Approximately Supermodular Scheduling Subject to Matroid Constraints

Luiz F. O. Chamon, Alexandre Amice, and Alejandro Ribeiro

***Abstract*—Control scheduling refers to the problem of assigning agents or actuators to act upon a dynamical system at specific times so as to minimize a quadratic control cost, such as the objective of the Linear-quadratic-Gaussian (LQG) or the Linear Quadratic Regulator (LQR). When budget or operational constraints are imposed on the schedule, this problem is in general NP-hard and its solution can therefore only be approximated even for moderately large systems. The quality of this approximation depends on the structure of both the constraints and the objective. This work shows that greedy scheduling is near-optimal when the constraints can be represented as an intersection of matroids, algebraic structures that encode requirements such as limits on the number of agents deployed per time slot, total number of actuator uses, and duty cycle restrictions. To do so, it proves that the LQG cost function is α-supermodular and provides a new $\alpha/(\alpha + P)$-optimality certificates for the greedy minimization of such functions over an intersections of P matroids. These certificates are shown to approach the $1/(1 + P)$ guarantee of supermodular functions in relevant settings. These results support the use of greedy algorithms in non-supermodular quadratic control problems as opposed to typical heuristics such as convex relaxations and surrogate figures of merit, e.g., the $\log\det$ of the controllability Gramian.**

I. INTRODUCTION

As the number and complexity of interconnected devices grows, fully observing and/or actuating these dynamical systems becomes infeasible. Choosing when and where to sense and act then becomes a fundamental problem with significant impact on the system performance. Many of these problems can be cast as discrete optimization problems in which a subset of available elements (sensors, tasks, or actuators) is chosen to optimize a control objective, e.g., quadratic estimation or control figures of merit [2]–[5]. The difficulty in solving these problems arises from the constraints imposed to meet cost, power, and/or communication requirements. For instance, we may only afford to use a few actuators per control cycle or agents may only be able to perform specific tasks at specific times. This leads to problems that are NP-hard to solve in general and whose solutions must therefore be approximated for anything beyond small-scale systems [6]–[12]. The choice of approximation method and its performance heavily depends on properties of the underlying problem.

In this work, we tackle the control scheduling problem in which agents or actuators are assigned to act at specific time instants so as to minimize a quadratic control cost. A common approach in this context is using a convex relaxation, typically including a sparsity promoting regularization [13]–[16]. Though practical, these methods lack approximation guarantees and may therefore result in solutions with poor performance. What is more, solving large-scale optimization problems can be challenging even when they are convex, especially when semidefinite relaxations are employed. Another avenue is to build the schedule using discrete optimization methods, such as tree pruning [17] or greedy search [18]–[20]. The latter solution is particularly attractive in practice due to its low complexity and iterative nature: actuators or agents are matched to time instants one at a time by selecting the feasible match that most reduces the objective.

Besides the computational aspect, greedy solutions also enjoy near-optimal guarantees. Two conditions are often used to derive these certificates. The first is the supermodularity of the objective, a diminishing return property displayed by set functions such as the log determinant or the rank of the controllability Gramian. The second, is that the problem requirements can be mapped into a combinatorial structure known as a *matroid* [21, Ch. 39]. These include, for instance, constraints on the maximum number of actuators/agents selected per time instant. Under these assumptions, greedy scheduling yields $1/2$-optimal solutions [22]. Other methods based on continuous extensions can be leveraged to improve this factor to $1 - 1/e$ [23].

In practice, however, both of these assumptions can be quite stringent. First, many cost functions used in control applications are not supermodular. In particular, this is the case of the quadratic control objectives commonly deployed in optimal control, e.g., for linear-quadratic-Gaussian (LQG) and linear quadratic (LQR) regulators [10], [12], that are supermodular only under stringent conditions [19]. Second, real-world applications typically have requirements too general to be mapped onto a matroid. For instance, there are often restrictions on both the number of actuators per time instant and the number of use per actuator, especially in battery-operated or power-starved systems. These issues can be sidestepped by using supermodular surrogates such as the log determinant [10], [18], despite it being a poor substitute for quadratic costs in general [5, Remark 1]. Alternatively, guarantees have been obtained under laxer supermodularity conditions. They are, however, only available for the uniform matroid constraint found in selection problems [5], [6], [24]–[28]. The generic, single matroid case was tackled in a preliminary version of this work [1].

In this paper, we set out to provide guarantees for the greedy solution of control scheduling problems subject to complex constraints represented as intersections of matroids. We start

Department of Electrical and Systems Engineering, University of Pennsylvania. e-mail: {luizf, amice, aribeiro}@seas.upenn.edu. Part of the results in this paper appeared in [1].

by describing the LQG/LQR control scheduling problem and showing how it can be posed as a constrained set function optimization problem (Section II). We then proceed to describe condition under which this problem is tractable. We first show how greedy solutions can be obtained if the constraints are an intersection of matroids (Section III-A) and derive near-optimal certificates for these solutions when the objective is approximately supermodular (Sections III-B and III-C). These results generalize those in [1] to the case of multiple matroid constraints. To conclude, we prove that the LQG cost function is approximately supermodular, thus yielding an explicit guarantee for the greedy control scheduling algorithm (Section IV). We argue that these guarantees improve in scenarios of practical interest by illustrating their typical values in simulations and motivate the usefulness of this new result in an agent dispatch application (Section V).

Notation: Lowercase and uppercase boldface letters represent vectors ($\boldsymbol{x}$) and matrices ($\boldsymbol{X}$) respectively, while calligraphic letters denote sets ($\mathcal{A}$). We write $|\mathcal{A}|$ for the cardinality of $\mathcal{A}$ and $\boldsymbol{X} \succeq 0$ ($\boldsymbol{X} \succ 0$) to say $\boldsymbol{X}$ is positive semi-definite (definite) matrix. Hence, $\boldsymbol{X} \preceq \boldsymbol{Y} \Leftrightarrow \boldsymbol{b}^T\boldsymbol{X}\boldsymbol{b} \leq \boldsymbol{b}^T\boldsymbol{Y}\boldsymbol{b}$ for all $\boldsymbol{b} \in \mathbb{R}^n$. We write $\lambda_{\max}$ and $\lambda_{\min}$ for the maximum and minimum eigenvalue of a matrix, respectively, and use $\mathbb{R}_+$ to denote the non-negative real numbers.

II. PROBLEM FORMULATION

Consider a discrete-time, linear dynamical system and let $\mathcal{V}_k$ denote the set of inputs available at time k as illustrated in Figure 1. Depending on the context, these abstract inputs can be used to represent actuators, agents, or both. Suppose that at each time instant, only a subset $\mathcal{S}_k \subseteq \mathcal{V}_k$ of inputs is used, so that the states $\boldsymbol{x}_k \in \mathbb{R}^n$ of the system evolve according to

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \sum_{i\in\mathcal{S}_k} \boldsymbol{b}_{i,k}u_{i,k} + \boldsymbol{w}_k, \tag{1}$$

where, for each time k, $\boldsymbol{A}_k$ denotes the state transition matrix, $\boldsymbol{b}_{i,k} \in \mathbb{R}^n$ is a vector representing the effect of applying the control action $u_{i,k}$ to the i-th input, and $\boldsymbol{w}_k$ is a zero-mean Gaussian vector that models the process noise. We assume that $\{\boldsymbol{w}_j, \boldsymbol{w}_k\}$ are independent for $j \neq k$ and that their covariance matrices $\mathbb{E}\,\boldsymbol{w}_k\boldsymbol{w}_k^T = \boldsymbol{W}_k$ are either positive definite ($\boldsymbol{W}_k \succ 0$) for all k or zero ($\boldsymbol{W}_k = \boldsymbol{0}$) for all k, which accounts for the deterministic dynamics case. Let $\overline{\mathcal{V}} = \mathcal{V}_0 \cup \mathcal{V}_1 \cup \cdots \cup \mathcal{V}_{N-1}$ be the set of all actuators available over the N-steps time window $[0, N-1]$ and let $\overline{\mathcal{V}} \supseteq \mathcal{S} = \mathcal{S}_0 \cup \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_{N-1}$ be called a *schedule*, as it denotes the set of active inputs at each time instant. We assume without loss of generality that $\mathcal{V}_j \cap \mathcal{V}_k = \emptyset$ for all $j \neq k$. Any input v available at more than a single time instant can then be represented by unique copies, e.g., as $v^j \in \mathcal{V}_j$ and $v^k \in \mathcal{V}_k$ (Figure 1).

Given a schedule $\mathcal{S} \subseteq \overline{\mathcal{V}}$, designing the control actions $u_{i,k}$ reduces to a classical optimal control problem, since (1) describes a well-defined, time-varying dynamical system. In particular, we consider the linear-quadratic-Gaussian (LQG) control problem

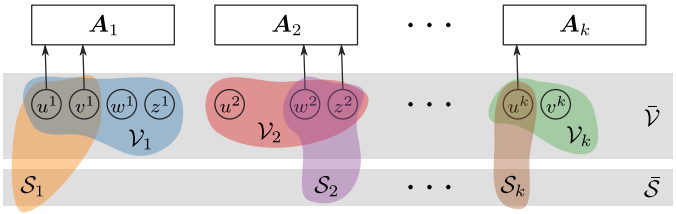


Figure 1. Illustration of time-specific and overall input sets and schedules.

$$V^\star(\mathcal{S}) = \min_{\mathcal{U}(\mathcal{S})} \mathbb{E}\left[\sum_{k=0}^{N-1}\left(\boldsymbol{x}_k^T\boldsymbol{Q}_k\boldsymbol{x}_k + \sum_{i\in\mathcal{S}\cap\mathcal{V}_k} r_{i,k}u_{i,k}^2\right) + \boldsymbol{x}_N^T\boldsymbol{Q}_N\boldsymbol{x}_N\right], \tag{2}$$

where $\mathcal{U}(\mathcal{S}) = \{u_{i,k} \mid i \in \mathcal{S}\cap\mathcal{V}_k\}_{k=0}^{N-1}$ is the set of valid control actions, $\boldsymbol{Q}_k \succ 0$ for all k are the state weights, and $r_{i,k} > 0$ for all i and k are the input weights. The relative value of these weights describe the trade-off between state regulation ($\boldsymbol{Q}_k$) and input cost ($r_{i,k}$). The expectation in (2) is taken with respect to the process noise sequence $\{\boldsymbol{w}_k\}$ and the initial state $\boldsymbol{x}_0$, assumed to be a Gaussian random variable with mean $\bar{\boldsymbol{x}}_0$ and covariance $\boldsymbol{\Sigma}_0 \succ 0$. When $\boldsymbol{w}_k = \boldsymbol{0}$ for all k, (2) reduces to the linear quadratic regulator (LQR) problem. It is useful to recall that (2) has a closed-form solution that we describe in the following proposition.

Proposition 1. *Given a schedule $\mathcal{S} \subseteq \overline{\mathcal{V}}$, the optimal value $V^\star(\mathcal{S})$ of the LQG problem in* (2) *can be written as*

$$V^\star(\mathcal{S}) = \operatorname{Tr}\left[\boldsymbol{\Sigma}_0\boldsymbol{P}_0(\mathcal{S})\right] + \sum_{k=0}^{N-1}\operatorname{Tr}\left[\boldsymbol{W}_k\boldsymbol{P}_{k+1}(\mathcal{S})\right], \tag{3}$$

where the $\boldsymbol{P}_k(\mathcal{S})$ are obtained via the backward recursion

$$\boldsymbol{P}_k(\mathcal{S}) = \boldsymbol{Q}_k + \boldsymbol{A}_k^T\left(\boldsymbol{P}_{k+1}^{-1}(\mathcal{S}) + \sum_{i\in\mathcal{S}\cap\mathcal{V}_k} r_{i,k}^{-1}\boldsymbol{b}_{i,k}\boldsymbol{b}_{i,k}^T\right)^{-1}\boldsymbol{A}_k, \tag{4}$$

starting with $\boldsymbol{P}_N = \boldsymbol{Q}_N$.

Proof. This result follows directly from the classical dynamic programming argument for the LQG by taking into account only the active inputs in $\mathcal{S}$ (e.g., see [29]). For ease of reference, we provide a derivation in the appendix. ■

Control scheduling refers to the problem of finding a schedule $\mathcal{S}$ that minimizes the control cost in (2) subject to time-input constraints, i.e.,

$$\begin{aligned} \underset{\mathcal{S}\subseteq\overline{\mathcal{V}}}{\text{minimize}} \quad & J(\mathcal{S}) \triangleq V^\star(\mathcal{S}) - V^\star(\emptyset) \\ \text{subject to} \quad & \mathcal{S} \in \mathcal{I}_p, \quad p = 1, \ldots, P, \end{aligned} \tag{PI}$$

where $\mathcal{I}_p \subseteq 2^{\overline{\mathcal{V}}}$ are families of subsets of $\overline{\mathcal{V}}$ that enumerates admissible schedules and $2^{\mathcal{X}}$ denotes the power set of $\mathcal{X}$ (the collection of all finite subsets). Typical scheduling requirements include (i) limits on the total number of control

actions, (ii) limits on the number of inputs used per time instant, (iii) restrictions on the consecutive use of inputs, and combinations thereof. Observe that the constant $V^\star(\emptyset)$ in the objective of (PI) does not affect the solution of the optimization problem. It is used so that $J(\emptyset) = 0$, which simplifies the presentation of our near-optimal certificates.

The complexity of (PI) is tightly related to the anatomy of its constraints and objective. Indeed, even obtaining a feasible schedules for (PI) can be hard depending on the structure of the $\mathcal{I}_p$. Not to mention obtaining a good one. In fact, (PI) is NP-hard even for constraints as simple as a budget on the total number of control actions ($p = 1$ and $\mathcal{I} = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid |\mathcal{S}| \leq s\}$) [6]–[12]. Still, depending on the nature of the objective and constraints, there exist simple algorithms able to provide near-optimal approximate solutions. In the following section, we examine an abstract version of (PI) and determine conditions under which these guaranteed approximations are possible, focusing on greedy methods. We will show that when $\mathcal{I}_p$ are independent sets of matroids (Section III-A), an algebraic structure that generalize the notions of linear independence in vector spaces, and the cost function is α-supermodular (Section III-B), a relaxed diminishing return property of set functions, greedy methods provide $[\alpha/(\alpha+P)]$-optimal schedules (Theorem 1). We then proceed to bound α for the cost J (Section IV) and give explicit guarantees for greedy solutions of (PI) in terms of known parameters of the problem (Theorem 2).

III. WHEN IS CONTROL SCHEDULING TRACTABLE?

In general, the control scheduling problem (PI) is NP-hard. In fact, it is NP-hard in very simple cases, such as when the total number of control actions is bounded [6]–[12]. What is more, even finding a feasible schedule can be challenging depending on the structure of the constraints and their representation. To determine when and how solutions of (PI) can be obtained (or more precisely, approximated), we study the generic set function optimization problem

$$\begin{aligned} \mathcal{X}^\star \in \underset{\mathcal{X}\subseteq\mathcal{E}}{\operatorname{argmin}} \quad & f(\mathcal{X}) \\ \text{subject to} \quad & \mathcal{X} \in \mathcal{I}, \end{aligned} \tag{PII}$$

of which (PI) is an instance. To see this is the case, let $\mathcal{E} = \overline{\mathcal{V}}$, $f(\mathcal{X}) = J(\mathcal{X})$, and $\mathcal{I} = \bigcap_{p=1}^{P} \mathcal{I}_p$. In what follows, we introduce the conditions on the constraints (Section III-A) and objective (Section III-B) of (PII) such that a simple algorithm (greedy search) can be used to generate feasible solutions that we then prove are near-optimal (Section III-C). We then proceed to show that (PI) satisfies these conditions and derive explicit certificates for greedy control scheduling.

A. Matroids

We begin by discussing a constraint structure for which feasible schedules can be easily constructed. As we mentioned before, this can, in and of itself, be quite a challenging problem. Explicitly, we restrict $\mathcal{I}$ to be an intersection of independent sets of matroids, entities that extend the notion of linear independence in vector spaces to arbitrary algebraic structures. Formally,

Definition 1. A *matroid* $M = (\mathcal{E}, \mathcal{I})$ consists of a finite set of elements $\mathcal{E}$ and a family $\mathcal{I} \subseteq 2^{\mathcal{E}}$ of subsets of $\mathcal{E}$ called *independent sets* that satisfy:

(P1) $\emptyset \in \mathcal{I}$;
(P2) if $\mathcal{A} \subseteq \mathcal{B}$ and $\mathcal{B} \in \mathcal{I}$, then $\mathcal{A} \in \mathcal{I}$;
(P3) if $\mathcal{A}, \mathcal{B} \in \mathcal{I}$ and $|\mathcal{A}| < |\mathcal{B}|$, then there exists $e \in \mathcal{B} \setminus \mathcal{A}$ such that $\mathcal{A} \cup \{e\} \in \mathcal{I}$.

Hence, we define the collection of valid schedules $\mathcal{I}$ in (PII) to be of the form

$$\mathcal{I} = \bigcap_{p=1}^{P} \mathcal{I}_p, \text{ such that } (\mathcal{E}, \mathcal{I}_p) \text{ is a matroid for all } p. \tag{5}$$

As we discuss in Section IV, typical scheduling constraints (in particular those listed in Section II) can be described in terms of independence sets and their intersection. It is worth noting that matroids are not closed under intersections, so that $\mathcal{I}$ need not be a matroid [21, Ch. 39].

The use of matroid interscetions as constraints in (PII) is attractive for two reasons. The first is a direct consequence of the inheritance property of matroids, i.e., P1 and P2 in Definition 1.

Proposition 2. *For each $\mathcal{A} \in \mathcal{I}$ with $\mathcal{I}$ as in* (5)*, there exists a chain $\emptyset = \mathcal{A}_0 \subset \mathcal{A}_1 \subset \cdots \subset \mathcal{A}_T = \mathcal{A}$ such that $\mathcal{A}_t \in \mathcal{I}$ for all t. In particular, there exists a chain with $T = |\mathcal{A}|$.*

Hence, every feasible schedule in (PII) can be constructed element-by-element. In particular, so can any optimal solution $\mathcal{X}^\star$. This suggests an "interior point"-type algorithm that greedily minimizes the objective f at each step while maintaining feasibility. Explicitly, a greedy solution of (PII) is constructed by taking $\mathcal{G}_0 = \emptyset$ and incorporating elements of $\mathcal{E}$ one at a time, so that at step t,

$$\mathcal{G}_{t+1} = \mathcal{G}_t \cup \{g_t\} \tag{6}$$

where

$$\begin{aligned} g_t \in \underset{e\in\mathcal{E}\setminus\mathcal{G}_t}{\operatorname{argmin}} \quad & f\left(\mathcal{G}_t \cup \{e\}\right) \\ \text{subject to} \quad & \mathcal{G}_t \cup \{e\} \in \mathcal{I}. \end{aligned} \tag{7}$$

The algorithm stops once no element can be added to $\mathcal{G}_t$ without violating feasibility, i.e., when the argmin set in (7) is empty. Denote that final iteration by T. Naturally, $T \leq |\mathcal{E}|$ and the algorithm does terminate. What is more, note from (6) that not only is $\mathcal{G}_T \in \mathcal{I}$ by construction, but due to Proposition 2, every set $\mathcal{A} \in \mathcal{I}$ can be constructed by (6) for an appropriate f (e.g., $f(\mathcal{X}) = |\mathcal{X} \cap \mathcal{A}|$). In other words, the greedy algorithm does not prune any solution from the feasibility set of (PII).

The second reason for using matroid intersections is related to the exchange property (P3 in Definition 1) and is laid out in the following proposition:

Proposition 3. *Let $\mathcal{A}, \mathcal{B} \in \mathcal{I}$ with $\mathcal{I}$ as in* (5)*. If $|\mathcal{B}| > P|\mathcal{A}|$, then there exist at least $|\mathcal{B}| - P|\mathcal{A}|$ elements b of $\mathcal{B} \setminus \mathcal{A}$ such that $\mathcal{A} \cup \{b\} \in \mathcal{I}$.*

Proof. The proof proceed by induction over the matroids in $\mathcal{I}$. The base case for first matroid ($p = 1$) is readily obtained from P3 in Definition 1. Indeed, take $e \in \mathcal{B} \setminus \mathcal{A}$ such that $\mathcal{A} \cup \{e\} \in \mathcal{I}_1$ and let $\mathcal{B}' = \mathcal{B} \setminus \{e\}$. This pruning process can be repeated as long as $|\mathcal{B}'| > |\mathcal{A}|$, i.e., at least $|\mathcal{B}| - |\mathcal{A}|$ times.

Now, suppose the claim holds for the first $P' - 1 < P$ independence sets, i.e., there exists a set $\mathcal{C} \subseteq \mathcal{B} \setminus \mathcal{A}$ such that

$$|\mathcal{C}| > |\mathcal{B}| - (P' - 1)|\mathcal{A}| \text{ and } \mathcal{A} \cup \{c\} \in \bigcap_{p=1}^{P'-1} \mathcal{I}_p \text{ for all } c \in \mathcal{C}. \tag{8}$$

Notice that $\mathcal{B} \in \mathcal{I} \Rightarrow \mathcal{B} \in \bigcap_{p=1}^{P'} \mathcal{I}_p$ and since $\mathcal{C} \subset \mathcal{B}$, the inheritance property of matroids (P2 in Definition 1) implies that $\mathcal{C} \in \bigcap_{p=1}^{P'} \mathcal{I}_p$ as well. In particular, since $\mathcal{C} \in \mathcal{I}_{P'}$, we again obtain from P3 in Definition 1 that there exist $\mathcal{C}' \subseteq \mathcal{C} \setminus \mathcal{A}$ such that

$$|\mathcal{C}'| > |\mathcal{C}| - |\mathcal{A}| \text{ and } \mathcal{A} \cup \{c\} \in \mathcal{I}_{P'} \text{ for all } c \in \mathcal{C}'. \tag{9}$$

Together, (8) and (9) yield that $|\mathcal{C}'| > |\mathcal{B}| - P'|\mathcal{A}|$ and that $\mathcal{A} \cup \{c\} \in \bigcap_{p=1}^{P'-1} \mathcal{I}_p \cap \mathcal{I}_{P'}$ for all $c \in \mathcal{C}'$. ■

Though more abstract, this property is fundamental to obtain near-optimal certificates for the procedure in (6). Indeed, the following noteworthy corollary is obtained for $\mathcal{B} = \mathcal{X}^\star$ and $\mathcal{A} = \mathcal{G}_t$:

Corollary 1. *If $|\mathcal{X}^\star| > Pt$, then there exist at least $|\mathcal{X}^\star| - Pt$ elements $x \in \mathcal{X}^\star$ such that $\mathcal{G}_t \cup \{x\} \in \mathcal{I}$. Hence, it must be that the algorithm in* (6) *terminates only after $T \geq |\mathcal{X}^\star|/P$.*

At any given point, there are therefore at least $|\mathcal{X}^\star| - Pt$ elements $x \in \mathcal{X}^\star$ for which $f(\mathcal{G}_{t+1}) \leq f(\mathcal{G}_t \cup \{x\})$. When combined with some diminishing return property, this greedy property gives near-optimal guarantees as long as the procedure in (6) runs long enough. That is where the lower bound on T is useful. It stems directly from the fact that, if (6) stops and returns a set with $|\mathcal{G}_T| < |\mathcal{X}^\star|/P$, Proposition 3 implies there exists at least one element $x \in \mathcal{X}^\star$ such that $\mathcal{G}_T \cup \{x\} \in \mathcal{I}$. This contradicts the fact that the feasibility set of (7) must be empty for the algorithm to terminate.

Naturally, near-optimality depends not only on the feasibility set $\mathcal{I}$, but also on the objective f. For instance, if f is a monotone decreasing modular function, then $\mathcal{G}_T$ is an optimal solution of (PII) with $P = 1$ in (5) [21, Ch. 40]; if f is a monotone decreasing supermodular function, then $\mathcal{G}_T$ would be $1/(1 + P)$-optimal [22]. It is well-known, however, that the cost function J of the original control scheduling problem (PI) is not modular and is supermodular only under strict conditions [10], [12], [19]. We therefore consider a broader class of objective functions f known as α-supermodular.

B. α-supermodularity

Supermodularity (*submodularity*) refers a "diminishing returns" property of certain set functions that yields near-optimality certificates for their greedy minimization (maximization). It holds, for instance, for the rank or $\log\det$ of the controlability (observability) Gramian of underactuated (underobserved) systems as well as the Shannon entropy and mutual information of a set of random variables [30], [31]. Supermodularity, however, is a stringent condition that, in particular, does not hold for the LQG objective of the control scheduling problem (PI) [9], [10], [12]. To account for this, different forms of *approximate supermodularity* (*submodularity*) have been proposed by allowing controlled violations of the original "diminishing returns" property. The rationale is that if a function is close to supermodular, then it should behave similarly to a truly supermodular function. We formalize these statements in the sequel using the concept of α-supermodularity [25], [26].

Let $f : 2^{\mathcal{E}} \to \mathbb{R}$ be a function defined over subsets $\mathcal{X} \subseteq \mathcal{E}$. We say f is *normalized* if $f(\emptyset) = 0$ and f is *monotone decreasing* if for all sets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{E}$ it holds that $f(\mathcal{A}) \geq f(\mathcal{B})$. Note that if a function is normalized and monotone decreasing, then it is non-positive valued, i.e., $f(\mathcal{X}) \leq 0$ for all $\mathcal{X} \subseteq \mathcal{E}$. Define the incremental variation

$$\Delta_u f(\mathcal{X}) = f(\mathcal{X}) - f(\mathcal{X} \cup \{u\}) \tag{10}$$

to be the change in the value of f from adding $u \in \mathcal{E}$ to the set $\mathcal{X}$. Observe that $u \in \mathcal{X}$ implies that $\Delta_u f(\mathcal{X}) = 0$. Then,

Definition 2. A set function $f : 2^{\mathcal{E}} \to \mathbb{R}$ is *supermodular* if

$$\Delta_u f(\mathcal{A}) \geq \Delta_u f(\mathcal{B}) \tag{11}$$

for all sets $\mathcal{A} \subset \mathcal{B} \subseteq \mathcal{E}$ and elements $u \in \mathcal{E} \setminus \mathcal{B}$. A function f is *submodular* if $-f$ is supermodular.

Supermodular, monotone decreasing functions play an important role in the context of scheduling problems due to the celebrated near-optimal guarantees for their greedy minimization. For instance, if $\mathcal{I}$ is a single uniform matroid, i.e., $\mathcal{I} = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid |\mathcal{S}| \leq s\}$ (cardinality constraint), then $\mathcal{G}_T$ is $(1 - 1/e)$-optimal [32]. In the general case of $\mathcal{I}$ as in (5), it holds that [22]

$$f(\mathcal{G}_T) \leq \frac{1}{1 + P} f(\mathcal{X}^\star). \tag{12}$$

Without the supermodularity of the objective, solving or even approximating simple instances of (PII) can be quite challenging: there exist functions δ-close to supermodular that cannot be approximately optimized in polynomial time unless δ is small [33]. Hence, *how* we allow this "diminishing returns" assumption to be violated is crucial in obtaining useful guarantees. To that effect, we introduce α-supermodularity. This concept was first proposed in the context of auction design [34], though it has since been rediscovered and deployed in discrete optimization, estimation, and control [5], [24]–[28].

Definition 3 (α-supermodularity)**.** A set function $f : 2^{\mathcal{E}} \to \mathbb{R}$ is *α-supermodular*, $\alpha \in \mathbb{R}_+$, if

$$\Delta_u f(\mathcal{A}) \geq \alpha \, \Delta_u f(\mathcal{B}). \tag{13}$$

for all sets $\mathcal{A} \subset \mathcal{B} \subseteq \mathcal{E}$ and all $u \in \mathcal{E} \setminus \mathcal{B}$. A function f is *α-submodular* if $-f$ is α-supermodular.

Notice that α-supermodularity is a hereditary property in the sense that any α-supermodular function is also α'-supermodular for all $\alpha' < \alpha$. What is more, (13) always holds for $\alpha = 0$ if f is monotone decreasing. For that reason, we

are often interested in the largest value of α for which (13) holds, i.e.,

$$\bar{\alpha} = \min_{\substack{\mathcal{A}\subset\mathcal{B}\subseteq\mathcal{E} \\ u\in\mathcal{E}\setminus\mathcal{B}}} \frac{\Delta_u f(\mathcal{A})}{\Delta_u f(\mathcal{B})}. \tag{14}$$

When $\bar{\alpha} \in (0,1)$, we say f is *approximately supermodular*. If $\bar{\alpha} \geq 1$, (13) implies (11), in which case we refer to f simply as *supermodular* [30], [31]. Thus, α-supermodularity can be used to both relax or strengthen the classical concept of supermodularity in Definition 2.

Although Definition 3 is in terms of singleton updates, the approximate diminishing return property in (13) also holds for set-valued updates. This equivalent definition is often convenient in derivations.

Proposition 4. *The function $f : 2^{\mathcal{E}} \to \mathbb{R}$ is an α-supermodular set function if and only if*

$$f(\mathcal{Y}) - f(\mathcal{Y}\cup\mathcal{C}) \geq \alpha\left[f(\mathcal{Z}) - f(\mathcal{Z}\cup\mathcal{C})\right] \tag{15}$$

for all sets $\mathcal{Y}\subset\mathcal{Z}\subseteq\mathcal{E}$ and $\mathcal{C}\in\mathcal{E}\setminus\mathcal{Z}$.

Proof. The necessity part is straightforward: taking $\mathcal{C}=\{u\}$ for $u\in\mathcal{E}\setminus\mathcal{Z}$ in (15) yields (13). Sufficiency follows by induction. Consider an arbitrary enumeration of $\mathcal{C}$ and let $\mathcal{C}_t = \{u_1,\dots,u_t\}$ be the set containing its first t elements. Then, the base case $\mathcal{C}_1=\{u_1\}$ holds directly from the definition of α-supermodularity in (13). Suppose now that (15) holds for $\mathcal{C}_{t-1}$, i.e.,

$$f(\mathcal{Y}) - f(\mathcal{Y}\cup\mathcal{C}_{t-1}) \geq \alpha\left[f(\mathcal{Z}) - f(\mathcal{Z}\cup\mathcal{C}_{t-1})\right]. \tag{16}$$

Let $\mathcal{A}=\mathcal{Y}\cup\mathcal{C}_{t-1}$, $\mathcal{B}=\mathcal{Z}\cup\mathcal{C}_{t-1}$, and $e=u_t$ in (13) to get

$$f(\mathcal{Y}\cup\mathcal{C}_{t-1}) - f(\mathcal{Y}\cup\mathcal{C}_{t-1}\cup\{u_t\}) \geq \alpha\left[f(\mathcal{Z}\cup\mathcal{C}_{t-1}) - f(\mathcal{Z}\cup\mathcal{C}_{t-1}\cup\{u_t\})\right]. \tag{17}$$

By summing (16) and (17), we conclude that (15) also holds for $\mathcal{C}_t=\mathcal{C}_{t-1}\cup\{u_t\}$. ∎

Another important property of α-supermodularity is that it is preserved by non-negative affine combinations.

Proposition 5. *Consider the set functions $f_i : 2^{\mathcal{E}} \to \mathbb{R}$ with f_i is α_i-supermodular. Then, for $\theta_i \geq 0$ and $\beta\in\mathbb{R}$, the function $g=\sum_i \theta_i f_i + \beta$ is $\min(\alpha_i)$-supermodular.*

Proof. See [5, Lemma 1] ∎

Definition 3 turns out to be a fruitful relaxation of Definition 2. For instance, it has been shown that if f is α-supermodular and $\mathcal{I}$ is composed of a single uniform matroid (cardinality constraint), then its greedy solution is $(1-e^{-\alpha})$-optimal [5], [24], [25], [34]. When $\mathcal{I}$ is a single arbitrary matroid ($P=1$), the greedy solution is $\alpha/2$-optimal [1]. In these cases, α not only quantifies the diminishing return violations, but also the loss in near-optimal guarantee due to these violations. Explicit lower bounds on α have been provided in the context of sampling, experiment design, estimation, and control [1], [5], [25], [26]. Still, these guarantees do not directly extend to the matroid intersection constraint in (5). In the next section, we determine how relaxing supermodularity to α-supermodularity affects the guarantee in (12).

Before proceeding, however, it is worth noting that α in (14) is related to the *supermodularity ratio*, a relaxation based on a different, though equivalent, definition of supermodularity. Explicitly, the supermodularity ratio is defined as the largest γ for which

$$\sum_{w\in\mathcal{W}} \Delta_w f(\mathcal{L}) \geq \gamma\left[f(\mathcal{L}) - f(\mathcal{L}\cup\mathcal{W})\right], \tag{18}$$

for all disjoint sets $\mathcal{W},\mathcal{L}\subseteq\mathcal{E}$. It was introduced in [6][1] in the context of variable selection, although the resulting guarantees depended on sparse eigenvalues that are NP-hard to compute. Explicit (P-computable) lower bounds on γ were derived in [27], [28], [35], although it is worth noting that the previous bounds on α obtained in [25], [36] also hold for the supermodularity ratio:

Proposition 6. *Let f be an α-supermodular and denote its supermodularity ratio by γ. Then, $\alpha\leq\gamma$.*

Proof. We proceed by showing that (18) holds with $\gamma=\alpha$ for any α-supermodular function. To do so, consider an enumeration of $\mathcal{W}=\{w_1,\dots,w_{|\mathcal{W}|}\}$ and write

$$f(\mathcal{L}) - f(\mathcal{L}\cup\mathcal{W}) = \sum_{k=1}^{|\mathcal{W}|} \Delta_{w_k} f\left(\mathcal{L}\cup\{w_1,\dots,w_{k-1}\}\right). \tag{19}$$

Since f is α-supermodular, we can upper bound each of the increments in (19) using (13) to obtain

$$f(\mathcal{L}) - f(\mathcal{L}\cup\mathcal{W}) \leq \alpha^{-1}\sum_{k=1}^{|\mathcal{W}|} \Delta_{w_k} f(\mathcal{L}). \tag{20}$$

Comparing (18) and (20) concludes the proof. ∎

C. Matroid-constrained α-supermodular minimization

At this point, we have fully specified the optimization problem we wish to solve and the algorithm we will use to do so. Explicitly, we are interested in using the greedy method (6) to approximate the solution of (PII) when the constraint $\mathcal{I}$ is an intersection of matroids as in (5) and the objective f is a monotone decreasing, α-supermodular function as per Definition 3. The main result of this section presented in Theorem 1 provides a near-optimal guarantee under these conditions. In other words, we show that, though NP-hard to solve exactly, α-supermodular minimization subject to an intersection of matroids can be approximated in polynomial time.

Theorem 1. *Consider* (PII) *with $\mathcal{I}$ being an intersection of matroids as in* (5) *and f being a normalized, monotone decreasing, α-supermodular function (i.e., $f(\mathcal{A})\leq 0$ for all $\mathcal{A}\subseteq\mathcal{E}$). Let $\mathcal{G}_T$ be its greedy solution from* (6)*. Then,*

$$f(\mathcal{G}_T) \leq \frac{\alpha}{\alpha+P} f(\mathcal{X}^\star). \tag{21}$$

[1] Although [6] and subsequent literature define γ in terms of submodularity, it can be recovered by taking $-f$ in (18). Recall that if f is supermodular, then $-f$ is submodular.

Theorem 1 provides a near-optimal certificate for the greedy solution of α-supermodular minimization problems subject to multiple matroid constraints. Since the values of f are non-positive due to the normalization assumption, the result in (21) may not be straightforward to interpret. Nevertheless, it can be written equivalently in terms of improvements with respect to the empty solution, namely,

$$\frac{f(\mathcal{G}) - f(\mathcal{X}^\star)}{f(\emptyset) - f(\mathcal{X}^\star)} \leq \frac{P}{\alpha + P}.$$

As expected from previous results, the guarantee in (21) bounds the suboptimality of greedy search in terms of the α-supermodularity of the cost function. When $\alpha < 1$, (21) quantifies the loss in performance guarantee due to the objective f violating the diminishing returns property. Confirming our initial intuition, the larger the violations, i.e., the smaller α, the worst the guarantee. On the other hand, (21) shows that the classical certificate for supermodular functions in (12) can be strengthened when f has stronger diminishing returns structures, i.e., when $\alpha \geq 1$. It is easy to see that Theorem 1 is consistent with previous results for single [32] and multiple [22] matroid constraints. Indeed, we can recover (12) by taking $\alpha = 1$ in (21).

Observe, however, that (21) decreases linearly with P. In other words, the guarantees for greedy search deteriorates as more matroids are needed to represent $\mathcal{I}$. In a sense, this is the constraint counterpart of α-supermodularity: the further away from a pure matroid the constraints are, the worst the greedy algorithm is guaranteed to perform. Effectively, (21) states that the more the more constraints need to be satisfied, i.e., the larger P, the harder the problem becomes. Note that P can be replaced by the minimum number of matroids needed to represent $\bigcap_{p=1}^{P} \mathcal{I}$, so that the guarantee from Theorem 2 can sometimes be improved. Determining this minimum number, however, can be quite intricate. Still, Theorem 1 is a *worst case* bound and as is the case with the classical result for greedy supermodular minimization [22], [32], better performance is often obtained in practice. Still, pathological cases that approach the bound can be constructed (see Section V).

It is worth noting that in the single matroid case ($P = 1$), Theorem 1 is strictly (though not significantly) stronger than the guarantee from [1] for $\alpha < 1$. This difference is more accentuated for small α: for instance, if $\alpha = 0.2$, (21) is 60% stronger than the certificate in [1]. A similar certificate was obtained in [34] with $\alpha \geq 1$ for a single matroid constraints.

We now proceed with the proof of Theorem 1.

Proof. This proof follows similarly to the one for the cardinality constraint (uniform matroid) problem, but relying on the exchange property of matroids through Proposition 3. It may be informative to recall the proof technique in that simpler case first, e.g., by referring to [5, Thm. 1].

Let $|\mathcal{X}^\star| = r$. Partition the optimal solution in (PII) such that $\mathcal{X}^\star = \bigcup_{j=0}^{\lfloor r/P \rfloor} \mathcal{X}_j^\star$, $|\mathcal{X}_j^\star| \leq P$, and

$$\mathcal{G}_t \cup \{x^\star\} \in \mathcal{I} \text{ for } x^\star \in \mathcal{X}_t^\star \text{ for all } t = 0, \dots, \left\lfloor \frac{r}{P} \right\rfloor. \quad (22)$$

Observe that such a partition exists due to the matroid exchange property from Proposition 3. Fix an arbitrary enumeration of each partition $\mathcal{X}_t^\star = \{x_{t1}^\star, \dots, x_{tp}^\star\}$. The following proof is independent of the specific enumeration.

Use the fact that f is monotone decreasing to write

$$\begin{aligned} f(\mathcal{X}^\star) &\geq f(\mathcal{G}_T \cup \mathcal{X}^\star) \\ &= f(\mathcal{G}_T) - \sum_{t=0}^{\lfloor r/P \rfloor} \left[\sum_{j=1}^{|\mathcal{X}_t^\star|} \Delta_{x_{tj}^\star} f(\mathcal{T}_t^{j-1}) \right], \end{aligned} \quad (23)$$

where the set $\mathcal{T}_t^j$ contains the greedy solution $\mathcal{G}_T$, the elements of all partitions $\mathcal{X}_i^\star$ up to $t-1$, and the first j elements of partition $\mathcal{X}_t^\star$. They can be defined recursively as $\mathcal{T}_0^0 = \mathcal{G}_T$, $\mathcal{T}_t^0 = \mathcal{T}_{t-1}^0 \cup \mathcal{X}_t^\star$, and $\mathcal{T}_t^j = \mathcal{T}_{t-1}^0 \cup \{x_{t1}^\star, \dots, x_{tj}^\star\}$. Then, we can bound (23) by distinguishing two cases: (i) if $x_{tj}^\star \notin \mathcal{T}_t^{j-1}$, the α-supermodularity of f yields

$$\Delta_{x_{tj}^\star} f\left(\mathcal{T}_t^{j-1}\right) \leq \alpha^{-1} \Delta_{x_{tj}^\star} f\left(\mathcal{G}_t\right), \quad (24)$$

since $\mathcal{G}_t \subseteq \mathcal{G}_T \subseteq \mathcal{T}_t^j$ for all t and j; (ii) if $x_{tj}^\star \in \mathcal{T}_t^{j-1}$, then $\Delta_{x_{tj}^\star} f\left(\mathcal{T}_t^{j-1}\right) = 0$ and (24) holds trivially. Using (24) in (23) gives

$$f(\mathcal{X}^\star) \geq f(\mathcal{G}_T) - \alpha^{-1} \sum_{t=0}^{\lfloor r/P \rfloor} \left[\sum_{j=1}^{|\mathcal{X}_t^\star|} \Delta_{x_{tj}^\star} f(\mathcal{G}_t) \right]. \quad (25)$$

The bound in (25) can be simplified using the greedy nature of the algorithm in (6). Indeed, observe that (22) implies that $\Delta_{x_{tj}^\star} f(\mathcal{G}_t) \leq \Delta_{g_t} f\left(\mathcal{G}_t\right)$ for g_t is the t-th element selected as in (7). Hence,

$$f(\mathcal{X}^\star) \geq f(\mathcal{G}_T) - \alpha^{-1} \sum_{t=0}^{\lfloor r/P \rfloor} \left[\sum_{j=1}^{|\mathcal{X}_t^\star|} \Delta_{g_t} f(\mathcal{G}_t) \right]. \quad (26)$$

Using the fact that the increments are always non-negative [see (10)] and since $|\mathcal{X}_t^\star| \leq P$, (26) reduces to

$$f(\mathcal{X}^\star) \geq f(\mathcal{G}_T) - \alpha^{-1} P \sum_{t=0}^{\lfloor r/P \rfloor} \Delta_{g_t} f(\mathcal{G}_t). \quad (27)$$

To conclude, observe from (10) that $f(\mathcal{G}_T) \leq f(\mathcal{G}_t) = -\sum_{t=0}^{t-1} \Delta_{g_t} f(\mathcal{G}_t)$ and recall from Proposition 3 that $T \geq r/P$. Thus,

$$f(\mathcal{X}^\star) \geq (1 + \alpha^{-1} P) f(\mathcal{G}_T),$$

which can be rearranged as in (21). ∎

IV. NEAR-OPTIMAL INPUT SCHEDULING

In the previous section, we established when, how, and how well solutions of the generic problem (PII) can be approximated. We did so by developing a theory for the greedy optimization of α-supermodular functions subject to multiple matroid constraints. In the sequel, we reconcile this general theory with the control scheduling problem (PI) we set out to solve in the beginning of this paper.

Start by noticing that the feasible set of (PI) is exactly the intersection of the $\mathcal{I}_p$ and that restricting them to be independent sets of matroids is in fact quite mild. Indeed, all typical schedule constraints we have discussed so far can be

Algorithm 1 Greedy algorithm for (PI)

Let $\mathcal{S}_0 \leftarrow \emptyset$, $\mathcal{Z}_0 \leftarrow \mathcal{E}$, and $t \leftarrow 0$
while $\mathcal{Z}_t \neq \emptyset$
  $g \leftarrow \operatorname{argmin}_{u \in \mathcal{Z}_t} V^\star(\mathcal{S}_t \cup \{u\})$
  $\mathcal{Z}_{t+1} \leftarrow \mathcal{Z}_t \setminus \{g\}$
  if $\mathcal{S}_t \cup \{g\} \in \cap_{p=1}^P \mathcal{I}_p$ **then**
    $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{g\}$
  else
    $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t$
  end if
  $t \leftarrow t+1$
end while
$\mathcal{S}_g \leftarrow \mathcal{S}_t$

described in those terms. They are in fact instances of uniform, partition, and transversal matroids respectively:

- bound on the total number of control actions: $\mathcal{I} = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid |\mathcal{S}| \leq s\}$,
- bound on the number of inputs used per time instant: $\mathcal{I} = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid |\mathcal{S} \cap \mathcal{V}_k| \leq s_k\}$, and
- restriction on the consecutive use of inputs: $\mathcal{I} = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid v^k \notin \mathcal{S}$ or $v^{k+1} \notin \mathcal{S}$, for $v^k, v^{k+1} \in \overline{\mathcal{V}}\}$,

Under these conditions, (PI) can be solved greedily using Algorithm 1. Note that Algorithm 1 does not directly implement (6) for (PI). Both procedures are nevertheless equivalent due to the matroidal structure of $\mathcal{I}$. Indeed, for any set $\mathcal{A} \subset \mathcal{B} \in \bigcap_{p=1}^P \mathcal{I}_p$, if $\mathcal{A} \cup \{g\} \notin \bigcap_{p=1}^P \mathcal{I}_p$ for $g \in \overline{\mathcal{V}}$, then $\mathcal{B} \cup \{g\} \notin \bigcap_{p=1}^P \mathcal{I}_p$. In other words, if adding an element g to $\mathcal{S}_t$ would make the schedule infeasible, then adding it to $\mathcal{S}_{t'}$, $t' > t$, would do the same. As opposed to (6), Algorithm 1 simply discards such elements. Additionally, it omits the constant normalization $V^\star(\emptyset)$ since it does not depend on the schedule.

To proceed, we must show that the objective function J is α-supermodular and monotone decreasing as well as provide an explicit lower bound on $\bar{\alpha}$ in (14). Without this bound, the guarantee from Theorem 1 would be of little practical value as it could not be computed *a priori*. What is more, near-optimality certificates based on α-supermodularity are vacuous unless we can show that α is strictly larger than zero and, at least under certain conditions, substantially so. We address these issue with the following proposition:

Proposition 7. *Let $\boldsymbol{A}_k$ in* (1) *be full rank for all k. The normalized actuator scheduling problem objective J is (i) monotonically decreasing and (ii) α-supermodular with*

$$\alpha \geq \min_{k=0,\dots,N-1} \alpha_k, \tag{28}$$

where

$$\alpha_k \geq \frac{\lambda_{\min}\left[\tilde{\boldsymbol{P}}_{k+1}^{-1}(\emptyset)\right]}{\lambda_{\max}\left[\tilde{\boldsymbol{P}}_{k+1}^{-1}(\overline{\mathcal{V}}) + \sum_{i \in \mathcal{V}_k} r_{i,k}^{-1} \tilde{\boldsymbol{b}}_{i,k} \tilde{\boldsymbol{b}}_{i,k}^T\right]}, \tag{29}$$

for $\tilde{\boldsymbol{P}}_{k+1}(\mathcal{S}) = \boldsymbol{H}_k^{1/2} \boldsymbol{P}_{k+1}(\mathcal{S}) \boldsymbol{H}_k^{1/2}$, $\tilde{\boldsymbol{b}}_{i,k} = \boldsymbol{H}_k^{-1/2} \boldsymbol{b}_{i,k}$, $\boldsymbol{H}_0 = \boldsymbol{A}_0 \boldsymbol{\Sigma}_0 \boldsymbol{A}_0^T$, and $\boldsymbol{H}_k = \boldsymbol{A}_k \boldsymbol{W}_{k-1} \boldsymbol{A}_k^T$ for $k \geq 1$.

Proof. See appendix. ■

Remark 1. The full rank hypothesis on $\boldsymbol{A}_k$ can be lifted using a continuity argument. However, the bound in (28) is trivial ($\alpha \geq 0$) for rank deficient state transition matrices, so we focus only on the case of practical significance.

Proposition 7 states that the objective of (PI) is α-supermodular for α as in (28). Notice that the lower bound on $\bar{\alpha}$ is explicit in that it can be evaluated in terms of the parameters of the dynamical system and the weights $\boldsymbol{Q}_k$ and $r_{i,k}$ in (2). In other words, (28) allows us to determine α *a priori* for the objective J and with Theorem 1, give near-optimality guarantees on the greedy solution of (PI). We collect this result in the theorem below.

Theorem 2. *Consider the control scheduling problem* (PI) *in which $(\overline{\mathcal{V}}, \mathcal{I}_p)$ is a matroid for each p and let $\mathcal{S}^\star$ be its optimal solution and $\mathcal{S}_g$ be its greedy solution obtained using Algorithm 1. Then,*

$$J(\mathcal{S}_g) \leq \frac{\alpha}{\alpha + P} J(\mathcal{S}^\star) \tag{30}$$

with $\alpha \geq \min_{k=0,\dots,N-1} \alpha_k$ for

$$\alpha_k \geq \frac{\lambda_{\min}\left[\tilde{\boldsymbol{P}}_{k+1}^{-1}(\emptyset)\right]}{\lambda_{\max}\left[\tilde{\boldsymbol{P}}_{k+1}^{-1}(\overline{\mathcal{V}}) + \sum_{i \in \mathcal{V}_k} r_{i,k}^{-1} \tilde{\boldsymbol{b}}_{i,k} \tilde{\boldsymbol{b}}_{i,k}^T\right]}, \tag{31}$$

where $\tilde{\boldsymbol{P}}_{k+1}(\mathcal{S}) = \boldsymbol{H}_k^{1/2} \boldsymbol{P}_{k+1}(\mathcal{S}) \boldsymbol{H}_k^{1/2}$, for $\boldsymbol{P}_k(\mathcal{S})$ defined in (4)*, and $\tilde{\boldsymbol{b}}_{i,k} = \boldsymbol{H}_k^{-1/2} \boldsymbol{b}_{i,k}$. The transformations are the positive-definite square roots of $\boldsymbol{H}_k$, defined as $\boldsymbol{H}_0 = \boldsymbol{A}_0 \boldsymbol{\Sigma}_0 \boldsymbol{A}_0^T$ and $\boldsymbol{H}_k = \boldsymbol{A}_k \boldsymbol{W}_{k-1} \boldsymbol{A}_k^T$ for $k \geq 1$.*

Theorem 2 provides a near-optimal certificate for the greedy solution of the control scheduling problem (PI) as a function of its parameters. Note that the larger the α, the better the guarantee, and that although J is not supermodular in general, there are situations in which its violations of the diminishing returns property are small.

To see when this is the case, observe that (31) is large when

$$\sum_{i \in \mathcal{V}_0} r_{i,0}^{-1} \tilde{\boldsymbol{b}}_i \tilde{\boldsymbol{b}}_i^T \text{ is small} \quad \text{and} \quad \tilde{\boldsymbol{P}}_1^{-1}\left(\overline{\mathcal{V}}\right) \approx \tilde{\boldsymbol{P}}_1^{-1}(\emptyset). \tag{32}$$

Conditions in (32) occur when $\operatorname{diag}(r_{i,k}) \gg \boldsymbol{Q}_k$, i.e., when the controller (2) gives more weight to the input cost than state regulation. This condition is readily obtained from the definition of $\boldsymbol{P}_k$ in (4). Figures 2 and 3 illustrates these observations by evaluating the bound from (31) for 100 random systems with $n = 100$ states controlled over a horizon of $N = 4$ steps with $\boldsymbol{Q} = \boldsymbol{I}$ and $r_{i,k} = \gamma$ (see Section V for details). Clearly, as the controller actions cost grows, i.e., as γ increases, α grows and Theorem 2 yields stronger guarantees. Note that this is a scenario of considerable practical value. It is well-known that if no restriction is imposed on the input cost, any controllable set of inputs can drive the system to any state in a single step (*dead-beat controller*). Hence, when $\operatorname{diag}(r_{i,k}) \ll \boldsymbol{Q}_k$, the optimal value of the

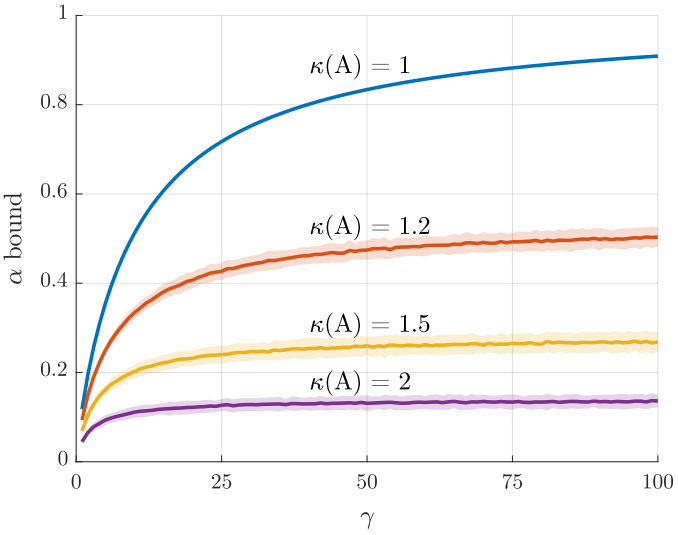


Figure 2. Bound on α from (28) for a deterministic dynamical system ($\boldsymbol{W}_k = \boldsymbol{0}$) and a schedule of length $N = 4$. Shaded regions span two standard deviations from the mean.

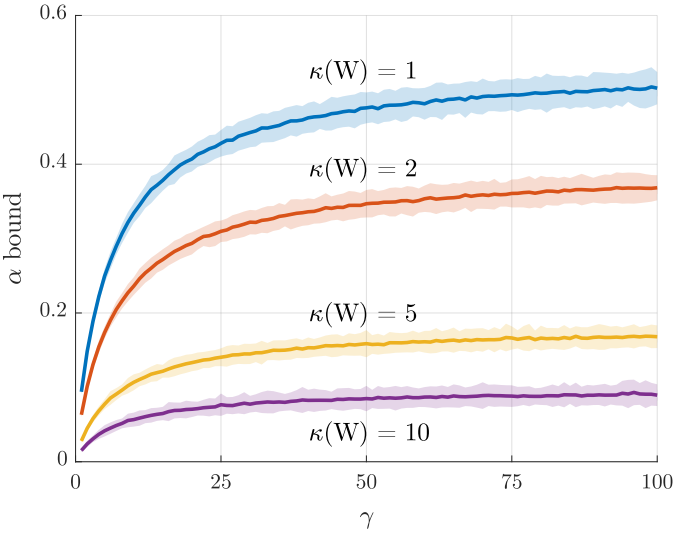


Figure 3. Bound on α from (28) for a dynamical system with disturbance and a schedule of length $N = 4$. Shaded regions span two standard deviations from the mean.

LQG only really differentiates between schedules that lead to controllable and uncontrollable input sets. On the other hand, careful scheduling can have a considerable impact when the actuation (or agent deployment) costs are high. This is also the case in which Theorem 2 provides stronger guarantees.

Observe also that since matrices are weighted by $\boldsymbol{H}_k$, the condition numbers of $\boldsymbol{A}$, $\boldsymbol{\Sigma}_0$, and $\boldsymbol{W}_k$ play an important role in the guarantees. Indeed, if $\boldsymbol{H}_k$ is poorly conditioned, there may be a large difference between the minimum and maximum eigenvalues in (31). This is illustrated in Figure 2, which shows that when the decay rate of the system modes are considerably different, i.e., the state transition matrix has large condition number $\kappa(\boldsymbol{A})$, the guarantees from Theorem 2 worsen. Figure 3 illustrates this phenomenon for the LQG controller, showing how the value of α decreases when the process noise does not affect the states homogeneously, i.e., the condition number of $\boldsymbol{W}_k$ grows.

In the sequel, we provide details on the experiments presented in this section and illustrate the use of greedy control scheduling under multiple non-trivial constraints using an agent dispatching application.

V. NUMERICAL EXAMPLES

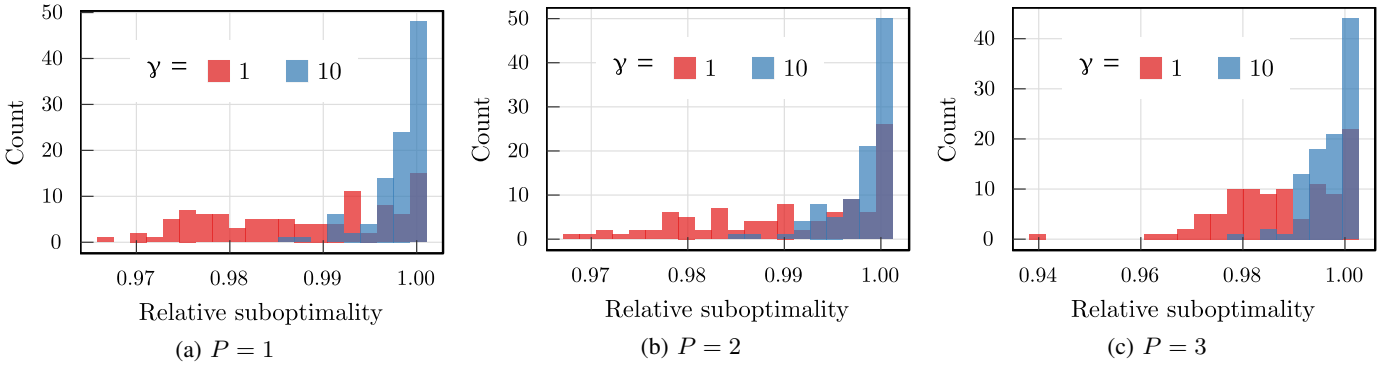
We start by detailing the experiments in Figures 2 and 3. We considered dynamical systems with $n = 100$ states for which the elements of $\boldsymbol{A}$ were drawn randomly from a standard Gaussian distribution, the input matrix $\boldsymbol{B} = \boldsymbol{I}$, i.e., direct state actuation, and $\boldsymbol{\Pi}_0 = 10^{-2}\boldsymbol{I}$. The state transition matrix $\boldsymbol{A}$ was normalized so that its spectral radius is 1.1, i.e., the dynamical systems are unstable. In Figure 2, the dynamical systems are deterministic, i.e., $\boldsymbol{W}_k = \boldsymbol{0}$. In Figure 3, $\boldsymbol{W}_k$ is a diagonal matrix whose elements were drawn uniformly at random from $[\kappa(\boldsymbol{W})^{-1}, 1]$ so that their condition number is $\kappa(\boldsymbol{W})$. The figures show the result for 100 independently drawn dynamical systems considering (PI) for $\boldsymbol{Q} = \boldsymbol{I}$, $r_{i,k} = \gamma$ for all i and k, and $N = 4$.

While Figures 2 and 3, along with Theorem 2, illustrate the wide range of parameters over which good performance certificates can be provided, it is worth noting that these are worst-case guarantees and that better results are common in practice. To illustrate this point, we evaluate the relative suboptimality of greedily selected schedules over 100 system realizations. Explicitly, we evaluate

$$\nu^\star(\mathcal{S}_g) = \frac{J(\mathcal{S}_g)}{J(\mathcal{S}^\star)},$$

where $\mathcal{S}_g$ and $\mathcal{S}^\star$ are the greedy (Algorithm 1) and optimal solutions of (PI) respectively. Since $\nu^\star$ depends on the optimal schedule, which can only be obtained by exhaustive search, we restrict ourselves to smaller dynamical systems with $n = 7$ states. State space matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are as before, with $\kappa(\boldsymbol{A}) = 1.2$, and $\boldsymbol{W}_k = \boldsymbol{0}$ (LQR case). We again take $\boldsymbol{Q} = \boldsymbol{I}$ and $r_{i,k} = \gamma$.

In Figure 4, we design schedules for $N = 4$ steps horizons with different numbers of matroid constraints P. In Figure 4a, we select at most 2 actuators per time step ($\mathcal{I}_1 = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid |\mathcal{S} \cap \mathcal{V}_k| \leq 2\}$). In addition to $\mathcal{I}_1$, Figure 4b also imposes that at most 5 control actions can be taken over the horizon ($\mathcal{I}_2 = \{\mathcal{S} \subseteq \overline{\mathcal{V}} \mid |\mathcal{S}| \leq 5\}$). Finally, Figure 4c includes a restriction on using the same input on consecutive time steps on top of $\mathcal{I}_1$ and $\mathcal{I}_2$. Despite the fact that the lower bounds on $\nu^\star$ (recall that J is a non-positive function) from Theorem 2 range from 0.03 to 0.25, Figure 4 show that the typical performance of greedy scheduling in practice is considerably better. Though it did not often find the optimal schedule (between 15% and 23% of the realizations for $\gamma = 1$ and 40% and 46% for $\gamma = 10$), the resulting schedule performances were at most 5% lower than the optimum. Note, however, that there exist specific dynamical systems for which greedy performs close to the guarantee in Theorem 2. Indeed,



(a) $P = 1$ (b) $P = 2$ (c) $P = 3$

Figure 4. Relative suboptimality of greedy scheduling for different constraints (100 system realizations). (a) $\mathcal{I}_1$ (less than 2 actuators per time step); (b) $\mathcal{I}_1$ and $\mathcal{I}_2$ (less than 5 control actions over the horizon); (c) $\mathcal{I}_1$, $\mathcal{I}_2$, and $\mathcal{I}_3$ (inputs cannot be used on consecutive time slots).

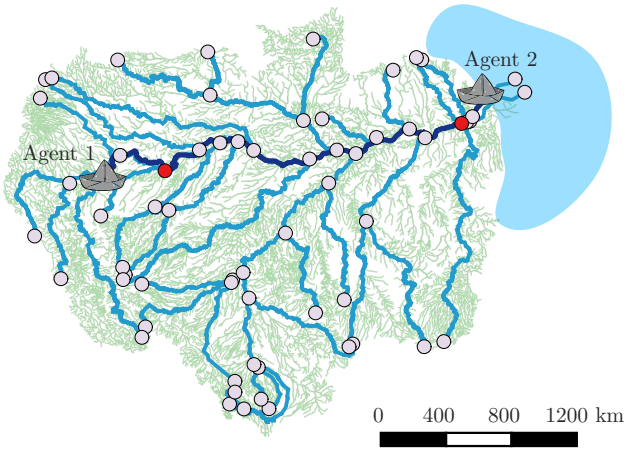


Figure 5. Amazon basin, amazon river (dark blue trace), system states (light grey circles), and chemical spill origins (red circles).

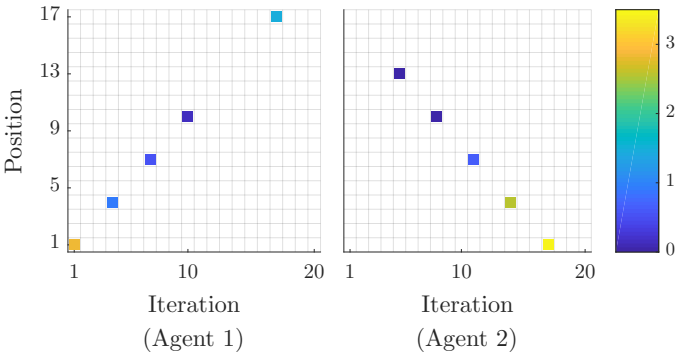


Figure 6. Greedy schedule and actuation energy of the spill control agents.

consider $\boldsymbol{A} = \boldsymbol{I}$ and

$$\boldsymbol{B} = \begin{bmatrix} 2 & 1 & 0 \\ 2 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Under constraint $\mathcal{I}_1$, i.e., if we schedule up to 2 inputs per time step, over a $N = 2$ steps horizon with $\gamma = 100$, the guarantee in (30) yields $\nu^\star(\mathcal{S}_g) \geq 0.392$, whereas in practice we achieve $\nu^\star(\mathcal{S}_g) \approx 0.423$.

A. Agent dispatching on the Amazon river

In this section, we illustrate the use of greedy control scheduling in an agent dispatching application to control the effect of spills on the Amazon river. Two agents navigate up and down the river (dark blue curve in Figure 5) over a preset route and use a chemical component to counteract damaging spills. Due to the limited capacity of each vessel and to avoid overusage, each agent is allowed to dump the component at most 5 times. What is more, at least 2 steps must be allowed between each action so the crew has time to setup. The goal of this dispatch is to reduce how much of the spill reaches the ocean (light blue water mass in Figure 5).

The Amazon drainage basin, showed in green in Figure 5, covers 7.5 million km^2 and is composed of over 7000 tributaries. We use a simplified description of the basin (blue traces in Figure 5) obtained by smoothing the original map [37]. Using this river network, we construct a weighted directed tree $\mathcal{G}$ whose vertices are the circles from in Figure 5 with the addition of a node midway between each circle. These vertices compose the $n = 127$ states $\boldsymbol{x}_k$ of the dynamical system. In $\mathcal{G}$, two vertices are connected if water flows between them, i.e., if there is a blue line between them in Figure 5. We assume that all river flow toward the ocean, denoted in Figure 5 by a light blue water mass. The adjacency matrix of $\mathcal{G}$ is then defined as

$$[\boldsymbol{G}]_{ij} = \begin{cases} \|\boldsymbol{z}_i - \boldsymbol{z}_j\|, & \text{if } (i,j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases} \tag{33}$$

where $\boldsymbol{z}_i \in \mathbb{R}^2$ is the position of node i on the map. We define its Laplacian as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{G}$, where $\boldsymbol{D}$ is a diagonal matrix whose elements are the sum of the columns of $\boldsymbol{G}$, and its symmetrized Laplacian as $\boldsymbol{L}' = \boldsymbol{D}' - (\boldsymbol{G} + \boldsymbol{G}^T)$, where $\boldsymbol{D}'$ is a diagonal matrix whose elements are the sum of the columns of $\boldsymbol{G} + \boldsymbol{G}^T$.

Using these Laplacians, we define the state transition matrix of the dynamical system in (1) as

$$\boldsymbol{A} = 0.901 \exp(-\boldsymbol{L}\Delta t) + 0.099 \exp(-\boldsymbol{L}'\Delta t), \tag{34}$$

where $\Delta t = 5$ is the sampling period. The dynamics in (34) are a combination of two processes: the first term corresponds to the *advection* process by which water flows to the ocean and the second term corresponds to a *diffusion* process. The combination coefficients are chosen so that the system is

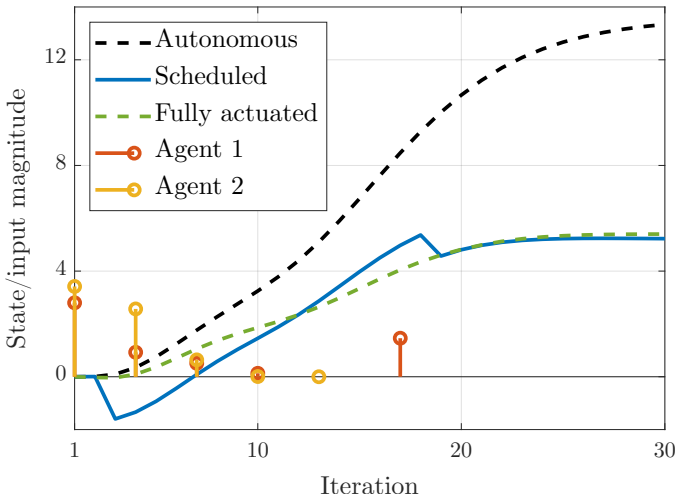


Figure 7. Agents actions and chemical concentrations in the ocean (light blue mass in Figure 5) for the autonomous system (no agent), greedy schedule, and full actuation.

marginally stable ($\|\boldsymbol{A}\| = 1$). In these experiments, we assume there is no process noise, i.e., $\boldsymbol{W}_k = \boldsymbol{0}$ for all k, and direct state actuation. The two spills are modeled by spikes in the initial state, namely $\boldsymbol{x}_0$ is zero except at the red nodes in Figure 5.

Agent 1 navigates the Amazon river (dark blue curve) left-to-right and Agent 2 navigates right-to-left (starting near the ocean). They can only actuate on their current position and are only allowed to do so on the states marked as circles in Figure 5. Thus, a centralized greedy scheduler designs a $N = 20$ steps action plan for the two agents taking into account their positions at each time step and their total number of actions and duty cycle constraints. To do so, it assumes $\boldsymbol{Q} = \boldsymbol{I}$ for both agents, but takes $r_{i,k}^{(1)} = 10$ for Agent 1 and $r_{i,k}^{(2)} = 20$ for Agent 2. In other words, Agent 1 is allowed to dump more cleaning component. The results of this dispatch are shown in Figures 6 and 7.

Notice that in the final schedule (Figure 6), the agents effectively act on top of the spills and towards the middle of the river to try and stop their spreading. Agent 1, in particular, saves cleaning reagent for one last action next to ocean. What is more, since it pays a lower price for actuation ($r_{i,k}^{(1)} < r_{i,k}^{(2)}$), it is able to use more reagent than Agent 2, which ends up concentrating its efforts in the beginning of its route (Figures 7). In the end, the vessels are able to mitigate the impact of the spills on the ocean waters, reducing contamination levels by 60% compared to the no-actuation solution. The final level achieved with these punctual actions is comparable to using an 64 agents that actuate every state at every time step with cost matrices $\boldsymbol{Q} = \boldsymbol{I}$ and $r_{i,k} = 2500$ (Figure 7).

VI. CONCLUSION

We studied the control scheduling problem in which agents or actuators are assigned to act upon a dynamical system at specific times so as to minimize the LQG/LQR objective. We considered the case in which budget and operational scheduling constraints can be encoded as an intersection of P matroids, which can describe any combination of limits on the number of agents deployed per time slot, total number of actuator uses, duty cycle restrictions... We showed that this structure allows schedules to be built greedily, i.e., by adding system inputs one-by-one choosing the one that most reduces the control cost. We then showed that for α-supermodular objectives this procedure is $\alpha/(\alpha + P)$-optimal and provided an explicit, system-dependent lower bound on α for the LQG/LQR cost. This bound approaches unity and yield supermodular-like guarantees in relevant application scenarios. Combining these results, we presented a near-optimal certificate for greedy scheduling that validates its empirical success. We believe that the near-optimality results derived for approximately supermodular functions are of independent interest and can be extended to other applications, such as experimental design using polymatroids. An important future challenge to overcome in these discrete control problems is taking into account state and actuation constraints.

REFERENCES

[1] L.F.O. Chamon, A. Amice, and A. Ribeiro, "Matroid-constrained approximately supermodular optimization for near-optimal actuator scheduling," in *Conf. on Decision and Contr.*, 2019.

[2] B.P. Gerkey and M.J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The Int. J. of Robot. Research*, vol. 23[9], pp. 939–954, 2004.

[3] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: Leveraging submodularity," in *Conf. on Decision and Contr.*, 2010, pp. 2572–2577.

[4] T.H. Summers, F.L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Trans. Contr. Netw. Syst.*, vol. 3[1], pp. 91–101, 2016.

[5] L.F.O. Chamon, G.J. Pappas, and A. Ribeiro, "Approximate supermodularity of Kalman filter sensor selection," *IEEE Trans. Autom. Control*, (accepted), arXiv:1912.03799.

[6] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection," in *Int. Conf. on Mach. Learning*, 2011.

[7] G. Sagnol, "Approximation of a maximum-submodular-coverage problem involving spectral functions, with application to experimental designs," *Discrete Appl. Math.*, vol. 161[1-2], pp. 258–276, 2013.

[8] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies," *J. Mach. Learning Research*, vol. 9, pp. 235–284, 2008.

[9] J. Ranieri, A. Chebira, and M. Vetterli, "Near-optimal sensor placement for linear inverse problems," *IEEE Trans. Signal Process.*, vol. 62[5], pp. 1135–1146, 2014.

[10] V. Tzoumas, M.A. Rahimian, G.J. Pappas, and A. Jadbabaie, "Minimal actuator placement with bounds on control effort," *IEEE Trans. Contr. Netw. Syst.*, vol. 3[1], pp. 67–78, 2016.

[11] 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.

[12] A. Olshevsky, "On (non)supermodularity of average control energy," *IEEE Trans. Contr. Netw. Syst.*, vol. 5[3], pp. 1177–1181, 2018.

[13] J.E. Weimer, B. Sinopoli, and B.H. Krogh, "A relaxation approach to dynamic sensor selection in large-scale wireless networks," in *Int. Conf. on Distributed Computing Syst.*, 2008, pp. 501–506.

[14] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Trans. Signal Process.*, vol. 57[2], pp. 451–462, 2009.

[15] N.K. Dhingra, M.R. Jovanović, and Z.Q. Luo, "An ADMM algorithm for optimal sensor and actuator selection," in *Conf. on Decision and Contr.*, 2014, pp. 4039–4044.

[16] C. Rusu, J. Thompson, and N.M. Robertson, "Sensor scheduling with time, energy, and communication constraints," *IEEE Trans. Signal Process.*, vol. 66[2], pp. 528–539, 2018.

[17] M.P. Vitus, W. Zhang, A. Abate, J. Hu, and C.J. Tomlin, "On efficient sensor scheduling for linear dynamical systems," *Automatica*, vol. 48[10], pp. 2482–2493, 2012.

[18] 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.
[19] P. Singh, M. Chen, L. Carlone, S. Karaman, E. Frazzoli, and D. Hsu, "Supermodular mean squared error minimization for sensor scheduling in optimal Kalman filtering," in *American Contr. Conf.*, 2017, pp. 5787–5794.
[20] A. Jadbabaie, A. Olshevsky, and M. Siami, "Deterministic and randomized actuator scheduling with guaranteed performance bounds," 2018, arXiv:1805.00606v1.
[21] A. Schrijver, *Combinatorial Optimization*, Springer, 2003.
[22] M.L. Fisher, G.L. Nemhauser, and L.A. Wolsey, "An analysis of approximations for maximizing submodular set functions—ii," in *Polyhedral combinatorics*, pp. 73–87. Springer, 1978.
[23] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a monotone submodular function subject to a matroid constraint," *SIAM J. on Computing*, vol. 40[6], pp. 1740–1766, 2011.
[24] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," in *SIAM Symposium on Discrete Algorithms*, 2014, pp. 1134–1148.
[25] L.F.O. Chamon and A. Ribeiro, "Near-optimality of greedy set selection in the sampling of graph signals," in *Global Conf. on Signal and Inform. Process.*, 2016, pp. 1265–1269.
[26] L.F.O. Chamon and A. Ribeiro, "Approximate supermodularity bounds for experimental design," in *Advances in Neural Information Processing Systems*, 2017, pp. 5403–5412.
[27] A. Bian, J.M. Buhmann, A. Krause, and S. Tschiatschek, "Guarantees for greedy maximization of non-submodular functions with applications," in *Int. Conf. on Mach. Learning*, 2017.
[28] T. Summers and M. Kamgarpour, "Performance guarantees for greedy maximization of non-submodular set functions in systems and control," in *European Contr. Conf.*, 2019, pp. 2796–2801.
[29] D.P. Bertsekas, *Dynamic Programming and Optimal Control – Vol. I*, Athena Scientific, 2017.
[30] A. Krause and D. Golovin, "Submodular function maximization," in *Tractability: Practical Approaches to Hard Problems*. Cambridge University Press, 2014.
[31] F. Bach, "Learning with submodular functions: A convex optimization perspective," *Foundations and Trends in Machine Learning*, vol. 6[2-3], pp. 145–373, 2013.
[32] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher, "An analysis of approximations for maximizing submodular set functions—I," *Mathematical Programming*, vol. 14[1], pp. 265–294, 1978.
[33] T. Horel and Y. Singer, "Maximization of approximately submodular functions," in *Advances in Neural Information Processing Systems*, 2016, pp. 3045–3053.
[34] B. Lehmann, D. Lehmann, and N. Nisan, "Combinatorial auctions with decreasing marginal utilities," *Games and Economic Behavior*, vol. 55[2], pp. 270–296, 2006.
[35] O. Karaca and M. Kamgarpour, "Exploiting weak supermodularity for coalition-proof mechanisms," in *Conf. on Decision and Contr.*, 2018, pp. 1118–1123.
[36] L.F.O. Chamon, G.J. Pappas, and A. Ribeiro, "The mean square error in Kalman filtering sensor selection is approximately supermodular," in *Conf. on Decision and Contr.*, 2017, pp. 343–350.
[37] F. Muller, F. Seyler, and J.-L. Guyot, "Utilisation d'imagerie radar (ROS) JERS-1 pour l'obtention de réseaux de drainage. Exemple du Rio Negro (Amazonie)," in *Hydrological and Geochemical Processes in Large Scale River Basins*, 1999, http://www.ore-hybam.org/index.php/eng/Data/Cartography/Amazon-basin-hydrography.
[38] R.A. Horn and C.R. Johnson, *Matrix analysis*, Cambridge University Press, 2013.

APPENDIX

PROOF OF PROPOSITION 1

Proof. This result follows directly from the classical dynamic programming argument for the LQG by considering only the inputs in $\mathcal{S}$ (see, e.g., [29]). We display the derivations here for ease of reference. Explicitly, we proceed by backward induction, first defining the cost-to-go function

$$V_j(\mathcal{S}) = \min_{\mathcal{U}_j(\mathcal{S})} \mathbb{E}\left[\sum_{k=j}^{N-1}\left(\boldsymbol{x}_k^T\boldsymbol{Q}_k\boldsymbol{x}_k + \sum_{i\in\mathcal{S}\cap\mathcal{V}_k} r_{i,k}u_{i,k}^2\right) + \boldsymbol{x}_N^T\boldsymbol{Q}_N\boldsymbol{x}_N\right], \quad (35)$$

where $\mathcal{U}_j(\mathcal{S}) = \{u_{i,k} | i \in \mathcal{S}\cap\mathcal{V}_k\}_{k=j}^{N-1}$ is the subsequence of control actions from time j to the end of the control horizon. Notice due to the additivity of (35), it admits the equivalent recursive definition

$$V_j(\mathcal{S}) = \min_{\mathcal{U}_j(\mathcal{S})} \mathbb{E}\left[V_{j+1}(\mathcal{S}) + \boldsymbol{x}_j^T\boldsymbol{Q}_j\boldsymbol{x}_j + \sum_{i\in\mathcal{S}\cap\mathcal{V}_j} r_{i,j}u_{i,j}^2\right]. \quad (36)$$

To proceed, we postulate that (35) has the form

$$V_j(\mathcal{S}) = \boldsymbol{x}_j^T\boldsymbol{P}_j(\mathcal{S})\boldsymbol{x}_j + q_j, \quad (37)$$

for some $\boldsymbol{P}_j(\mathcal{S}) \succ 0$ and $q_j > 0$ and show that this is indeed the case by recursion. To do so, observe that for the base case $j = N$, (35) becomes $V_N(\mathcal{S}) = \boldsymbol{x}_N^T\boldsymbol{Q}_N\boldsymbol{x}_N$ and (37) holds trivially by taking $\boldsymbol{P}_N(\mathcal{S}) = \boldsymbol{Q}_N \succ 0$ and $q_N = 0$. Now, assume that (37) holds for $j+1$. Using the system dynamics (1) in (37), we can expand $V_j(\mathcal{S})$ to read

$$\begin{aligned} V_j(\mathcal{S}) = \min_{\mathcal{U}_j(\mathcal{S})} \mathbb{E}\Bigg[&\boldsymbol{x}_j^T\left(\boldsymbol{Q}_j + \boldsymbol{A}_j^T\boldsymbol{P}_{j+1}\boldsymbol{A}_j\right)\boldsymbol{x}_j \\ &+ \left(\sum_{i\in\mathcal{S}\cap\mathcal{V}_j}\boldsymbol{b}_{i,j}u_{i,j}\right)^T\boldsymbol{P}_{j+1}(\mathcal{S})\left(\sum_{i\in\mathcal{S}\cap\mathcal{V}_j}\boldsymbol{b}_{i,j}u_{i,j}\right) \\ &+ 2\boldsymbol{x}_j^T\boldsymbol{A}_j^T\boldsymbol{P}_{j+1}(\mathcal{S})\left(\sum_{i\in\mathcal{S}\cap\mathcal{V}_j}\boldsymbol{b}_{i,j}u_{i,j}\right) \\ &+ 2\boldsymbol{w}_j^T\boldsymbol{P}_{j+1}(\mathcal{S})\left(\sum_{i\in\mathcal{S}\cap\mathcal{V}_j}\boldsymbol{b}_{i,j}u_{i,j}\right) + 2\boldsymbol{x}_j^T\boldsymbol{A}_j^T\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{w}_j \\ &+ \boldsymbol{w}_j^T\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{w}_j + \sum_{i\in\mathcal{S}\cap\mathcal{V}_j} r_{i,j}u_{i,j}^2 + q_{j+1}\Bigg]. \end{aligned} \quad (38)$$

Recall that the expected value is taken over the process noises noise $\boldsymbol{w}_k$ and the initial state $\boldsymbol{x}_0$. Note that the terms linear in $\boldsymbol{w}_j$ vanish since it is zero-mean and that (38) is actually a quadratic optimization problem in the $\{u_{i,j}\}$. This is straightforward to see by rewriting (38) in matrix form:

$$\begin{aligned} V_j(\mathcal{S}) = \min_{\boldsymbol{u}_j}\ &\boldsymbol{u}_j^T\left(\boldsymbol{R}_j + \boldsymbol{B}_j^T\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{B}_j\right)\boldsymbol{u}_j \\ &+ 2\boldsymbol{x}_j^T\boldsymbol{A}_j^T\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{B}_j\boldsymbol{u}_j \\ &+ \boldsymbol{x}_j^T\left(\boldsymbol{Q}_j + \boldsymbol{A}_j^T\boldsymbol{P}_{j+1}\boldsymbol{A}_j\right)\boldsymbol{x}_j \\ &+ \mathrm{Tr}\left(\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{W}_j\right) + q_{j+1}, \end{aligned} \quad (39)$$

where $\boldsymbol{u}_j = [u_{i,j}]_{i\in\mathcal{S}\cap\mathcal{V}_j}$ is a $|\mathcal{S}\cap\mathcal{V}_j| \times 1$ vector that collects the control actions, $\boldsymbol{B}_j = [\boldsymbol{b}_{i,j}]_{i\in\mathcal{S}\cap\mathcal{V}_j}$ is an $n \times |\mathcal{S}\cap\mathcal{V}_j|$ matrix whose columns contain the input vectors corresponding to each control action, $\boldsymbol{R}_j = \mathrm{diag}(r_{i,j})$, and we used

the that $\mathbb{E}\left[\boldsymbol{w}_j^T \boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{w}_j\right] = \operatorname{Tr}\left(\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{W}_j\right)$. Observe that (39) is classical LQR problem, up to constant terms [29]. Its minimum is then of the form (37) with

$$\boldsymbol{P}_j(\mathcal{S}) = \boldsymbol{Q}_j + \boldsymbol{A}_j^T \left(\boldsymbol{P}_{j+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S} \cap \mathcal{V}_j} r_{i,j}^{-1} \boldsymbol{b}_{i,j} \boldsymbol{b}_{i,j}^T \right)^{-1} \boldsymbol{A}_j,$$

and $q_j = \operatorname{Tr}\left(\boldsymbol{P}_{j+1}(\mathcal{S})\boldsymbol{W}_j\right) + q_{j+1}$. Unrolling the recursion and using the fact that $\mathbb{E}[\boldsymbol{x}_0 \boldsymbol{x}_0^T] = \boldsymbol{\Sigma}_0$ yields the result in (3). ■

Proof of Proposition 7

Proof. Start by defining

$$J_k(\mathcal{S}) = \operatorname{Tr}\left[\boldsymbol{\Pi}_{k-1} \boldsymbol{P}_k(\mathcal{S})\right] \tag{40}$$

with $\boldsymbol{\Pi}_{-1} = \boldsymbol{\Sigma}_0$ and $\boldsymbol{\Pi}_k = \boldsymbol{W}_k$ for $k = 0, \dots, N-2$ and from (3), notice that the objective J of (PI) can be written as

$$J(\mathcal{S}) = \sum_{k=0}^{N-1} J_k(\mathcal{S}) + \operatorname{Tr}\left[\boldsymbol{W}_{N-1} \boldsymbol{Q}_N\right] - V^\star(\emptyset). \tag{41}$$

Hence, using the non-negative affine combination property in Proposition 5, we can ignore the terms constant in $\mathcal{S}$ and lower bound the approximate supermodularity of J in terms of the approximate supermodularity of its components. Explicitly, if J_k in (40) is α_k-supermodular, then J is $\min(\alpha_k)$-supermodular.

We can further reduce the problem by using (4) to get

$$\begin{aligned} J_k(\mathcal{S}) = \operatorname{Tr}\left[\boldsymbol{H}_k \left(\boldsymbol{P}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S} \cap \mathcal{V}_k} r_{i,k}^{-1} \boldsymbol{b}_{i,k} \boldsymbol{b}_{i,k}^T \right)^{-1} \right] \\ + \operatorname{Tr}\left[\boldsymbol{\Pi}_{k-1} \boldsymbol{Q}_k\right]. \end{aligned} \tag{42}$$

where we used the circular shift property of the trace to get $\boldsymbol{H}_k = \boldsymbol{A}_k \boldsymbol{\Pi}_{k-1} \boldsymbol{A}_k^T$. Thus, applying Proposition 5 again, we can ignore the constant term in (42) and restrict to studying the α-supermodularity of

$$\bar{J}_k(\mathcal{S}) = \operatorname{Tr}\left[\boldsymbol{H}_k \left(\boldsymbol{P}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S} \cap \mathcal{V}_k} r_{i,k}^{-1} \boldsymbol{b}_{i,k} \boldsymbol{b}_{i,k}^T \right)^{-1} \right]. \tag{43}$$

To proceed, notice that $\boldsymbol{H}_k \succ 0$ since $\boldsymbol{\Pi}_k \succ 0$ and $\boldsymbol{A}_k$ is full rank for all k. Hence, it has a unique positive definite square root $\boldsymbol{H}^{1/2} \succ 0$ such that $\boldsymbol{H} = \boldsymbol{H}^{1/2}\boldsymbol{H}^{1/2}$ [38]. Deploying the circular shift property of the trace and the invertibility of $\boldsymbol{H}^{1/2}$, (43) can be written as

$$\bar{J}_k(\mathcal{S}) = \operatorname{Tr}\left[\left(\tilde{\boldsymbol{P}}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S} \cap \mathcal{V}_k} r_{i,k}^{-1} \tilde{\boldsymbol{b}}_{i,k} \tilde{\boldsymbol{b}}_{i,k}^T \right)^{-1} \right]. \tag{44}$$

with $\tilde{\boldsymbol{P}}_{k+1}(\mathcal{S}) = \boldsymbol{H}_k^{1/2} \boldsymbol{P}_{k+1}(\mathcal{S}) \boldsymbol{H}_k^{1/2}$ and $\tilde{\boldsymbol{b}}_{i,k} = \boldsymbol{H}_k^{-1/2} \boldsymbol{b}_{i,k}$. The function $\bar{J}_k$ in (44) has a form that allows us to leverage the following result from [5, Thm. 3], which we reproduce here for ease of reference:

Lemma 1. *Let $h : 2^{\mathcal{E}} \to \mathbb{R}$ be the set trace function*

$$h(\mathcal{A}) = \operatorname{Tr}\left[\left(\boldsymbol{M}_\emptyset + \sum_{i \in \mathcal{A}} \boldsymbol{M}_i \right)^{-1} \right], \tag{45}$$

where $\mathcal{A} \subseteq \mathcal{E}$, $\boldsymbol{M}_\emptyset \succ 0$, and $\boldsymbol{M}_i \succeq 0$ for all $i \in \mathcal{E}$. Then, h is (i) monotonically decreasing and (ii) α-supermodular with

$$\alpha \geq \frac{\lambda_{\min}\left[\boldsymbol{M}_\emptyset\right]}{\lambda_{\max}\left[\boldsymbol{M}_\emptyset + \sum_{i \in \mathcal{V}} \boldsymbol{M}_i\right]} > 0. \tag{46}$$

Comparing (44) and (45), we can bound the α_k for J_k as

$$\alpha_k \geq \min_{\mathcal{S} \subseteq \overline{\mathcal{V}}} \frac{\lambda_{\min}\left[\tilde{\boldsymbol{P}}_{k+1}^{-1}(\mathcal{S})\right]}{\lambda_{\max}\left[\tilde{\boldsymbol{P}}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{V}_k} r_{i,k}^{-1} \tilde{\boldsymbol{b}}_{i,k} \tilde{\boldsymbol{b}}_{i,k}^T\right]}. \tag{47}$$

Nevertheless, the bound in (47) depends on the choice of $\mathcal{S}$. To obtain a closed-form expression, we use the following proposition:

Proposition 8. *For any $\mathcal{S} \subseteq \overline{\mathcal{V}}$, it holds that*

$$\tilde{\boldsymbol{P}}_k(\overline{\mathcal{V}}) \preceq \tilde{\boldsymbol{P}}_k(\mathcal{S}) \preceq \tilde{\boldsymbol{P}}_k(\emptyset), \text{ for } k = 1, \dots, N-1. \tag{48}$$

Using (48) in (47) and the fact that matrix inversion is operator antitone yields the bound in (28). ■

All that remains is therefore to prove Proposition 8.

Proof of Proposition 8. Start by noticing that since $\boldsymbol{H}^{1/2}$ is full rank, it is enough to establish (8) directly for $\boldsymbol{P}_k$. Indeed, $\boldsymbol{A} \preceq \boldsymbol{B} \Leftrightarrow \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} \leq \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x}$ for all $\boldsymbol{x} \in \mathbb{R}^n$ and for $\boldsymbol{M} \succ 0$, this is equivalent to taking $\boldsymbol{x} = \boldsymbol{M}\boldsymbol{y}$ for all $\boldsymbol{y} \in \mathbb{R}^n$.

We prove both inequalities by recursion. For the upper bound in (48), note from (4) that $\tilde{\boldsymbol{P}}_k$ can be increased by using no actuators at instant k. Formally, for any choice of $\mathcal{S} \subseteq \overline{\mathcal{V}}$,

$$\begin{aligned} \boldsymbol{P}_k(\mathcal{S}) &= \boldsymbol{Q}_k + \boldsymbol{A}_k^T \left(\boldsymbol{P}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S} \cap \mathcal{V}_k} r_{i,k}^{-1} \boldsymbol{b}_{i,k} \boldsymbol{b}_{i,k}^T \right)^{-1} \boldsymbol{A}_k \\ &\preceq \boldsymbol{Q}_k + \boldsymbol{A}_k^T \boldsymbol{P}_{k+1}(\mathcal{S}) \boldsymbol{A}_k = \boldsymbol{P}_k(\mathcal{S} \setminus \mathcal{V}_k). \end{aligned}$$

Additionally, if $\bar{\boldsymbol{P}}_{k+1} \succeq \boldsymbol{P}_{k+1}(\mathcal{S})$ for all $\mathcal{S} \subseteq \overline{\mathcal{V}}$, it holds that

$$\boldsymbol{P}_k(\mathcal{S}) \preceq \boldsymbol{Q}_k + \boldsymbol{A}_k^T \bar{\boldsymbol{P}}_{k+1} \boldsymbol{A}_k \triangleq \bar{\boldsymbol{P}}_k. \tag{49}$$

Starting from $\boldsymbol{P}_N \preceq \boldsymbol{Q}_N \triangleq \bar{\boldsymbol{P}}_N$, we obtain that $\boldsymbol{P}_k$ is upper bounded by taking $\mathcal{S} = \emptyset$, i.e., using no actuators.

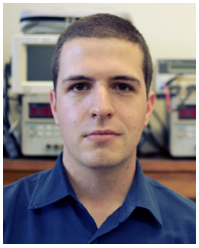
The lower bound is obtained in a similar fashion by using all possible actuators. Explicitly,

$$\begin{aligned} \boldsymbol{P}_k(\mathcal{S}) &= \boldsymbol{Q}_k + \boldsymbol{A}_k^T \left(\boldsymbol{P}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{S} \cap \mathcal{V}_k} r_{i,k}^{-1} \boldsymbol{b}_{i,k} \boldsymbol{b}_{i,k}^T \right)^{-1} \boldsymbol{A}_k \\ &\succeq \boldsymbol{Q}_k + \boldsymbol{A}_k^T \left(\boldsymbol{P}_{k+1}^{-1}(\mathcal{S}) + \sum_{i \in \mathcal{V}_k} r_{i,k}^{-1} \boldsymbol{b}_{i,k} \boldsymbol{b}_{i,k}^T \right)^{-1} \boldsymbol{A}_k \\ &= \boldsymbol{P}_k(\mathcal{S} \cup \mathcal{V}_k). \end{aligned}$$

Moreover, if $\underline{\boldsymbol{P}}_{k+1} \preceq \boldsymbol{P}_{k+1}(\mathcal{S})$ for all $\mathcal{S} \subseteq \overline{\mathcal{V}}$, we have that

$$\boldsymbol{P}_k(\mathcal{S}) \succeq \boldsymbol{Q}_k + \boldsymbol{A}_k^T \left(\underline{\boldsymbol{P}}_{k+1}^{-1} + \sum_{i \in \mathcal{V}_k} r_{i,k}^{-1} \boldsymbol{b}_{i,k} \boldsymbol{b}_{i,k}^T \right)^{-1} \boldsymbol{A}_k \triangleq \underline{\boldsymbol{P}}_k. \tag{50}$$

Starting from $\boldsymbol{P}_N \succeq \boldsymbol{Q}_N \triangleq \underline{\boldsymbol{P}}_N$ yields the lower bound in (48). ■



Luiz F. O. Chamon (S'12) received the B.Sc. and M.Sc. degree in electrical engineering from the University of São Paulo, São Paulo, Brazil, in 2011 and 2015. In 2009, he was an undergraduate exchange student at the Masters in Acoustics of the École Centrale de Lyon, Lyon, France. He is currently working toward the Ph.D. degree in electrical and systems engineering at the University of Pennsylvania (Penn), Philadelphia. In 2009, he was an Assistant Instructor and Consultant on nondestructive testing at INSACAST Formation Continue. From 2010 to 2014, he worked as a Signal Processing and Statistical Consultant on a project with EMBRAER. His research interest include signal processing, optimization, statistics, and control.

Alexandre Amice received the B.S.E. in Electrical Engineering and Mathematics and the M.S.E. in Robotics from the University of Pennsylvania, Pennsylvania, USA in 2020. His research interests include optimization, dynamical systems, and control.

Alejandro Ribeiro received the B.Sc. degree in electrical engineering from the Universidad de la Republica Oriental del Uruguay, Montevideo, in 1998 and the M.Sc. and Ph.D. degree in electrical engineering from the Department of Electrical and Computer Engineering, the University of Minnesota, Minneapolis in 2005 and 2007.

From 1998 to 2003, he was a member of the technical staff at Bellsouth Montevideo. After his M.Sc. and Ph.D studies, in 2008 he joined the University of Pennsylvania (Penn), Philadelphia, where he is currently the Rosenbluth Associate Professor at the Department of Electrical and Systems Engineering. His research interests are in the applications of statistical signal processing to the study of networks and networked phenomena. His focus is on structured representations of networked data structures, graph signal processing, network optimization, robot teams, and networked control.

Dr. Ribeiro received the 2014 O. Hugo Schuck best paper award, and paper awards at CDC 2017, 2016 SSP Workshop, 2016 SAM Workshop, 2015 Asilomar SSC Conference, ACC 2013, ICASSP 2006, and ICASSP 2005. His teaching has been recognized with the 2017 Lindback award for distinguished teaching and the 2012 S. Reid Warren, Jr. Award presented by Penn's undergraduate student body for outstanding teaching. Dr. Ribeiro is a Fulbright scholar class of 2003 and a Penn Fellow class of 2015.