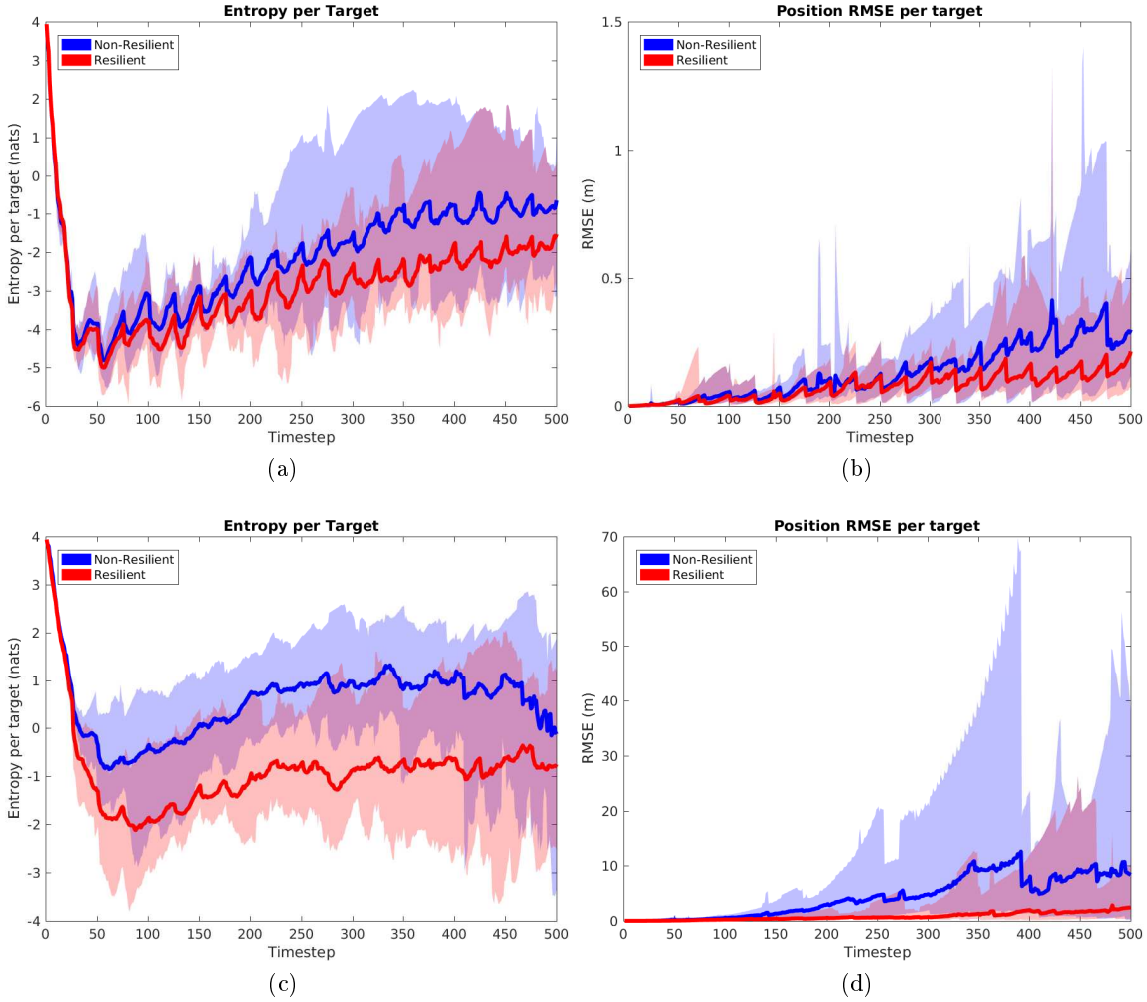


Figure 17: The figures depict the average entropy and position RMSE (root mean square error) per target, averaged over the robots. Figs. (a-b) were obtained from a simulation with 10 robots, 10 targets, with 2 jamming attacks. Figs. (c-d) have the same configuration but up to 6 jamming attacks. The blue colors correspond to the non-resilient algorithm, and the red colors correspond to the resilient algorithm. The shaded regions are the spread between the minimum and maximum values of the information measure, and the solid lines are the mean value. The plots are the aggregate of ten trials, each executed over 500 time-steps.

we fix the initial positions of both the robots and targets, and the robots are given a prior distribution of the targets before starting the simulation. The targets start with a zero velocity, and in the event that a target leaves the environment its velocity is reflected to remain in bounds. Across all simulations we fix the remaining parameters as follows: $T = 25$, $\tau = 0.5$, $r_{sense} = 10$, $\psi = 94°$, $\sigma_r = .15$m, $\sigma_b = 5°$, $q = .001$. Finally, we run Algorithm 6 in a receding horizon fashion every T time-steps, for a total of 500 steps, and average each configuration over 10 trials. The robots are forced to execute the entire T-step trajectory without re-planning, due to the jamming attack that occurs at the onset of every planning

	Mean RMSE (m)		Peak RMSE (m)	
	NR	Resilient	NR	Resilient
$n = 5$, $M = 10$				
$\alpha = 1$	0.28	0.19	9.62	2.09
$\alpha = 2$	1.47	0.68	26.07	15.71
$\alpha = 4$	10.67	4.9	225.47	103.82
$n = 10$, $M = 5$				
$\alpha = 2$	0.35	0.14	57.65	1.87
$\alpha = 4$	0.39	0.28	6.66	3.17
$\alpha = 6$	2.07	0.65	93.27	15.63
$n = 10$, $M = 10$				
$\alpha = 2$	0.13	0.08	1.4	1.32
$\alpha = 4$	0.24	0.23	4.19	2.66
$\alpha = 6$	4.39	1.2	69.77	26.4

TABLE I: The table depicts the estimation performance, measured by average and peak RMSE per tracked target, for a variety of configurations. The number n denotes the number of mobile sensors, (i.e., $n = |\mathcal{V}|$), M denotes the number of moving targets, and α denotes the number of failures. NR denotes the non-resilient algorithm, while Resilient is Algorithm 1. All results are across 500 timesteps, averaged over ten trials per configuration.

phase. Our results are depicted in Fig. 17 and Table I.
We observe in Fig. 17 that the performance of the resilient Algorithm 22 is superior both with respect to the average entropy and the RMSE per target. Importantly, as the number of jamming attacks grows, the Algorithm 22's superiority becomes more pronounced, and for the non-resilient algorithm the peaks in RMSE error grow much larger.

Table I suggests that the resilient Algorithm 22 achieves a lower average error than the non-resilient algorithm, and, crucially, is highly effective in reducing the peak estimation error; in particular, Algorithm 22 achieves a performance that is 2 to 30 times better in comparison to the performance achieved by the non-resilient algorithm. We also observe that the impact of Algorithm 22 is most prominent when the number of attacks is large relative to the size of the robot team.

11.5.2. Experiments on multi-target tracking with mobile robots

We implement Algorithm 22 in a multi-UAV scenario with two quadrotors tracking the positions of two static ground targets, shown in Fig. 18. The UAV trajectories are computed off-board but in *real-time* on a laptop with an Intel Core i7 CPU. The UAVs are localized using the Vicon Motion Capture system. The UAVs are quad-rotors equipped with Qualcomm Flight^{TM}. The UAVs use Vicon pose estimates to generate noisy measurements corresponding to a downward facing camera which has a 360° field-of-view, and a 1 meter sensing radius. The UAVs move in a 4x8 meter testing laboratory environment with no obstacles. One robot is jammed at all times.

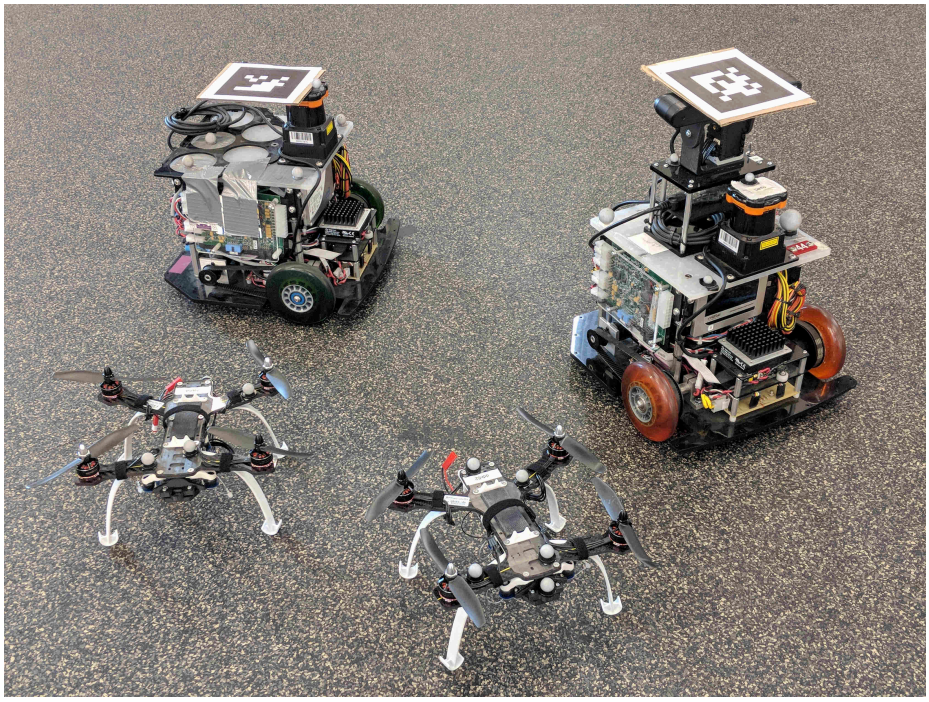


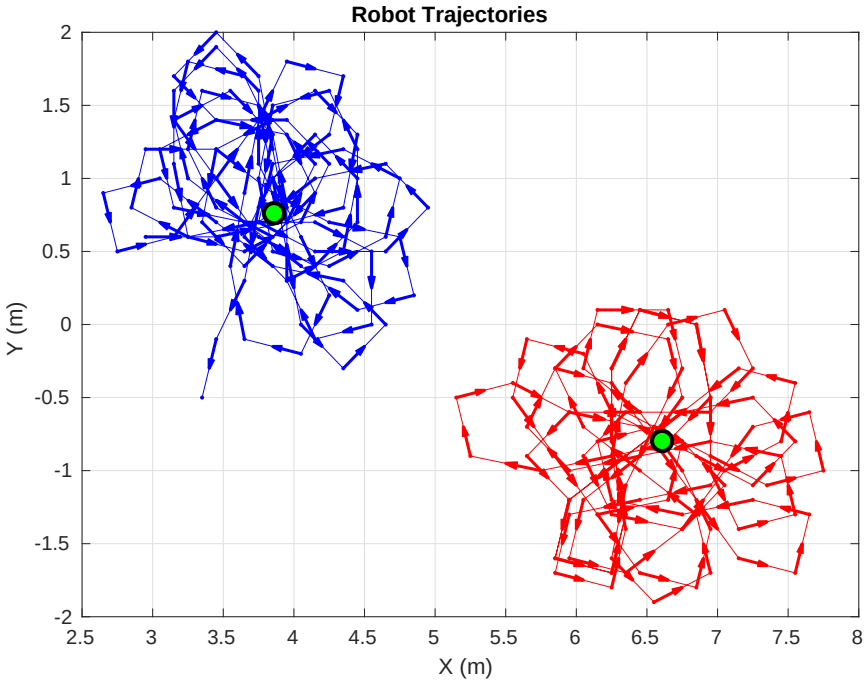
Figure 18: The experimental setup with two quad-rotors equipped with Qualcomm Flight™, and two Scarabs as ground targets.

The goal of the hardware experiments is to acquire a visual interpretation of the properties of the trajectories designed using the resilient Algorithm 22. To isolate the effect of resilience, we simplify the problem to static targets (i.e. stationary) and to the smallest possible team, i.e., 2 robots.

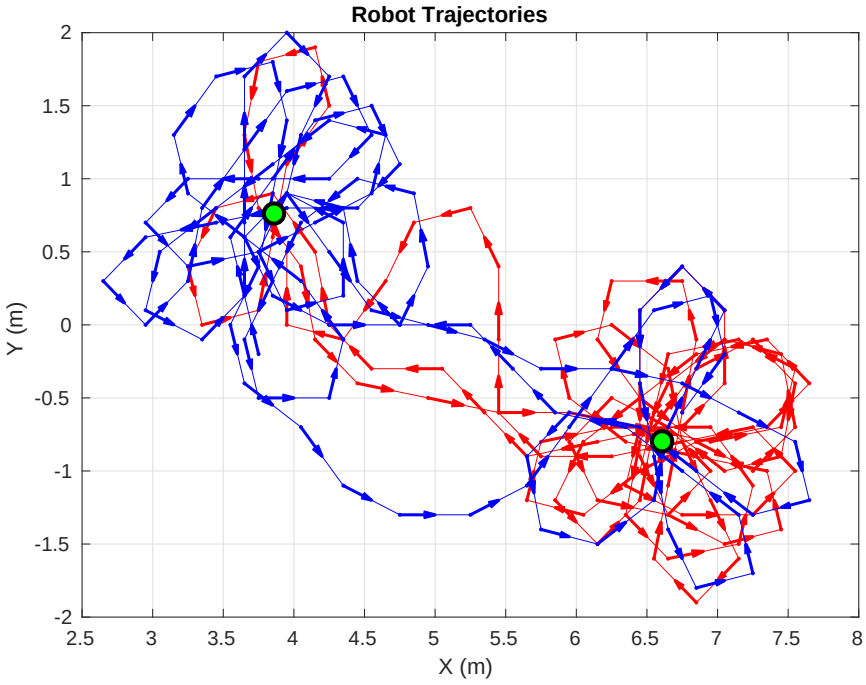
We observe from the experiments that the trajectories planned by the UAVs under the non-resilient algorithm stick to the target they are closest to, whereas under the resilient Algorithm 22, the UAVs switch amongst the two targets (Fig. 19). Intuitively, the reason is that the resilient algorithm always assumes that one of the robots will fail, in which case the optimal strategy for one UAV is to track two targets is to switch amongst the targets, whereas the non-resilient algorithm assumes that none of the robots will fail, in which case the optimal strategy for two UAVs is to allocate themselves to the closest target. When there is the possibility of one UAV failing, switching amongst the targets is preferable, since both robots have information about both targets.

11.6. Concluding Remarks & Future Work

We made the first steps to ensure the success of critical active information gathering tasks against failures and denial-of-service attacks, per Problem 6. In particular, we provided the first algorithm for Problem 6, and proved it guarantees near-optimal performance against system-wide failures, even with minimal robot communication. We motivated the need for resilient active information gathering, and showcased the success of our algorithm, with simulated and real-world experiments in a series of multi-robot target tracking scenarios.



(a)



(b)

Figure 19: The plot in (a) depicts the experimental robot trajectories in the non-resilient algorithm. The figure in (b) depicts the resilient algorithm. The targets are in green.

This chapter opens a number of avenues for future research, both in theory and in applications. Future work in theory includes the resilient design of the robot's communication network against network-wide failures, to balance the trade-off between *minimal communication* and *connectedness*, which is necessitated in scenarios that are both resource constrained (e.g., where bandwidth or battery is limited), and failure-prone (e.g., where attacks can disrupt communication links). Future work in applications includes the experimental testing of resilient active information gathering with mobile robots in environmental monitoring, search and rescue scenarios, and simultaneous localization and mapping.

11.7. Appendix: Proof of Results

11.7.1. Preliminary lemmas and definitions

Notation. In the appendix we use the following notation to support the proofs in this chapter: in particular, consider a finite ground set $\mathcal{V}$ and a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$. Then, for any set $\mathcal{X} \subseteq \mathcal{V}$ and any set $\mathcal{X}' \subseteq \mathcal{V}$, the symbol $f(\mathcal{X}|\mathcal{X}')$ denotes the marginal value $f(\mathcal{X} \cup \mathcal{X}') - f(\mathcal{X}')$. Moreover, the symbol κ_f is the total curvature of f (Definition 45), and the symbol c_f is the total curvature of f (Definition 46).

This appendix contains lemmas that will support the proof of Theorem 24 in this chapter; moreover, it contains a generalized description of the algorithm *coordinate descent* [249, Section IV] (to any non-decreasing information objective function in the active robot set), and a lemma, which will support the proof of Proposition 12 in this chapter.

Lemmas that support the proof of Theorem 24

The proof of the lemmas is also found in [56, 231].

Lemma 45. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing and submodular set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. For any $\mathcal{A} \subseteq \mathcal{V}$, it is:*

$$f(\mathcal{A}) \geq (1 - \kappa_f) \sum_{a \in \mathcal{A}} f(a).$$

Proof of Lemma 45 Let $\mathcal{A} = \{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$. We prove Lemma 47 by proving the following two inequalities:

$$f(\mathcal{A}) \geq \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V} \setminus \{a_i\}), \tag{11.16}$$

$$\sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V} \setminus \{a_i\}) \geq (1 - \kappa_f) \sum_{i=1}^{|\mathcal{A}|} f(a_i). \tag{11.17}$$

We begin with the proof of ineq. (11.16):

$$
\begin{aligned}
f(\mathcal{A}) &= f(\mathcal{A}|\emptyset) && (11.18)\\
&\geq f(\mathcal{A}|\mathcal{V}\setminus\mathcal{A}) && (11.19)\\
&= \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V}\setminus\{a_i, a_{i+1}, \ldots, a_{|\mathcal{A}|}\}) && (11.20)\\
&\geq \sum_{i=1}^{|\mathcal{A}|} f(a_i|\mathcal{V}\setminus\{a_i\}), && (11.21)
\end{aligned}
$$

where ineqs. (11.19) to (11.21) hold for the following reasons: ineq. (11.19) is implied by eq. (11.18) because f is submodular and $\emptyset \subseteq \mathcal{V}\setminus\mathcal{A}$; eq. (11.20) holds since for any sets $\mathcal{X}\subseteq\mathcal{V}$ and $\mathcal{Y}\subseteq\mathcal{V}$ it is $f(\mathcal{X}|\mathcal{Y}) = f(\mathcal{X}\cup\mathcal{Y}) - f(\mathcal{Y})$, and it also $\{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ denotes the set $\mathcal{A}$; and ineq. (11.21) holds since f is submodular and $\mathcal{V}\setminus\{a_i, a_{i+1}, \ldots, a_\mu\} \subseteq \mathcal{V}\setminus\{a_i\}$. These observations complete the proof of ineq. (11.16).

We now prove ineq. (11.17) using the Definition 45 of κ_f, as follows: since $\kappa_f = 1 - \min_{v\in\mathcal{V}} \frac{f(v|\mathcal{V}\setminus\{v\})}{f(v)}$, it is implied that for all elements $v\in\mathcal{V}$ it is $f(v|\mathcal{V}\setminus\{v\}) \geq (1-\kappa_f)f(v)$. Therefore, adding the latter inequality across all elements $a\in\mathcal{A}$ completes the proof of ineq. (11.17). ■

Lemma 46. *Consider any finite ground set $\mathcal{V}$, a non-decreasing and submodular function $f: 2^{\mathcal{V}} \mapsto \mathbb{R}$ and non-empty sets $\mathcal{Y}, \mathcal{P} \subseteq \mathcal{V}$ such that for all elements $y\in\mathcal{Y}$ and all elements $p\in\mathcal{P}$ it is $f(y) \geq f(p)$. Then, it is:*

$$f(\mathcal{P}|\mathcal{Y}) \leq |\mathcal{P}| f(\mathcal{Y}).$$

Proof of Lemma 46 Consider any element $y\in\mathcal{Y}$ (such an element exists since Lemma 46 considers that $\mathcal{Y}$ is non-empty); then,

$$
\begin{aligned}
f(\mathcal{P}|\mathcal{Y}) &= f(\mathcal{P}\cup\mathcal{Y}) - f(\mathcal{Y}) && (11.22)\\
&\leq f(\mathcal{P}) + f(\mathcal{Y}) - f(\mathcal{Y}) && (11.23)\\
&= f(\mathcal{P})\\
&\leq \sum_{p\in\mathcal{P}} f(p) && (11.24)\\
&\leq |\mathcal{P}| \max_{p\in\mathcal{P}} f(p)\\
&\leq |\mathcal{P}| f(y) && (11.25)\\
&\leq |\mathcal{P}| f(\mathcal{Y}), && (11.26)
\end{aligned}
$$

where eq. (11.22) to ineq. (11.26) hold for the following reasons: eq. (11.22) holds since for any sets $\mathcal{X}\subseteq\mathcal{V}$ and $\mathcal{Y}\subseteq\mathcal{V}$, it is $f(\mathcal{X}|\mathcal{Y}) = f(\mathcal{X}\cup\mathcal{Y}) - f(\mathcal{Y})$; ineq. (11.23) holds since f is submodular and, as a result, the submodularity Definition 44 implies that for any set $\mathcal{A}\subseteq\mathcal{V}$ and $\mathcal{A}'\subseteq\mathcal{V}$, it is $f(\mathcal{A}\cup\mathcal{A}') \leq f(\mathcal{A}) + f(\mathcal{A}')$; ineq. (11.24) holds for the same reason

as ineq. (11.23); ineq. (11.25) holds since or all elements $y \in \mathcal{Y}$ and all elements $p \in \mathcal{P}$ it is $f(y) \geq f(p)$; finally, ineq. (11.26) holds because f is monotone and $y \in \mathcal{Y}$. ■

Lemma 47. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. For any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, it is:*

$$f(\mathcal{A} \cup \mathcal{B}) \geq (1 - c_f) \left(f(\mathcal{A}) + \sum_{b \in \mathcal{B}} f(b) \right).$$

Proof of Lemma 47 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i | \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}). \tag{11.27}$$

In addition, Definition 46 of total curvature implies:

$$\begin{aligned} f(b_i | \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}) &\geq (1 - c_f) f(b_i | \emptyset) \\ &= (1 - c_f) f(b_i), \end{aligned} \tag{11.28}$$

where the latter equation holds since $f(\emptyset) = 0$. The proof is completed by substituting (11.28) in (11.27) and then taking into account that $f(\mathcal{A}) \geq (1 - c_f) f(\mathcal{A})$ since $0 \leq c_f \leq 1$. ■

Lemma 48. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. For any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \setminus \mathcal{B} \neq \emptyset$, it is:*

$$f(\mathcal{A}) + (1 - c_f) f(\mathcal{B}) \geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}) + f(\mathcal{A} \cap \mathcal{B}).$$

Proof of Lemma 48 Let $\mathcal{A} \setminus \mathcal{B} = \{i_1, i_2, \ldots, i_r\}$, where $r = |\mathcal{A} - \mathcal{B}|$. From Definition 46 of total curvature c_f, for any $i = 1, 2, \ldots, r$, it is $f(i_j | \mathcal{A} \cap \mathcal{B} \cup \{i_1, i_2, \ldots, i_{j-1}\}) \geq (1 - c_f) f(i_j | \mathcal{B} \cup \{i_1, i_2, \ldots, i_{j-1}\})$. Summing these r inequalities,

$$f(\mathcal{A}) - f(\mathcal{A} \cap \mathcal{B}) \geq (1 - c_f) \left(f(\mathcal{A} \cup \mathcal{B}) - f(\mathcal{B}) \right),$$

which implies the lemma. ■

Corollary 13. *Consider a finite ground set $\mathcal{V}$ and a non-decreasing set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that f is non-negative and $f(\emptyset) = 0$. For any set $\mathcal{A} \subseteq \mathcal{V}$ and any set $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, it is:*

$$f(\mathcal{A}) + \sum_{b \in \mathcal{B}} f(b) \geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}).$$

Proof of Corollary 13 Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$.

$$\begin{aligned} f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i) &\geq (1 - c_f) f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i)) \qquad (11.29) \\ &\geq (1 - c_f) f(\mathcal{A} \cup \{b_1\}) + \sum_{i=2}^{|\mathcal{B}|} f(b_i) \\ &\geq (1 - c_f) f(\mathcal{A} \cup \{b_1, b_2\}) + \sum_{i=3}^{|\mathcal{B}|} f(b_i) \\ &\vdots \\ &\geq (1 - c_f) f(\mathcal{A} \cup \mathcal{B}), \end{aligned}$$

where (11.29) holds since $0 \leq c_f \leq 1$, and the rest due to Lemma 48 since $\mathcal{A} \cap \mathcal{B} = \emptyset$ implies $\mathcal{A} \setminus \{b_1\} \neq \emptyset$, $\mathcal{A} \cup \{b_1\} \setminus \{b_2\} \neq \emptyset$, ..., $\mathcal{A} \cup \{b_1, b_2, \ldots, b_{|\mathcal{B}|-1}\} \setminus \{b_{|\mathcal{B}|}\} \neq \emptyset$.

■

Generalized Coordinate Descent and a lemma that supports the proof of Proposition 12

In this section we generalize the proof in [249] that the algorithm *coordinate descent* proposed therein guarantees for the information objective function of mutual information an approximation performance up to a multiplicative factor 1/2 the optimal. In particular, we extend the proof to *any* non-decreasing and submodular information objective function, as well as to *any* non-decreasing information objective function.

The algorithm coordinate descent works as follows: consider an *arbitrary* ordering of the robots in $\mathcal{V}$, such that $\mathcal{V} \equiv \{1, 2, \ldots, n\}$, and suppose that robot 1 chooses first its controls, without considering the other robots; in other words, robot 1 solves the single robot version of Problem 6, i.e. $\mathsf{P}(\{1\}, 0)$, to obtain control inputs $u^{cd}_{1:T}(\{1\})$ such that:

$$u^{cd}_{1:T}(\{1\}) \in \underset{\hat{u}_t \in \mathcal{U}_{1,t}, t=1,2,\ldots,T}{\arg\min} J(\hat{u}_{1:T}). \qquad (11.30)$$

Afterwards, robot 1 communicates its chosen control sequence to robot 2, and robot 2, given the control sequence of robot 1, computes its control input as follows:

$$u^{cd}_{1:T}(\{2\}) \in \underset{\hat{u}_t \in \mathcal{U}_{2,t}, t=1,2,\ldots,T}{\arg\min} J(u^{cd}_{1:T}(\{1\}), \hat{u}_{1:T}). \qquad (11.31)$$

This continues such that robot $i+1$ solves a single robot problem, given the control inputs

from the robots $1, 2, \ldots, i$:

$$u^{cd}_{1:T}(\{i\}) \in \underset{\hat{u}_t \in \mathcal{U}_{2,t}, t=1,2,\ldots,T}{\arg\min} J(u^{cd}_{1:T}(\{1, 2, \ldots, i\}), \hat{u}_{1:T}). \tag{11.32}$$

Notably, if we let $u^*_{1:T}(\{i\})$ be the control inputs for the i-th robot resulting from the optimal solution to the n robot problem, then from the coordinate descent algorithm it is:

$$J(u^{cd}_{1:T}(\{1, 2, \ldots, i\}), u^*_{1:T}(\{i\})) \leq J(u^{cd}_{1:T}(\{1, 2, \ldots, i\}). \tag{11.33}$$

Lemma 49. (Approximation performance of coordinate descent) *Consider a set of robots $\mathcal{V}$, and an instance of problem $\mathsf{P}(\mathcal{V}, 0)$, per eq. (11.6). Denote the optimal control inputs for problem $\mathsf{P}(\mathcal{V}, 0)$, across all robots and all times, by $u^*_{1:T}(\mathcal{V})$. The coordinate descent algorithm returns control inputs $u^{cd}_{1:T}(\mathcal{V})$, across all robots and all times, such that:*

- *if the objective function J is non-decreasing submodular in the active robot set, and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then, it is:*

$$\frac{J(u^{cd}_{1:T}(\mathcal{V}))}{J(u^*_{1:T}(\mathcal{V}))} \geq \frac{1}{2}. \tag{11.34}$$

- *If the objective function J is non-decreasing in the active robot set, and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then, it is:*

$$\frac{J(u^{cd}_{1:T}(\mathcal{V}))}{J(u^*_{1:T}(\mathcal{V}))} \geq \frac{1 - c_J}{2}. \tag{11.35}$$

Proof of Lemma 49 For notational simplicity, assume an ordering among the robots in $\mathcal{V}$, and let $\mathcal{V} = \{1, 2, \ldots, n\}$, and $u_{\mathcal{A}} \triangleq u_{1:T}(\mathcal{A})$ for some set $\mathcal{A}$ of active robots. Moreover, let $J(u^a_{\mathcal{A}}, u^b_{\mathcal{B}})$ be the value of the objective function when the robots in set $\mathcal{A}$ design controls with a scheme a, and robots in set $\mathcal{B}$ design controls with scheme b. Then:

- if the objective function J is non-decreasing and submodular in the active robot set,

and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then:

$$J(u^*_{1:n}) \leq J(u^*_{1:n}) + \sum_{i=1}^{n}[J(u^{cd}_{1:i}, u^*_{i+1:n}) \tag{11.36}$$
$$- J(u^{cd}_{1:i-1}, u^*_{i+1:n})]$$
$$= J(u^{cd}_{1:n}) + \sum_{i=1}^{n}[J(u^{cd}_{1:i-1}, u^*_{i:n}) \tag{11.37}$$
$$- J(u^{cd}_{1:i-1}, u^*_{i+1:n})]$$
$$= J(u^{cd}_{1:n}) + \sum_{i=1}^{n} J(u^*_i|\{u^{cd}_{1:i-1}, u^*_{i+1,n}\}) \tag{11.38}$$
$$\leq J(u^{cd}_{1:n}) + \sum_{i=1}^{n} J(u^*_i|u^{cd}_{1:i-1}) \tag{11.39}$$
$$\leq J(u^{cd}_{1:n}) + \sum_{i=1}^{n} J(u^{cd}_i|u^{cd}_{1:i-1}) \tag{11.40}$$
$$= J(u^{cd}_{1:n}) + J(u^{cd}_{1:n}) \tag{11.41}$$
$$\leq 2J(u^{cd}_{1:n}), \tag{11.42}$$

where ineq. (11.36) holds due to monotonicity of J; eq. 11.37) is a shift in indexes of the first term in the sum; eq. (11.38) is an expression of the sum as a sum of marginal gains; ineq. (11.39) holds due to submodularity; ineq. (11.40) holds by the coordinate-descent policy (per eq. (11.33)); eq. (11.41) holds due to the definition of the marginal gain symbol $J(u^*_i|u^{cd}_{1:i-1})$ (for any $i = 1, 2, \ldots, n$) as $J(u^*_i, u^{cd}_{1:i-1}) - J(u^{cd}_{1:i-1})$; finally, a re-arrangement of the terms in eq. (11.42) gives $J(u^{cd}_{1:n})/J(u^*_{1:n}) \geq 1/2$.

- If the objective function J is non-decreasing in the active robot set, and (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$, then multiplying both sides of

eq. (11.38) (which holds for any non-decreasing J) with $(1-c_J)$, we have:

$$\begin{aligned}
&(1-c_J)J(u^*_{1:n}) \\
&\quad = (1-c_J)J(u^{cd}_{1:n}) + \\
&\qquad\qquad (1-c_J)\sum_{i=1}^{n} J(u^*_i|\{u^{cd}_{1:i-1}, u^*_{i+1,n}\}) \\
&\quad \leq J(u^{cd}_{1:n}) + (1-c_J)\sum_{i=1}^{n} J(u^*_i|\{u^{cd}_{1:i-1}, u^*_{i+1,n}\}) && (11.43)\\
&\quad \leq J(u^{cd}_{1:n}) + \sum_{i=1}^{n} J(u^*_i|u^{cd}_{1:i-1}) && (11.44)\\
&\quad \leq J(u^{cd}_{1:n}) + \sum_{i=1}^{n} J(u^{cd}_i|u^{cd}_{1:i-1}) && (11.45)\\
&\quad = J(u^{cd}_{1:n}) + J(u^{cd}_{1:n}) && (11.46)\\
&\quad \leq 2J(u^{cd}_{1:n}), && (11.47)
\end{aligned}$$

where, ineq. (11.43) holds since $0 \leq c_J \leq 1$; ineq. (11.44) holds since J is non-decreasing in the set of active robots, and Definition 46 of total curvature implies that for any non-decreasing set function $g : 2^{\mathcal{V}} \mapsto \mathbb{R}$, for any element $v \in \mathcal{V}$, and for any set $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}$, it is:

$$(1-c_g)g(v|\mathcal{B}) \leq g(\{v\}|\mathcal{A}); \tag{11.48}$$

ineq. (11.45) holds by the coordinate-descent algorithm; eq. (11.46) holds due to the definition of the marginal gain symbol $J(u^*_i|u^{cd}_{1:i-1})$ (for any $i = 1, 2, \ldots, n$) as $J(u^*_i, u^{cd}_{1:i-1}) - J(u^{cd}_{1:i-1})$; finally, a re-arrangement of terms gives $J(u^{cd}_{1:n})/J(u^*_{1:n}) \geq (1-c_J)/2$. ■

11.7.2. Proof of Theorem 24

We first prove Theorem 24's part 1 (approximation performance), and then, Theorem 24's part 2 (communication rounds).

Proof of Theorem 24's part 1 (approximation performance)

The proof follows the steps of the proof of [56, Theorem 1] and of the proof of [231, Theorem 1].

We first prove ineq. (11.11); then, we prove ineq. (11.12).

To the above ends, we use the following notation (along with the notation introduced in Theorem 24 and in Appendix A): given that using Algorithm 22 the robots in $\mathcal{V}$ select control inputs $u_{1:T}(\mathcal{V})$, then, for notational simplicity:

- for any active robot set $\mathcal{R} \subseteq \mathcal{V}$, let $J(\mathcal{R}) \triangleq J[u_{1:T}(\mathcal{R})]$.

- let $\mathcal{A}^\star \triangleq \mathcal{A}^\star[u_{1:T}(\mathcal{V})]$;
- let $\mathcal{L}^+ \triangleq \mathcal{L} \setminus \mathcal{A}^\star$, i.e., $\mathcal{S}_1$ are the remaining robots in $\mathcal{L}$ after the removal of the robots in $\mathcal{A}^\star$;
- let $(\mathcal{V} \setminus \mathcal{L})^+ \triangleq (\mathcal{V} \setminus \mathcal{L}) \setminus \mathcal{A}^\star$, i.e., $\mathcal{S}_2$ are the remaining robots in $\mathcal{V} \setminus \mathcal{L}$ after the removal of the robots in $\mathcal{A}^\star$.

Proof of ineq. (11.11) Consider that the objective function J is non-decreasing and submodular in the active robot set, such that (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$. We first prove the part $1 - \kappa_J$ of the bound in the right-hand-side of ineq. (11.11), and then, the part $h(|\mathcal{V}|, \alpha)$ of the bound in the right-hand-side of ineq. (11.11).

To prove the part $1 - \kappa_J$ of the bound in the right-hand-side of ineq. (11.11), we follow the steps of the proof of [56, Theorem 1], and make the following observations:

$$
\begin{aligned}
&J(\mathcal{V} \setminus \mathcal{A}^\star) \\
&\quad = J(\mathcal{L}^+ \cup (\mathcal{V} \setminus \mathcal{L})^+) && (11.49)\\
&\quad \geq (1 - \kappa_J) \sum_{v \in \mathcal{L}^+ \cup (\mathcal{V} \setminus \mathcal{L})^+} J(v) && (11.50)\\
&\quad \geq (1 - \kappa_J) \left(\sum_{v \in (\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+} J(v) + \sum_{v \in (\mathcal{V} \setminus \mathcal{L})^+} J(v) \right) && (11.51)\\
&\quad \geq (1 - \kappa_J) J\{[(\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+] \cup (\mathcal{V} \setminus \mathcal{L})^+\} && (11.52)\\
&\quad = (1 - \kappa_J) J(\mathcal{V} \setminus \mathcal{L}), && (11.53)
\end{aligned}
$$

where eq. (11.49) to (11.53) hold for the following reasons: eq. (11.49) follows from the definitions of the sets $\mathcal{L}^+$ and $(\mathcal{V} \setminus \mathcal{L})^+$; ineq. (11.50) follows from ineq. (11.49) due to Lemma 45; ineq. (11.51) follows from ineq. (11.50) because for all elements $v \in \mathcal{L}^+$ and all elements $v' \in (\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+$ it is $J(v) \geq J(v')$ (note that due to the definitions of the sets $\mathcal{L}^+$ and $(\mathcal{V} \setminus \mathcal{L})^+$ it is $|\mathcal{L}^+| = |(\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+|$, that is, the number of non-removed elements in $\mathcal{L}$ is equal to the number of removed elements in $\mathcal{V} \setminus \mathcal{L}$); finally, ineq. (11.52) follows from ineq. (11.51) because the set function J is submodular and, as a result, the submodularity Definition 44 implies that for any sets $\mathcal{S} \subseteq \mathcal{V}$ and $\mathcal{S}' \subseteq \mathcal{V}$, it is $J(\mathcal{S}) + J(\mathcal{S}') \geq J(\mathcal{S} \cup \mathcal{S}')$ [70, Proposition 2.1]. We now complete the proof of the part $1 - \kappa_J$ of the bound in the right-hand-side of ineq. (11.11) by proving that in ineq. (11.53) it is:

$$J(\mathcal{V} \setminus \mathcal{L}) \geq J^\star, \tag{11.54}$$

when the robots in $\mathcal{V}$ solve optimally the problems in Algorithm 22's step 4, per the statement of Theorem 24. In particular, if for any active robot set $\mathcal{R} \subseteq \mathcal{V}$, we let $\bar{u}_{1:T}(\mathcal{R}) \triangleq \{\bar{u}_{i,t} : \quad \bar{u}_{i,t} \in \mathcal{U}_{i,t}, \quad i \in \mathcal{R}, \quad t = 1, 2, \ldots, T\}$ denote a collection of control inputs to the

robots in $\mathcal{R}$, then it is:

$$J(\mathcal{V} \setminus \mathcal{L}) \equiv \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \mathcal{L})] \tag{11.55}$$

$$\geq \min_{\substack{\bar{\mathcal{L}} \subseteq \mathcal{V}, \\ |\bar{\mathcal{L}}| \leq \alpha}} \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \bar{\mathcal{L}})] \tag{11.56}$$

$$\geq \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} \min_{\substack{\bar{\mathcal{L}} \subseteq \mathcal{V}, \\ |\bar{\mathcal{L}}| \leq \alpha}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \bar{\mathcal{L}})] \tag{11.57}$$

$$\equiv J^{\star}; \tag{11.58}$$

where the ineqs. (11.55)-(11.58) hold for the following reasons: the equivalence in eq. (11.55) holds since the robots in $\mathcal{V}$ solve optimally the problems in Algorithm 22's step 4, per the statement of Theorem 24; (11.56) holds since we minimize over the set $\mathcal{L}$; (11.57) holds because for any set $\hat{\mathcal{L}} \subseteq \mathcal{V}$ and any control inputs $\hat{u}_{1:T}(\mathcal{R}) \triangleq \{\hat{u}_{i,t} : \ \hat{u}_{i,t} \in \mathcal{U}_{i,t}, \ \ i \in \mathcal{R}, \ \ t = 1,2,\ldots,T\}$:

$$\max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \hat{\mathcal{L}})] \geq J[\hat{u}_{1:T}(\mathcal{V} \setminus \hat{\mathcal{L}})] \Rightarrow$$

$$\min_{\substack{\bar{\mathcal{L}} \subseteq \mathcal{V}, \\ |\bar{\mathcal{L}}| \leq \alpha}} \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \bar{\mathcal{L}})] \geq$$

$$\min_{\substack{\bar{\mathcal{L}} \subseteq \mathcal{V}, \\ |\bar{\mathcal{L}}| \leq \alpha}} J[\hat{u}_{1:T}(\mathcal{V} \setminus \bar{\mathcal{L}})] \Rightarrow$$

$$\min_{\substack{\bar{\mathcal{L}} \subseteq \mathcal{V}, \\ |\bar{\mathcal{L}}| \leq \alpha}} \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \bar{\mathcal{L}})] \geq$$

$$\max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1,2\ldots,T}} \min_{\substack{\bar{\mathcal{L}} \subseteq \mathcal{V}, \\ |\bar{\mathcal{L}}| \leq \alpha}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \bar{\mathcal{L}})],$$

where the last one is eq. (11.57); finally, the equivalence in eq. (11.58) holds since $J^{\star}$ (per the statement of Theorem 24) denotes the optimal value to Problem 6. Overall, we proved that ineq. (11.58) proves ineq. (11.54); and, now, the combination of ineq. (11.53) and ineq. (11.54) proves the part $1 - \kappa_J$ of the bound in the right-hand-side of ineq. (11.11).

We finally prove the part $1/(1+\alpha)$ of the bound in the right-hand-side of ineq. (11.11), and complete this way the proof of Theorem 24. To this end, we follow the steps of the proof of [56, Theorem 1], and use the notation introduced in Fig. 20, along with the following notation:

$$\eta \triangleq \frac{J(\mathcal{A}_2^{\star} | \mathcal{V} \setminus \mathcal{A}^{\star})}{J(\mathcal{V} \setminus \mathcal{L})} \tag{11.59}$$

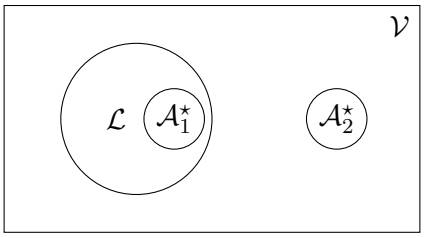


Figure 20: Venn diagram, where the set $\mathcal{L}$ is the robot set defined in step 2 of Algorithm 22, and the set $\mathcal{A}_1^\star$ and the set $\mathcal{A}_2^\star$ are such that $\mathcal{A}_1^\star = \mathcal{A}^\star \cap \mathcal{L}$, and $\mathcal{A}_2^\star = \mathcal{A}^\star \cap (\mathcal{V} \setminus \mathcal{L})$ (observe that these definitions imply $\mathcal{A}_1^\star \cap \mathcal{A}_2^\star = \emptyset$ and $\mathcal{A}^\star = \mathcal{A}_1^\star \cup \mathcal{A}_2^\star$).

Later in this proof, we prove $0 \leq \eta \leq 1$. We first observe that:

$$J(\mathcal{V} \setminus \mathcal{A}^\star) \geq \max\{J(\mathcal{V} \setminus \mathcal{A}^\star), J(\mathcal{L}^+)\}; \tag{11.60}$$

in the following paragraphs, we prove the three inequalities:

$$J(\mathcal{V} \setminus \mathcal{A}^\star) \geq (1-\eta) J(\mathcal{V} \setminus \mathcal{L}), \tag{11.61}$$

$$J(\mathcal{L}^+) \geq \eta \frac{1}{\alpha} J(\mathcal{V} \setminus \mathcal{L}), \tag{11.62}$$

$$\max\{(1-\eta), \eta \frac{1}{\alpha}\} \geq \frac{1}{\alpha+1}. \tag{11.63}$$

Then, if we substitute ineq. (11.61), ineq. (11.62) and ineq. (11.63) to ineq. (11.60), and take into account that $J(\mathcal{V} \setminus \mathcal{L}) \geq 0$, then:

$$J(\mathcal{V} \setminus \mathcal{A}^\star) \geq \frac{1}{\alpha+1} J(\mathcal{V} \setminus \mathcal{L}),$$

which implies the part $1/(1+\alpha)$ of the bound in the right-hand-side of ineq. (11.11), after taking into account ineq. (11.54).

We next complete the proof of the part $1/(1+\alpha)$ of the bound in the right-hand-side of ineq. (11.11) by proving $0 \leq \eta \leq 1$, ineq. (11.61), ineq. (11.62), and ineq. (11.63).

Proof of ineq. $0 \leq \eta \leq 1$ We first prove $\eta \geq 0$, and then $\eta \leq 1$: $\eta \geq 0$, since $\eta \equiv J(\mathcal{A}_2^\star | \mathcal{V} \setminus \mathcal{A}^\star)/J(\mathcal{V} \setminus \mathcal{L})$, and J is non-negative; and $\eta \leq 1$, since $J(\mathcal{V} \setminus \mathcal{L}) \geq J(\mathcal{A}_2^\star)$, due to monotonicity of J and that $\mathcal{A}_2^\star \subseteq \mathcal{V} \setminus \mathcal{L}$, and $J(\mathcal{A}_2^\star) \geq J(\mathcal{A}_2^\star | \mathcal{V} \setminus \mathcal{A}^\star)$, due to submodularity of J and that $\emptyset \subseteq \mathcal{V} \setminus \mathcal{A}^\star$.

Proof of ineq. (11.61) We complete the proof of ineq. (11.61) in two steps. First, it can be verified that:

$$\begin{aligned} &f(\mathcal{V} \setminus \mathcal{A}^\star) = f(\mathcal{V} \setminus \mathcal{L}) - \\ &J(\mathcal{A}_2^\star | \mathcal{V} \setminus \mathcal{A}^\star) + J(\mathcal{L} | \mathcal{V} \setminus \mathcal{L}) - J(\mathcal{A}_1^\star | \mathcal{V} \setminus \mathcal{A}_1^\star), \end{aligned} \tag{11.64}$$

since for any sets $\mathcal{X} \subseteq \mathcal{V}$ and $\mathcal{Y} \subseteq \mathcal{V}$, it is $J(\mathcal{X}|\mathcal{Y}) \equiv J(\mathcal{X} \cup \mathcal{Y}) - J(\mathcal{Y})$. Second, eq. (11.64) implies ineq. (11.61), since $J(\mathcal{A}_2^\star|\mathcal{V} \setminus \mathcal{A}^\star) = \eta J(\mathcal{V} \setminus \mathcal{L})$, and $J(\mathcal{L}|\mathcal{V} \setminus \mathcal{L}) - J(\mathcal{A}_1^\star|\mathcal{V} \setminus \mathcal{A}_1^\star) \geq 0$; the latter is true due to the following two observations: $J(\mathcal{L}|\mathcal{V} \setminus \mathcal{L}) \geq J(\mathcal{A}_1^\star|\mathcal{V} \setminus \mathcal{L})$, since J is monotone and $\mathcal{A}_1^\star \subseteq \mathcal{L}$; and $J(\mathcal{A}_1^\star|\mathcal{V} \setminus \mathcal{L}) \geq J(\mathcal{A}_1^\star|\mathcal{V} \setminus \mathcal{A}_1^\star)$, since J is submodular and $\mathcal{V} \setminus \mathcal{L} \subseteq \mathcal{V} \setminus \mathcal{A}_1^\star$ (see also Fig. 20).

Proof of ineq. (11.62) To prove ineq. (11.62), since it is $\mathcal{A}_2^\star \neq \emptyset$ (and, as a result, also $\mathcal{L}^+ \neq \emptyset$), and for all elements $a \in \mathcal{L}^+$ and all elements $b \in \mathcal{A}_2^\star$, it is $J(a) \geq J(b)$, from Lemma 46 we have:

$$\begin{aligned} J(\mathcal{A}_2^\star|\mathcal{L}^+) &\leq |\mathcal{A}_2^\star| J(\mathcal{L}^+) \\ &\leq \alpha J(\mathcal{L}^+), \end{aligned} \tag{11.65}$$

since $|\mathcal{A}_2^\star| \leq \alpha$. Overall,

$$J(\mathcal{L}^+) \geq \frac{1}{\alpha} J(\mathcal{A}_2^\star|\mathcal{L}^+) \tag{11.66}$$

$$\geq \frac{1}{\alpha} J(\mathcal{A}_2^\star|\mathcal{L}^+ \cup (\mathcal{V} \setminus \mathcal{L})^+) \tag{11.67}$$

$$= \frac{1}{\alpha} J(\mathcal{A}_2^\star|\mathcal{V} \setminus \mathcal{A}^\star) \tag{11.68}$$

$$= \eta \frac{1}{\alpha} J(\mathcal{V} \setminus \mathcal{L}), \tag{11.69}$$

where ineq. (11.66) to eq. (11.69) hold for the following reasons: ineq. (11.66) follows from ineq. (11.65); ineq. (11.67) holds since J is submodular and $\mathcal{L}^+ \subseteq \mathcal{L}^+ \cup (\mathcal{V} \setminus \mathcal{L})^+$; eq. (11.68) holds due to the definitions of the sets $\mathcal{L}^+$, $(\mathcal{V} \setminus \mathcal{L})^+$ and $\mathcal{A}^\star$; finally, eq. (11.69) holds due to the definition of η. Overall, the latter derivation concludes the proof of ineq. (11.62).

Proof of ineq. (11.63) Let $b = 1/\alpha$. We complete the proof first for the case where $(1-\eta) \geq \eta b$, and then for the case $(1-\eta) < \eta b$: i) When $(1-\eta) \geq \eta b$, $\max\{(1-\eta), \eta b\} = 1-\eta$ and $\eta \leq 1/(1+b)$. Due to the latter, $1 - \eta \geq b/(1+b) = 1/(\alpha+1)$ and, as a result, (11.63) holds. ii) When $(1-\eta) < \eta b$, $\max\{(1-\eta), \eta b\} = \eta b$ and $\eta > 1/(1+b)$. Due to the latter, $\eta b > b/(1+b)$ and, as a result, (11.63) holds.

We completed the proof of $0 \leq \eta \leq 1$, and of ineqs. (11.61), (11.62) and (11.63). Thus, we also completed the proof of the part $1/(1+\alpha)$ of the bound in the right-hand-side of ineq. (11.11), and, in sum, the proof of ineq. (11.11).

Proof of ineq. (11.12) Consider that the objective function J is non-decreasing in the active robot set, such that (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$.

The proof follows the steps of the proof of [231, Theorem 1], by making the following

observations:

$$
\begin{aligned}
& J(\mathcal{V} \setminus \mathcal{A}^\star) \\
& \quad = J(\mathcal{L}^+ \cup (\mathcal{V} \setminus \mathcal{L})^+) && (11.70) \\
& \quad \geq (1 - c_J) \sum_{v \in \mathcal{L}^+ \cup (\mathcal{V} \setminus \mathcal{L})^+} J(v) && (11.71) \\
& \quad \geq (1 - c_J) \left(\sum_{v \in (\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+} J(v) + \sum_{v \in (\mathcal{V} \setminus \mathcal{L})^+} J(v) \right) && (11.72) \\
& \quad \geq (1 - c_J)^2 J\{[(\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+] \cup (\mathcal{V} \setminus \mathcal{L})^+\} && (11.73) \\
& \quad = (1 - c_J)^2 J(\mathcal{V} \setminus \mathcal{L}), && (11.74)
\end{aligned}
$$

where eq. (11.70) to (11.74) hold for the following reasons: eq. (11.70) follows from the definitions of the sets $\mathcal{L}^+$ and $(\mathcal{V} \setminus \mathcal{L})^+$; ineq. (11.71) follows from ineq. (11.70) due to Lemma 47; ineq. (11.72) follows from ineq. (11.71) because for all elements $v \in \mathcal{L}^+$ and all elements $v' \in (\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+$ it is $J(v) \geq J(v')$ (note that due to the definitions of the sets $\mathcal{L}^+$ and $(\mathcal{V} \setminus \mathcal{L})^+$ it is $|\mathcal{L}^+| = |(\mathcal{V} \setminus \mathcal{L}) \setminus (\mathcal{V} \setminus \mathcal{L})^+|$, that is, the number of non-removed elements in $\mathcal{L}$ is equal to the number of removed elements in $\mathcal{V} \setminus \mathcal{L}$); finally, ineq. (11.73) follows from ineq. (11.72) because the set function J is non-decreasing and Corollary 13 applies. Overall, the combination of ineq. (11.74) and ineq. (11.54) (observe that ineq. (11.54) still holds if the objective function J is merely non-decreasing) proves ineq. (11.12). ■

Proof of Theorem 24's part 2 (communication rounds)

We described the steps of Algorithm 22 in Section 11.3.1. In particular, Algorithm 22 is composed of four steps:

Computation of robots' marginal contributions in the absence of attacks (step 1 of Algorithm 22) This step requires zero rounds of communication among the robots, since each robot $i \in \mathcal{V}$, by solving the problem in eq. (11.7), merely computes its own marginal contribution to the information gathering task in Problem 6 in the absence of any other robot in $\mathcal{V} \setminus \{i\}$, and in the absence of any attacks and failures.

Computation of robot set $\mathcal{L}$ with the α largest marginal contributions in the absence of attacks (step 2 of Algorithm 22) This step requires at most $2|\mathcal{V}|$ communication rounds, since in this step the robots in $\mathcal{V}$ share their marginal contribution to the information gathering task, which they computed in Algorithm 22's step 1, and decide which subset $\mathcal{L}$ of them composes a set of α robots with the α largest marginal contributions; this procedure can be executed with minimal communication (at most $2|\mathcal{V}|$ communication rounds), e.g., by accumulating (through the communication network) to one robot all the marginal contributions $\{q_i : i \in \mathcal{V}\}$, and, then, by letting this robot to select the set $\mathcal{L}$, and to communicate it back to the rest of the robots.

Computation of control inputs of robots in $\mathcal{L}$ (step 3 of Algorithm 22) This steps requires zero rounds of communication among the robots, since each robot in the set $\mathcal{L}$, per Algorithm 22's step 2, merely adopts the controls it computed in Algorithm 22's step 1 (e.g., using the algorithm in [42]).

Computation of control inputs of robots in $\mathcal{V} \setminus \mathcal{L}$ (step 4 of Algorithm 22) This step is executed in ρ rounds per the statement of Theorem 24.

In sum, Algorithm 22 requires $2|\mathcal{V}| + \rho$ rounds of communication among the robots in $\mathcal{V}$ to terminate. ■

11.7.3. Proof of Proposition 12

We first prove Proposition 12's part 1 (approx. bounds), and then, Proposition 12's part 2 (communication rounds).

Proof of Proposition 12's part 1 (approximation bounds)

The proof follows the steps of the proof of Theorem 24; hence, we describe here only the steps where the proof differs.

We first prove ineq. (11.13); then, we prove ineq. (11.14).

Proof of ineq. (11.13) Consider that the objective function J is non-decreasing and submodular in the active robot set, such that (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$. Since, per Proposition 12, Algorithm 22 calls the coordinate descent algorithm in step 4, the equivalence in eq. (11.55) is now invalid, and, in particular, using Lemma 49, the following inequality holds instead:

$$J(\mathcal{V} \setminus \mathcal{L}) \geq \frac{1}{2} \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1, 2 \ldots, T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \mathcal{L})]. \tag{11.75}$$

Using ineq. (11.75), and following the same steps as in eqs. (11.55)-(11.58), we conclude:

$$J(\mathcal{V} \setminus \mathcal{L}) \geq \frac{1}{2} J^{\star}. \tag{11.76}$$

Using ineq. (11.76) the same way that ineq. (11.54) was used in the proof of Theorem 24's part 1, ineq. (11.14) is proved.

Proof of ineq. (11.14) Consider that the objective function J is non-decreasing in the active robot set, such that (without loss of generality) J is non-negative and $J[u_{1:T}(\emptyset)] = 0$. Similarly with the observations we made in the proof of ineq. (11.13), since, per Proposition 12, Algorithm 22 calls the coordinate descent algorithm in step 4, the equivalence in eq. (11.55) is now invalid, and, in particular, using Lemma 49, the following inequality holds

instead:

$$J(\mathcal{V} \setminus \mathcal{L}) \geq \frac{1 - c_J}{2} \max_{\substack{\bar{u}_{i,t} \in \mathcal{U}_{i,t}, i \in \mathcal{V}, \\ t = 1, 2 \ldots, T}} J[\bar{u}_{1:T}(\mathcal{V} \setminus \mathcal{L})]. \tag{11.77}$$

Using ineq. (11.77), and following the same steps as in eqs. (11.55)-(11.58), we conclude:

$$J(\mathcal{V} \setminus \mathcal{L}) \geq \frac{1 - c_J}{2} J^{\star}. \tag{11.78}$$

Using ineq. (11.78) the same way that ineq. (11.54) was used in the proof of Theorem 24's part 1, ineq. (11.14) is proved. ■

Proof of Proposition 12's part 2 (communication rounds)

The description of the generalized coordinate descent in Appendix 11.7.1 implies that the generalized coordinate descent terminates in at most $|\mathcal{V}|$ rounds, since each robot in $\mathcal{V}$ needs to communicate with at most one robot in $\mathcal{V}$ and at most once. Therefore, per the notation in Theorem 24, for the generalized coordinate descent it is $\rho = |\mathcal{V}|$. Overall, per Theorem 24's part 2, when Algorithm 22 calls generalized coordinate descent in step 4, it requires $2|\mathcal{V}| + \rho = 3|\mathcal{V}|$ rounds of communication among the robots in $\mathcal{V}$ to terminate. ■

BIBLIOGRAPHY

[1] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "A submodular optimization approach to controlled islanding under cascading failure," in *ACM/IEEE 8th International Conference on Cyber-Physical Systems*, 2017, pp. 187–196.

[2] V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1279–1291, 2012.

[3] C. Nieto-Granda, J. G. Rogers III, and H. Christensen, "Multi-robot exploration strategies for tactical tasks in urban environments," in *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics, 2013, pp. 87 410B–87 410B.

[4] T. Summers, "Actuator placement in networks using optimal control performance metrics," in *IEEE 55th Conference on Decision and Control*, 2016, pp. 2703–2708.

[5] S. T. Jawaid and S. L. Smith, "Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems," *Automatica*, vol. 61, pp. 282–288, 2015.

[6] P. Tokekar, V. Isler, and A. Franchi, "Multi-target visual tracking with aerial robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 3067–3072.

[7] A. Olshevsky, "Minimal controllability problems," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 3, pp. 249–258, 2014.

[8] A. Clark, L. Bushnell, and R. Poovendran, "On leader selection for performance and controllability in multi-agent systems," in *IEEE 51st Annual Conference on Decision and Control (CDC)*, Dec 2012, pp. 86–93.

[9] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "Toward synchronization in networks with nonlinear dynamics: A submodular optimization framework," *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 5055–5068, 2017.

[10] H. Zhang, R. Ayoub, and S. Sundaram, "Sensor selection for kalman filtering of linear dynamical systems: Complexity, limitations and greedy algorithms," *Automatica*, vol. 78, pp. 202 – 210, 2017.

[11] R. K. Williams, A. Gasparri, and G. Ulivi, "Decentralized matroid optimization for topology constraints in multi-robot allocation problems," in *IEEE Conference on Robotics and Automation*, 2017, pp. 293–300.

[12] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions – II," in *Polyhedral combinatorics*, 1978, pp. 73–87.

[13] U. Feige, "A threshold of $ln(n)$ for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[14] R. Iyer, S. Jegelka, and J. Bilmes, "Fast semidifferential-based submodular function optimization," in *International Conference on International Conference on Machine Learning*, 2013, pp. 855–863.

[15] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," *Math. of Operations Research*, vol. 42, no. 4, pp. 1197–1218, 2017.

[16] A. Krause and D. Golovin, "Submodular function maximization," *Tractability: Practical Approaches to Hard Problems*, vol. 3, p. 19, 2012.

[17] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a monotone submodular function subject to a matroid constraint," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1740–1766, 2011.

[18] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 40–52, 2014.

[19] P. Muller and H. Weber, "Analysis and optimization of certain qualities of controllability and observability for linear dynamical systems," *Automatica*, vol. 8, no. 3, pp. 237 – 246, 1972.

[20] 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, 2006.

[21] V. Tzoumas, Y. Xue, S. Pequito, P. Bogdan, and G. J. Pappas, "Selecting sensors in biological fractional-order systems," *IEEE Transactions on Control of Network Systems*, 2018, in press.

[22] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[23] A. S. Willsky, "A survey of design methods for failure detection in dynamic systems," *Automatica*, vol. 12, no. 6, pp. 601–611, 1976.

[24] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks," *Computer*, vol. 35, no. 10, pp. 54–62, 2002.

[25] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "Submodular optimization for voltage control," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 502–513, 2018.

[26] R. B. Myerson, *Game theory*. Harvard University Press, 2013.

[27] Z. Liu, A. Clark, L. Bushnell, D. Kirschen, and R. Poovendran, "Controlled islanding via weak submodularity," *arXiv preprint arXiv:1803.05042*, 2018.

[28] G. Yan, G. Tsekenis, B. Barzel, J.-J. Slotine, Y.-Y. Liu, and A.-L. Barabási, "Spectrum of controlling and observing complex networks," *Nature Physics*, vol. 11, no. 9, p. 779, 2015.

[29] V. Tzoumas, L. Carlone, G. J. Pappas, and A. Jadbabaie, "Sensing-Constrained LQG Control," *arXiv e-prints: 1709.08826*, 2017.

[30] A. Schrijver, *Combinatorial optimization: Polyhedra and efficiency*. Springer Science & Business Media, 2003, vol. 24.

[31] R. K. Iyer, S. Jegelka, and J. A. Bilmes, "Curvature and optimal algorithms for learning and minimizing submodular functions," in *Advances in Neural Inform. Processing Systems*, 2013, pp. 2742–2750.

[32] A. A. Bian, J. M. Buhmann, A. Krause, and S. Tschiatschek, "Guarantees for greedy maximization of non-submodular functions with applications," in *Int. Conf. on Machine Learning*, 2017, pp. 498–507.

[33] M. Conforti and G. Cornuéjols, "Submodular set functions, matroids and the greedy algorithm," *Discrete Applied Mathematics*, vol. 7, no. 3, pp. 251 – 274, 1984.

[34] J. B. Orlin, A. S. Schulz, and R. Udwani, "Robust monotone submodular function maximization," *arXiv preprint:1507.06616*, 2015.

[35] I. Bogunovic, J. Zhao, and V. Cevher, "Robust maximization of non-submodular objectives," *ArXiv e-prints:1802.07073*, 2018.

[36] B. Mirzasoleiman, A. Karbasi, and A. Krause, "Deletion-robust submodular maximization: Data summarization with 'the right to be forgotten'," in *International Conference on Machine Learning*, 2017, pp. 2449–2458.

[37] E. Kazemi, M. Zadimoghaddam, and A. Karbasi, "Deletion-robust submodular maximization at scale," *ArXiv e-prints:1711.07112*, 2017.

[38] K. Poularakis, G. Iosifidis, G. Smaragdakis, and L. Tassiulas, "One step at a time: Optimizing SDN upgrades in ISP networks," in *IEEE Conference on Computer Communications*, 2017, pp. 1–9.

[39] D. Golovin and A. Krause, "Adaptive submodularity: A new approach to active learning and stochastic optimization," in *Annual Conference on Learning Theory*, 2010, pp. 333–345.

[40] G. L. Nemhauser and L. A. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Mathematics of operations research*, vol. 3, no. 3, pp. 177–188, 1978.

[41] S. Mitrovic, I. Bogunovic, A. Norouzi-Fard, J. M. Tarnawski, and V. Cevher, "Streaming robust submodular maximization," in *Advances in Neural Information Processing Systems*, 2017, pp. 4560–4569.

[42] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6447–6454.

[43] M. Michini, M. A. Hsieh, E. Forgoston, and I. B. Schwartz, "Robotic tracking of coherent structures in flows," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 593–603, 2014.

[44] A. Krause and C. Guestrin, "A note on the budgeted maximization of submodular functions," 2005.

[45] A. Jadbabaie, A. Olshevsky, G. J. Pappas, and V. Tzoumas, "Minimal reachability is hard to approximate," *arXiv preprint arXiv:1710.10244*, 2017.

[46] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, no. 7346, pp. 167–173, 2011.

[47] F. Muller and A. Schuppert, "Few inputs can reprogram biological networks," *Nature*, vol. 478, no. 7369, pp. E4–E4, 2011.

[48] T. Zhou, "Minimal inputs/outputs for a networked system," *IEEE Control Systems Letters*, vol. 1, no. 2, pp. 298–303, 2017.

[49] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "Minimizing convergence error in multi-agent systems via leader selection: A supermodular optimization approach," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1480–1494, 2014.

[50] A. Clark, L. Bushnell, and R. Poovendran, "A supermodular optimization framework for leader selection under link noise in linear multi-agent systems," *IEEE Trans. on Aut. Contr.*, vol. 59, no. 2, pp. 283–296, 2014.

[51] S. Pequito, S. Kar, and A. P. Aguiar, "A framework for structural input/output and control configuration selection in large-scale systems," *IEEE Trans. on Automatic Control*, vol. 61, no. 2, pp. 303–318, 2016.

[52] T. H. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2016.

[53] V. Tzoumas, M. A. Rahimian, G. J. Pappas, and A. Jadbabaie, "Minimal actuator placement with bounds on control effort," *IEEE Trans. on Control of Network Systems*, vol. 3, no. 1, pp. 67–78, 2016.

[54] Y. Zhao, F. Pasqualetti, and J. Cortés, "Scheduling of control nodes for improved network controllability," in *IEEE 55th Conference on Decision and Control*, 2016, pp. 1859–1864.

[55] S. Pequito, G. Ramos, S. Kar, A. Aguiar, and J. Ramos, "Robust minimal controllability problem," *Automatica*, vol. 82, pp. 261–268, 2017.

[56] V. Tzoumas, K. Gatsis, A. Jadbabaie, and G. J. Pappas, "Resilient monotone submodular function maximization," in *IEEE Conference on Decision and Control*, 2017, pp. 1362–1367.

[57] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Sensor placement for optimal kalman filtering," in *Proceedings of the American Control Conference*, 2016, pp. 191–196.

[58] ——, "Near-optimal sensor scheduling for batch state estimation," in *IEEE 55th Conference on Decision and Control*, 2016, pp. 2695–2702.

[59] H. Zhang, R. Ayoub, and S. Sundaram, "Sensor selection for kalman filtering of linear dynamical systems: Complexity, limitations and greedy algorithms," *Automatica*, vol. 78, pp. 202–210, 2017.

[60] L. Carlone and S. Karaman, "Attention and anticipation in fast visual-inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2017, pp. 3886–3893.

[61] M. Amin and J. Stringer, "The electric power grid: Today and tomorrow," *MRS bulletin*, vol. 33, no. 04, pp. 399–407, 2008.

[62] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "MinGen: Minimal generator set selection for small signal stability in power systems: A submodular framework," in *Proceedings of the IEEE 55th Conference on Decision and Control*, 2016, pp. 4122–4129.

[63] *California Partners for Advanced Transit and Highways*, 2006. [Online]. Available: http://www.path.berkeley.edu/

[64] S. Gu *et al.*, "Controllability of structural brain networks," *Nature communications*, vol. 6, 2015.

[65] C. Tu, R. P. Rocha, M. Corbetta, S. Zampieri, M. Zorzi, and S. Suweis, "Warnings and Caveats in Brain Controllability," *ArXiv e-prints*, 2017.

[66] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Minimal reachability problems," in *Proceedings of the IEEE 54th Annual Conference on Decision and Control*, 2015, pp. 4220–4225.

[67] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "Towards

scalable voltage control in smart grid: A submodular optimization approach," in *Proceedings of the 7th International Conference on Cyber-Physical Systems*, 2016, p. 20.

[68] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," in *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2014, pp. 1134–1148.

[69] S. Arora and B. Barak, *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[70] G. Nemhauser, L. Wolsey, and M. Fisher, "An analysis of approximations for maximizing submodular set functions – I," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[71] C.-T. Chen, *Linear System Theory and Design*, 3rd ed. New York, NY, USA: Oxford University Press, Inc., 1998.

[72] D. Foster, H. Karloff, and J. Thaler, "Variable selection is hard," in *Proceedings of the Conference on Learning Theory*, 2015, pp. 696–709.

[73] V. Tzoumas, M. Rahimian, G. Pappas, and A. Jadbabaie, "Minimal actuator placement with bounds on control effort," *arXiv preprint:1409.3289*, 2014.

[74] M. Newman, A.-L. Barabási, and D. Watts, *The structure and dynamics of networks*. Princeton University Press, 2006.

[75] A. M. Hermundstad, D. S. Bassett, K. S. Brown, E. M. Aminoff, D. Clewett, S. Freeman, A. Frithsen, A. Johnson, C. M. Tipper, M. B. Miller *et al.*, "Structural foundations of resting-state and task-based functional connectivity in the human brain," *Proceedings of the National Academy of Sciences*, vol. 110, no. 15, pp. 6169–6174, 2013.

[76] Y. Katz, K. Tunstrøm, C. C. Ioannou, C. Huepe, and I. D. Couzin, "Inferring the structure and dynamics of interactions in schooling fish," *Proceedings of the National Academy of Sciences*, vol. 108, no. 46, pp. 18 720–18 725, 2011.

[77] F. Jordán and I. Scheuring, "Network ecology: topological constraints on ecosystem dynamics," *Physics of Life Reviews*, vol. 1, no. 3, pp. 139–172, 2004.

[78] G. Orosz, J. Moehlis, and R. M. Murray, "Controlling biological networks by time-delayed signals," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1911, pp. 439–454, 2010.

[79] I. Rajapakse, M. Groudine, and M. Mesbahi, "What can systems theory of networks offer to biology?" *PLoS computational biology*, vol. 8, no. 6, p. e1002543, 2012.

[80] A. Khanafer and T. Basar, "Information spread in networks: Control, games, and

equilibria," in *Information Theory and Applications Workshop (ITA), 2014*, 2014, pp. 1–10.

[81] L. Chen, N. Li, L. Jiang, and S. H. Low, "Optimal demand response: problem formulation and deterministic case," in *Control and Optimization Theory for Electric Smart Grids*, A. Chakrabortty and M. Ilic, Eds. Springer, 2012.

[82] A. Olshevsky, "Minimal controllability problems," *IEEE Transactions on Control of Network Systems*, 2014, in press.

[83] G. Ramos, S. Pequito, S. Kar, A. P. Aguiar, and J. Ramos, "On the np-completeness of the minimal controllability problem," *arXiv preprint arXiv:1401.4209*, 2014.

[84] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 40–52, March 2014.

[85] C.-T. Chen, *Linear System Theory and Design*, 3rd ed. New York, NY, USA: Oxford University Press, Inc., 1998.

[86] G. Yan, J. Ren, Y.-C. Lai, C.-H. Lai, and B. Li, "Controlling complex networks: How much energy is needed?" *Phys. Rev. Lett.*, vol. 108, p. 218703, May 2012.

[87] J. Sun and A. E. Motter, "Controllability transition and nonlocality in network control," *Phys. Rev. Lett.*, vol. 110, p. 208701, May 2013.

[88] F. Cortesi, T. Summers, and J. Lygeros, "Submodularity of energy related controllability metrics," in *IEEE 53rd Annual Conference on Decision and Control*, 2014, pp. 2883–2888.

[89] S. Jafari, A. Ajorlou, and A. G. Aghdam, "Leader localization in multi-agent systems subject to failure: A graph-theoretic approach," *Automatica*, vol. 47, no. 8, pp. 1744–1750, 2011.

[90] C. Commault and J.-M. Dion, "Input addition and leader selection for the controllability of graph-based systems," *Automatica*, vol. 49, no. 11, pp. 3322 – 3328, 2013.

[91] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "Minimizing convergence error in multi-agent systems via leader selection: A supermodular optimization approach," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1480–1494, June 2014.

[92] A. Clark, L. Bushnell, and R. Poovendran, "A supermodular optimization framework for leader selection under link noise in linear multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 283–296, Feb 2014.

[93] D. S. Bernstein, *Matrix mathematics: theory, facts, and formulas.* Princeton University Press, 2009.

[94] L. A. Wolsey, "An analysis of the greedy algorithm for the submodular set covering problem," *Combinatorica*, vol. 2, no. 4, pp. 385–393, 1982.

[95] K. B. Petersen and M. S. Pedersen, *The matrix cookbook*, 2012.

[96] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*. New York, NY, USA: Wiley-Interscience, 1988.

[97] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, 1987, pp. 1–6.

[98] A. Reusken, "Approximation of the determinant of large sparse symmetric positive definite matrices," *SIAM J. Matrix Anal. Appl.*, vol. 23, no. 3, pp. 799–818, Mar. 2001.

[99] M. Minoux, "Accelerated greedy algorithms for maximizing submodular set functions," in *Optimization Techniques*. Springer, 1978, pp. 234–243.

[100] P. Benner, J.-R. Li, and T. Penzl, "Numerical solution of large-scale lyapunov equations, riccati equations, and linear-quadratic optimal control problems," *Numerical Linear Algebra with Applications*, vol. 15, no. 9, pp. 755–777, 2008.

[101] S. Arora and B. Barak, *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[102] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Sensor placement for optimal Kalman filtering," in *Amer. Contr. Conf.*, 2016, pp. 191–196.

[103] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear estimation*. Prentice Hall, 2000.

[104] F. Lin, M. Fardad, and M. R. Jovanovic, "Design of optimal sparse feedback gains via the alternating direction method of multipliers," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2426–2431, 2013.

[105] S. Pequito, G. Ramos, S. Kar, A. P. Aguiar, and J. Ramos, "On the Exact Solution of the Minimal Controllability Problem," *arXiv preprint arXiv: 1401.4209*, 2014.

[106] T. H. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2016.

[107] V. Tzoumas, M. A. Rahimian, G. J. Pappas, and A. Jadbabaie, "Minimal actuator placement with bounds on control effort," *IEEE Transactions on Control of Network Systems*, 2015, in press.

[108] N. Matni and V. Chandrasekaran, "Regularization for design," in *IEEE 53rd Annual Conference on Decision and Control (CDC)*, 2014, pp. 1111–1118.

[109] S. Pequito, S. Kar, and A. Aguiar, "A framework for structural input/output and control configuration selection in large-scale systems," *IEEE Trans. on Automatic Control*, vol. 61, no. 2, pp. 303–318, 2016.

[110] V. Tzoumas, M. A. Rahimian, G. J. Pappas, and A. Jadbabaie, "Minimal actuator placement with optimal control constraints," in *Proceedings of the American Control Conference*, July 2015, pp. 2081 – 2086.

[111] G. Yan, G. Tsekenis, B. Barzel, J.-J. Slotine, Y.-Y. Liu, and A.-L. Barabási, "Spectrum of controlling and observing complex networks," *Nature Physics*, no. 11, pp. 779–786, 2015.

[112] Y. Zhao and J. Cortés, "Gramian-based reachability metrics for bilinear networks," *arXiv preprint arXiv:1509.02877*, 2015.

[113] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[114] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *J.l of Machine Learning Research*, vol. 9, pp. 235–284, 2008.

[115] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: Leveraging submodularity," in *49th IEEE Conference on Decision and Control (CDC),*, 2010, pp. 2572–2577.

[116] A. Krause and C. Guestrin, *Beyond convexity: Submodularity in machine learning*, 2008, iCML Tutorials.

[117] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Minimal reachability problems," in *54th IEEE Conference on Decision and Control (CDC)*, December 2015, to appear.

[118] C. Jiang, Y. Chai Soh, and H. Li, "Sensor placement by maximal projection on minimum eigenspace for linear inverse problem," *arXiv preprint arXiv: 1506.00747*, 2015.

[119] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Transactions on Signal Processing*, vol. 57, no. 2, pp. 451–462, 2009.

[120] N. K. Dhingra, M. R. Jovanovic, and Z.-Q. Luo, "An admm algorithm for optimal sensor and actuator selection," in *IEEE 53rd Annual Conference on Decision and Control (CDC)*, 2014, pp. 4039–4044.

[121] U. Munz, M. Pfister, and P. Wolfrum, "Sensor and actuator placement for linear systems based on h_2 and h_∞ optimization," *IEEE Transactions on Automatic Control*, vol. 59, no. 11, pp. 2984–2989, Nov 2014.

[122] M.-A. Belabbas, "Geometric methods for optimal sensor design," *arXiv preprint arXiv: 1503.05968*, 2015.

[123] D. P. Bertsekas, *Dynamic programming and optimal control, Vol. I.* Athena Scientific, 2005.

[124] S. Venkatesh, *The Theory of Probability: Explorations and Applications.* Cambridge University Press, 2012.

[125] Y. Fang, K. Loparo, X. Feng *et al.*, "Inequalities for the trace of matrix product," *IEEE Transactions on Automatic Control,*, vol. 39, no. 12, pp. 2489–2490, 1994.

[126] V. Kumar, D. Rus, and S. Singh, "Robot and sensor networks for first responders," *IEEE Pervasive computing*, vol. 3, no. 4, pp. 24–33, 2004.

[127] R. Nowak, U. Mitra, and R. Willett, "Estimating inhomogeneous fields using wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 6, pp. 999–1006, 2004.

[128] M. P. Vitus, "Sensor placement for improved robotic navigation," *Robotics: Science and Systems VI*, p. 217, 2011.

[129] E. Masazade, M. Fardad, and P. K. Varshney, "Sparsity-promoting extended kalman filtering for target tracking in wireless sensor networks," *IEEE Signal Processing Letters*, vol. 19, no. 12, pp. 845–848, 2012.

[130] H. Rowaihy, S. Eswaran, M. Johnson, D. Verma, A. Bar-Noy, T. Brown, and T. La Porta, "A survey of sensor selection schemes in wireless sensor networks," in *Defense and Security Symposium.* International Society for Optics and Photonics, 2007, pp. 65 621A–65 621A.

[131] A. O. Hero III and D. Cochran, "Sensor management: Past, present, and future," *IEEE Sensors Journal*, vol. 11, no. 12, pp. 3064–3075, 2011.

[132] S. Liu, M. Fardad, E. Masazade, and P. K. Varshney, "Optimal periodic sensor scheduling in networks of dynamical systems," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3055–3068.

[133] S. Anderson, T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression," *Autonomous Robots*, vol. 39, no. 3, pp. 221–238, 2015.

[134] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular slam: Why filter?" in *2010 IEEE International Conference on Robotics and Automation (ICRA),*. IEEE, 2010, pp. 2657–2664.

[135] V. Roy, A. Simonetto, and G. Leus, "Spatio-temporal sensor management for environmental field estimation," *Signal Processing*, vol. 128, pp. 369 – 381, 2016.

[136] J. Liu, D. Petrovic, and F. Zhao, "Multi-step information-directed sensor querying in distributed sensor networks," in *Acoustics, Speech, and Signal Processing, 2003.*

Proceedings.(ICASSP'03). 2003 IEEE International Conference on, vol. 5. IEEE, 2003, pp. V–145.

[137] M. P. Vitus, W. Zhang, A. Abate, J. Hu, and C. J. Tomlin, "On efficient sensor scheduling for linear dynamical systems," *Automatica*, vol. 48, no. 10, pp. 2482–2493, 2012.

[138] Y. Mo, R. Ambrosino, and B. Sinopoli, "Sensor selection strategies for state estimation in energy constrained wireless sensor networks," *Automatica*, vol. 47, no. 7, pp. 1330–1338, 2011.

[139] H. Zhang, R. Ayoub, and S. Sundaram, "Sensor selection for optimal filtering of linear dynamical systems: Complexity and approximation," in *IEEE Conference on Decision and Control (CDC)*, 2015.

[140] M. F. Huber, A. Kuwertz, F. Sawo, and U. D. Hanebeck, "Distributed greedy sensor scheduling for model-based reconstruction of space-time continuous physical phenomena," in *Information Fusion, 2009. FUSION'09. 12th International Conference on.* IEEE, 2009, pp. 102–109.

[141] J. Vondrák, "Submodularity and curvature: The optimal algorithm," *RIMS Kokyuroku Bessatsu*, vol. 23, pp. 253–266, 2010.

[142] L. G. Molinari, "Determinants of block tridiagonal matrices," *Linear algebra and its applications*, vol. 429, no. 8, pp. 2221–2226, 2008.

[143] A. Nemirovski, "Interior point polynomial time methods in convex programming."

[144] J. E. Weimer, B. Sinopoli, and B. H. Krogh, "A relaxation approach to dynamic sensor selection in large-scale wireless networks," in *28th International Conference on Distributed Computing Systems Workshops (ICDCS)*, 2008, pp. 501–506.

[145] A. S. Leong, S. Dey, and J. S. Evans, "Asymptotics and power allocation for state estimation over fading channels," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 1, pp. 611–633, January 2011.

[146] C. Moler and C. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM review*, vol. 45, no. 1, pp. 3–49, 2003.

[147] T. McMillen, T. Williams, and P. Holmes, "Nonlinear muscles, passive viscoelasticity and body taper conspire to create neuromechanical phase lags in anguilliform swimmers," *PLoS computational biology*, vol. 4, no. 8, p. e1000157, 2008.

[148] Y. Kobayashi, H. Watanabe, T. Hoshi, K. Kawamura, and M. G. Fujie, "Viscoelastic and nonlinear liver modeling for needle insertion simulation," in *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery.* Springer, 2012, pp. 41–67.

[149] C. Wex, M. Fröhlich, K. Brandstädter, C. Bruns, and A. Stoll, "Experimental analysis of the mechanical behavior of the viscoelastic porcine pancreas and preliminary case study on the human pancreas," *Journal of the mechanical behavior of biomedical materials*, vol. 41, pp. 199–207, 2015.

[150] K. Wang, R. McCarter, J. Wright, J. Beverly, and R. Ramirez-Mitchell, "Viscoelasticity of the sarcomere matrix of skeletal muscles. the titin-myosin composite filament is a dual-stage molecular spring," *Biophysical Journal*, vol. 64, no. 4, pp. 1161–1177, 1993.

[151] T. M. Best, J. McElhaney, W. E. Garrett, and B. S. Myers, "Characterization of the passive responses of live skeletal muscle using the quasi-linear theory of viscoelasticity," *Journal of biomechanics*, vol. 27, no. 4, pp. 413–419, 1994.

[152] T. C. Doehring, A. D. Freed, E. O. Carew, and I. Vesely, "Fractional order viscoelasticity of the aortic valve cusp: an alternative to quasilinear viscoelasticity," *Journal of biomechanical engineering*, vol. 127, no. 4, pp. 700–708, 2005.

[153] E. Macé, I. Cohen, G. Montaldo, R. Miles, M. Fink, and M. Tanter, "In vivo mapping of brain elasticity in small animals using shear wave imaging," *IEEE transactions on medical imaging*, vol. 30, no. 3, pp. 550–558, 2011.

[154] S. Nicolle, L. Noguer, and J.-F. Palierne, "Shear mechanical properties of the spleen: experiment and analytical modelling," *Journal of the mechanical behavior of biomedical materials*, vol. 9, pp. 130–136, 2012.

[155] N. Grahovac and M. Žigić, "Modelling of the hamstring muscle group by use of fractional derivatives," *Computers & Mathematics with Applications*, vol. 59, no. 5, pp. 1695–1700, 2010.

[156] V. E. Tarasov, *Fractional dynamics: applications of fractional calculus to dynamics of particles, fields and media*. Springer Science & Business Media, 2011.

[157] P. Bogdan, B. M. Deasy, B. Gharaibeh, T. Roehrs, and R. Marculescu, "Heterogeneous structure of stem cells dynamics: statistical models and quantitative predictions," *Scientific reports*, vol. 4, 2014.

[158] M. Ghorbani and P. Bogdan, "A cyber-physical system approach to artificial pancreas design," in *Proceedings of the ninth IEEE/ACM/IFIP international conference on hardware/software codesign and system synthesis*. IEEE Press, 2013, p. 17.

[159] Y. Xue, S. Rodriguez, and P. Bogdan, "A spatio-temporal fractal model for a cps approach to brain-machine-body interfaces," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2016*. IEEE, 2016, pp. 642–647.

[160] Y. Xue, S. Pequito, J. R. Coelho, P. Bogdan, and G. J. Pappas, "Minimum number of sensors to ensure observability of physiological systems: A case study," in *Commu-*

nication, Control, and Computing (Allerton), 2016 54th Annual Allerton Conference on. IEEE, 2016, pp. 1181–1188.

[161] Y. Xue and P. Bogdan, "Constructing compact causal mathematical models for complex dynamics," in *Proceedings of the 8th International Conference on Cyber-Physical Systems*. ACM, 2017, pp. 97–107.

[162] C. Song, S. Havlin, and H. A. Makse, "Self-similarity of complex networks," *arXiv preprint cond-mat/0503078*, 2005.

[163] K.-I. Goh, G. Salvi, B. Kahng, and D. Kim, "Skeleton and fractal scaling in complex networks," *Physical review letters*, vol. 96, no. 1, p. 018701, 2006.

[164] F. Klimm, D. S. Bassett, J. M. Carlson, and P. J. Mucha, "Resolving structural variability in network models and the brain," *PLoS computational biology*, vol. 10, no. 3, p. e1003491, 2014.

[165] Y. Xue and P. Bogdan, "Reliable multi-fractal characterization of weighted complex networks: Algorithms and implications," *Scientific Reports*, vol. 7, 2017.

[166] J. Klafter, S. Lim, and R. Metzler, *Fractional dynamics: recent advances*. World Scientific, 2012.

[167] S. Guermah, S. Djennoune, and M. Bettayeb, "Controllability and observability of linear discrete-time fractional-order systems." *Applied Mathematics and Computer Science*, vol. 18, no. 2, pp. 213–222, 2008.

[168] Y. Zhao and J. Cortes, "Gramian-based reachability metrics for bilinear networks," *IEEE Transactions on Control of Network Systems*, vol. PP, no. 99, pp. 1–1, 2016.

[169] S. Pequito, N. Popli, S. Kar, M. Ilic, and A. Aguiar, "A framework for actuator placement in large scale power systems: Minimal strong structural controllability," in *IEEE 5th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2013, pp. 416–419.

[170] Y. Xue, S. Pequito, J. M. Coelho, P. Bogdan, and G. J. Pappas, "Minimum number of sensors to ensure observability of physiological systems: A case study," in *54th Annual Allerton Conference on Communication, Control, and Computing*, 2016.

[171] S. Pequito, P. Bogdan, and G. J. Pappas, "Minimum number of probes for brain dynamics observability," in *54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 306–311.

[172] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Sensor placement for optimal kalman filtering: Fundamental limits, submodularity, and algorithms," in *American Control Conference (ACC)*, 2016, pp. 191–196.

[173] M.-A. Belabbas, "Geometric methods for optimal sensor design," in *Proc. R. Soc. A*, vol. 472, no. 2185. The Royal Society, 2016, p. 20150312.

[174] S. Guermah, S. Djennoune, and M. Bettayeb, "Controllability and observability of linear discrete-time fractional-order systems," *Int. J. Appl. Math. Comput. Sci.*, vol. 18, no. 2, pp. 213–222, Jun. 2008.

[175] V. Tzoumas, N. A. Atanasov, A. Jadbabaie, and G. J. Pappas, "Scheduling nonlinear sensors for stochastic process estimation," in *American Control Conference*, 2017, pp. 580–585.

[176] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: A general-purpose brain-computer interface (BCI) system," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, 2004.

[177] "EEG motor movement/imagery dataset," *PhysioNet*. [Online]. Available: http://www.physionet.org/pn4/eegmmidb/

[178] Y. Xue, S. Rodriguez, and P. Bogdan, "A spatio-temporal fractal model for a CPS approach to brain-machine-body interfaces," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2016, pp. 642–647.

[179] "Emotiv EPOC website." [Online]. Available: https://emotiv.com/epoc.php

[180] C.-W. Ko, J. Lee, and M. Queyranne, "An exact algorithm for maximum entropy sampling," *Operations Research*, vol. 43, no. 4, pp. 684–691, 1995.

[181] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[182] V. Tzoumas, N. A. Atanasov, A. Jadbabaie, and G. J. Pappas, "Scheduling nonlinear sensors for stochastic process estimation," in *American Control Conference*, 2017, pp. 580–585.

[183] N. Karnad, *Robot Motion Planning for Tracking and Capturing Adversarial, Cooperative and Independent Targets*. U. of Minnesota, 2015.

[184] S. Karlin, *A first course in stochastic processes*. Academic press, 2014.

[185] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Near-Optimal Sensor Scheduling for Batch State Estimation: Complexity, Algorithms, and Limits," in *55th IEEE Conference on Decision and Control (CDC)*, 2016.

[186] J. L. Ny, E. Feron, and M. A. Dahleh, "Scheduling continuous-time kalman filters," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1381–1394, 2011.

[187] X. Shen, S. Liu, and P. K. Varshney, "Sensor selection for nonlinear systems in large

sensor networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 2664–2678, 2014.

[188] Y. He and E. K. P. Chong, "Sensor scheduling for target tracking: A monte carlo sampling approach," *Digital Signal Processing*, vol. 16, no. 5, pp. 533–545, 2006.

[189] R. Durrett, *Probability: theory and examples*. Cambridge University Press, 2010.

[190] J. Vondrák, "Optimal approximation for the submodular welfare problem in the value oracle model," in *Proceedings of the fortieth annual ACM symposium on Theory of computing*. ACM, 2008, pp. 67–74.

[191] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[192] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical mathematics*. Springer Science & Business Media, 2010, vol. 37.

[193] V. Tzoumas, L. Carlone, G. J. Pappas, and A. Jadbabaie, "Control and Sensing Co-design," *ArXiv e-prints: 1802.08376*, 2018.

[194] N. Elia and S. Mitter, "Stabilization of linear systems with limited information," *IEEE Trans. on Automatic Control*, vol. 46, no. 9, pp. 1384–1400, 2001.

[195] G. Nair and R. Evans, "Stabilizability of stochastic linear systems with finite feedback data rates," *SIAM Journal on Control and Optimization*, vol. 43, no. 2, pp. 413–436, 2004.

[196] S. Tatikonda and S. Mitter, "Control under communication constraints," *IEEE Trans. on Automatic Control*, vol. 49, no. 7, pp. 1056–1068, 2004.

[197] V. Borkar and S. Mitter, "LQG control with communication constraints," *Comm., Comp., Control, and Signal Processing*, pp. 365–373, 1997.

[198] J. L. Ny and G. Pappas, "Differentially private filtering," *IEEE Trans. on Automatic Control*, vol. 59, no. 2, pp. 341–354, 2014.

[199] E. Shafieepoorfard and M. Raginsky, "Rational inattention in scalar LQG control," in *Proceedings of the 52th IEEE Conference in Decision and Control*, 2013, pp. 5733–5739.

[200] J. L. Ny, E. Feron, and M. A. Dahleh, "Scheduling continuous-time kalman filters," *IEEE Trans. on Aut. Control*, vol. 56, no. 6, pp. 1381–1394, 2011.

[201] E. Nozari, F. Pasqualetti, and J. Cortés, "Time-invariant versus time-varying actuator scheduling in complex networks," in *American Control Conference*, 2017, pp. 4995–5000.

[202] T. Summers and J. Ruths, "Performance bounds for optimal feedback control in networks," *arXiv e-prints:1707.04528*, 2017.

[203] T. Summers and M. Kamgarpour, "Performance guarantees for greedy maximization of non-submodular set functions in systems and control," *arXiv e-prints:1712.04122*, 2017.

[204] A. F. Taha, N. Gatsis, T. Summers, and S. Nugroho, "Time-varying sensor and actuator selection for uncertain cyber-physical systems," *ArXiv e-prints:1708.07912*, 2017.

[205] G. Nair, F. Fagnani, S. Zampieri, and R. Evans, "Feedback control under data rate constraints: An overview," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 108–137, 2007.

[206] J. Hespanha, P. Naghshtabrizi, and Y.Xu, "A survey of results in net- worked control systems," *Pr. of the IEEE*, vol. 95, no. 1, p. 138, 2007.

[207] J. Baillieul and P. Antsaklis, "Control and communication challenges in networked real-time systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 9–28, 2007.

[208] T. Tanaka and H. Sandberg, "SDP-based joint sensor and controller design for information-regularized optimal LQG control," in *IEEE 54th Annual Conference on Decision and Control*, 2015, pp. 4486–4491.

[209] T. Tanaka, P. M. Esfahani, and S. K. Mitter, "LQG control with minimum directed information: Semidefinite programming approach," *IEEE Trans. on Automatic Control*, vol. 63, no. 1, pp. 37–52, 2018.

[210] L. A. Wolsey, "An analysis of the greedy algorithm for the submodular set covering problem," *Combinatorica*, vol. 2, no. 4, pp. 385–393, 1982.

[211] S. Khuller, A. Moss, and J. S. Naor, "The budgeted maximum coverage problem," *Info. Processing Letters*, vol. 70, no. 1, pp. 39–45, 1999.

[212] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Operations Research Letters*, vol. 32, no. 1, pp. 41–43, 2004.

[213] V. Tzoumas, L. Carlone, G. J. Pappas, and A. Jadbabaie, "Sensing-Constrained LQG Control," *arXiv e-prints: 1709.08826*, 2017.

[214] U. Feige, "A threshold of $ln(n)$ for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[215] D. S. Bernstein, *Matrix mathematics*. Princeton University Press, 2005.

[216] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," *J. of Symbolic Comp.*, vol. 9, no. 3, pp. 251–280, 1990.

[217] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Resilient Non-Submodular Maximization over Matroid Constraints," *ArXiv e-prints: 1804.01013*.

[218] A. Clark, L. Bushnell, and R. Poovendran, "On leader selection for performance and controllability in multi-agent systems," in *IEEE 51st Annual Conference on Decision and Control*, 2012, pp. 86–93.

[219] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a submodular set function subject to a matroid constraint," in *International Conference on Integer Programming and Combinatorial Optimization*, 2007, pp. 182–196.

[220] L. Carlone and S. Karaman, "Attention and anticipation in fast visual-inertial navigation," in *Int. C. on Rob. and Autom.*, 2017, pp. 3886–3893.

[221] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[222] C. Boutsidis, M. W. Mahoney, and P. Drineas, "An improved approximation algorithm for the column subset selection problem," in *ACM-SIAM Symposium on Discrete algorithms*, 2009, pp. 968–977.

[223] E. R. Elenberg, R. Khanna, A. G. Dimakis, and S. Negahban, "Restricted strong convexity implies weak submodularity," *ArXiv e-prints:1612.00804*, 2016.

[224] V. Cevher and A. Krause, "Greedy dictionary selection for sparse representation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 979–988, 2011.

[225] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection," in *International Conference on Machine Learning*, 2011, pp. 1057–1064.

[226] R. Khanna, E. Elenberg, A. Dimakis, S. Negahban, and J. Ghosh, "Scalable greedy feature selection via weak submodularity," in *Artificial Intelligence and Statistics*, 2017, pp. 1560–1568.

[227] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," in *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2015, pp. 1134–1148.

[228] D. Sharma, A. Kapoor, and A. Deshpande, "On greedy maximization of entropy," in *Inter. Conf. on Machine Learning*, 2015, pp. 1330–1338.

[229] B. Schlotfeldt, V. Tzoumas, D. Thakur, and G. J. Pappas, "Resilient Active Information Gathering with Mobile Robots," *ArXiv e-prints: 1803.09730*, 2018.

[230] C. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[231] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Resilient Monotone Sequential Maximization," *ArXiv e-prints: 1803.07954*, 2018.

[232] D. Golovin and A. Krause, "Adaptive submodularity: Theory and applications in active learning and stochastic optimization," *Journal of Artificial Intelligence Research*, vol. 42, pp. 427–486, 2011.

[233] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "Scalable and distributed submodular maximization with matroid constraints," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, 2015, pp. 435–442.

[234] B. Schlotfeldt, V. Tzoumas, D. Thakur, and G. J. Pappas, "Resilient Active Information Gathering with Mobile Robots," *ArXiv e-prints: 1803.09730*.

[235] S. Karaman and E. Frazzoli, "High-speed flight in an ergodic forest," in *IEEE Intern. Confer. on Robotics and Automation*, 2012, pp. 2899–2906.

[236] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[237] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *IEEE/RSJ Int. Conf. on Intel. Robots and Systems*, 2017, pp. 2135–2142.

[238] M. Santos, Y. Diaz-Mercado, and M. Egerstedt, "Coverage control for multirobot teams with heterogeneous sensing capabilities," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 919–925, 2018.

[239] A. Krause, "Optimizing sensing: Theory and applications," Ph.D. dissertation, Carnegie Mellon University, 2008.

[240] J. L. Williams, "Information theoretic sensor management," Ph.D. dissertation, Massachusetts Institute of Technology, 2007.

[241] G. M. Hoffmann and C. J. Tomlin, "Mobile sensor network control using mutual information methods and particle filters," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 32–47, 2010.

[242] B. J. Julian, M. Angermann, M. Schwager, and D. Rus, "Distributed robotic sensor networks: An information-theoretic approach," *The Inter. Journal of Robotics Research*, vol. 31, no. 10, pp. 1134–1154, 2012.

[243] P. Dames, M. Schwager, V. Kumar, and D. Rus, "A decentralized control policy for adaptive information gathering in hazardous environments," in *IEEE Conference on Decision and Control*, 2012, pp. 2807–2813.

[244] P. Dames and V. Kumar, "Autonomous localization of an unknown number of targets using teams of mobile sensors," *IEEE Trans. on Autom. Science and Eng.*, vol. 12, no. 3, pp. 850–864, 2015.

[245] B. Charrow, V. Kumar, and N. Michael, "Approximate representations for multi-robot control policies that maximize mutual information," *Autonomous Robots*, vol. 37, no. 4, pp. 383–400, 2014.

[246] T. H. Chung, J. W. Burdick, and R. M. Murray, "A decentralized motion coordination strategy for dynamic target tracking," in *IEEE International Conference on Robotics and Automation*, 2006, pp. 2416–2422.

[247] C. M. Kreucher, "An information-based approach to sensor resource allocation," Ph.D. dissertation, University of Michigan, 2005.

[248] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime planning for decentralized multi-robot active information gathering," *IEEE Robotics and Automation Letters*, 2018.

[249] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Decentralized active information acquisition: Theory and application to multi-robot SLAM," in *IEEE Conf. on Robotics and Autom.*, 2015, pp. 4775–4782.

[250] J. B. Orlin, A. S. Schulz, and R. Udwani, "Robust monotone submodular function maximization," in *Inter. Conference on Integer Programming and Combinatorial Optimization*, 2016, pp. 312–324.