Robust and Adaptive Sequential Submodular Optimization

Vasileios Tzoumas,[⋆] *Member, IEEE,* Ali Jadbabaie,[†] *Fellow, IEEE,* George J. Pappas,[‡] *Fellow, IEEE.*

***Abstract*—Emerging applications of control, estimation, and machine learning, from target tracking to decentralized model fitting, pose resource constraints that limit which of the available sensors, actuators, or data can be simultaneously used across time. Therefore, many researchers have proposed solutions within discrete optimization frameworks where the optimization is performed over finite sets. By exploiting notions of discrete convexity, such as submodularity, the researchers have been able to provide scalable algorithms with provable suboptimality bounds. In this paper, we consider such problems but in adversarial environments, where in every step a number of the chosen elements in the optimization is removed due to failures/attacks. Specifically, we consider for the first time a sequential version of the problem that allows us to observe the failures and adapt, while the attacker also adapts to our response. We call the novel problem *Robust Sequential submodular Maximization* (RSM). Generally, the problem is computationally hard and no scalable algorithm is known for its solution. However, in this paper we propose *Robust and Adaptive Maximization* (RAM), the first scalable algorithm. RAM runs in an online fashion, adapting in every step to the history of failures. Also, it guarantees a near-optimal performance, even against any number of failures among the used elements. Particularly, RAM has both provable per-instance a priori bounds and tight and/or optimal a posteriori bounds. Finally, we demonstrate RAM's near-optimality in simulations across various application scenarios, along with its robustness against several failure types, from worst-case to random.**

I. Introduction

Control, estimation, and machine learning applications of the Internet of Things (IoT) and autonomous robots [1] require the sequential optimization of systems in scenarios such as:

• *Sensor scheduling:* An unmanned aerial vehicle (UAV) is assisted for its navigation by on-board and on-ground sensors. Ideally, the UAV would use all available sensors for navigation. However, limited on-board capacity for measurement-processing necessitates a sequential sensor scheduling problem [2]: at each time step, which few sensors should be used for the UAV to effectively navigate itself?

• *Target tracking:* A wireless sensor network (WSN) is designated to monitor a mobile target. Limited battery power necessitates a sequential sensor activation problem [3]: at each time step, which few sensors should be activated for the WSN to effectively track the target?

• *Decentralized model fitting:* A team of mobile robots collects data to learn the model of an unknown environmental process. The data are transmitted to a fusion center, performing the statistical analysis. Ideally, all robots would transmit their data to the center at the same time. But instead, communication bandwidth constraints necessitate a sequential transmission problem [4]: at each time step, which few robots should transmit their data for the center to effectively learn the model?

Similar applications of sensor and data scheduling, but also of actuator scheduling as well as infrastructure design are studied in [5]–[16]. Particularly, all applications above require the sequential selection of a few elements, among a finite set of available ones, to optimize performance across multiple steps subject to resource constraints. For example, the target tracking application above requires the sequential activation of a few sensors across the WSN, to optimize an estimation error subject to power constraints. Importantly, the activated sensors may vary in time, since each sensor may measure different parts of the target's state (e.g., some sensors may measure only position, others only speed). Formally, all above applications motivate the sequential optimization problem[1]

$$\begin{aligned} \max_{\mathcal{A}_1\subseteq\mathcal{V}_1} \cdots \max_{\mathcal{A}_T\subseteq\mathcal{V}_T} \;\; & f(\mathcal{A}_1,\ldots,\mathcal{A}_T),\\ \text{s.t.} \quad & |\mathcal{A}_t| = \alpha_t, \quad t = 1,\ldots,T, \end{aligned} \tag{1}$$

where T is a given horizon; $\mathcal{V}_t$ is a given finite set of available elements to choose from at t such that $\mathcal{V}_t \cap \mathcal{V}_{t'} = \emptyset$ for all $t,t' = 1,\ldots,T$;[2] $f : 2^{\mathcal{V}_1} \cup \cdots \cup 2^{\mathcal{V}_T} \mapsto \mathbb{R}$ is a given objective function; α_t is a given cardinality constraint, capturing the resource constraints at t; and $\mathcal{A}_t$ are the chosen elements at t, resulting from the solution of eq. (1). Notably, in all above applications, and [5]–[16], f is non-decreasing, and without loss of generality one may consider $f(\emptyset) = 0$. For example, in [11], f is the trace of the inverse of the controllability Gramian, which captures the average control effort for driving the system; and in [8], f is the logdet of the error covariance of the minimum mean square batch-state estimator. Specifically, in [8], f is also submodular, a diminishing returns property that captures the intuition that a sensor's contribution to f's value diminishes when more sensors are activated already.

[⋆]Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109, USA. At the time the paper was accepted for publication: Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. vtzoumas@umich.edu

[†]Institute for Data, Systems and Society, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. jadbabai@mit.edu

[‡]Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104 USA. pappasg@seas.upenn.edu

The work was supported by the AFOSR Complex Networks program, and by the ARL CRA DCIST W911NF-17-2-0181 program.

[1]Calligraphic fonts denote finite discrete sets (e.g., $\mathcal{A}$). $2^{\mathcal{A}}$ denotes $\mathcal{A}$'s power set. $|\mathcal{A}|$ its cardinality. $\mathcal{A}\setminus\mathcal{B}$ denotes set difference: the elements in $\mathcal{A}$ not in $\mathcal{B}$. Given a set function $f : 2^{\mathcal{V}_1} \cup \cdots \cup 2^{\mathcal{V}_T} \mapsto \mathbb{R}$, and $\mathcal{A}_1 \subseteq \mathcal{V}_1, \ldots, \mathcal{A}_t \subseteq \mathcal{V}_t$, for some positive integer $t \leq T$, the $f(\mathcal{A}_1,\ldots,\mathcal{A}_t)$ denotes $f(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_t \cup \emptyset \cup \cdots \cup \emptyset)$ where the $\emptyset$ is repeated $T-t$ times, and $\emptyset$ denotes the empty set. $\mathbb{R}$ denotes the set of real numbers.

[2]Even if the elements in $\mathcal{V}_1,\ldots,\mathcal{V}_T$ correspond to the same system modules, e.g., sensors, the elements among different $\mathcal{V}_t$ are differentiated because they are chosen at different times. For example, consider the case where $T = 2$, and two sensors s_1 and s_2 are available to be chosen at each t; then, by denoting with $s_{i,t}$ that sensor i is available to be chosen at t, it is $\mathcal{V}_1 = \{s_{1,1}, s_{2,1}\}$ and $\mathcal{V}_2 = \{s_{1,2}, s_{2,2}\}$, and, naturally, $\mathcal{V}_1 \cap \mathcal{V}_2 = \emptyset$.

Although the problem in eq. (1) is computationally hard, efficient algorithms have been proposed for its solution: when f is monotone and submodular, then eq. (1) is NP-hard [17] and the greedy algorithm in [18, Section 4] guarantees a constant suboptimality bound across all problem instances; and when f is only monotone, then eq. (1) is inapproximable (no polynomial time algorithm guarantees a constant bound across all instances) [19], [20] but the greedy algorithm in [18] guarantees per-instance bounds instead [21]–[23].

In this paper, however, we shift focus to a novel reformulation of eq. (1) that is robust against failures/attacks. Particularly, in all above applications, at any time t, actuators can be cyber-attacked [24], sensors can malfunction [25], and communication channels can be blocked [4], all resulting to denial-of-service (DoS) failures, in the sense that the actuators, sensors, channels, etc. will shut down (stop working), at least temporarily. Hence, in such failure-prone and adversarial scenarios, eq. (1) may fail to protect any of the above applications, since it ignores the possibility of DoS failures. Thus, towards guaranteed protection, a robust reformulation becomes necessary *that can both adapt to the history of incured failures and account for future ones*.

Therefore, in this paper we introduce a novel robust optimization framework, named *Robust Sequential submodular Maximization* (`RSM`), that goes beyond the failure-free eq. (1) and accounts for DoS failures/attacks. Specifically, we define `RSM` as the following robust reformulation of eq. (1):

`RSM` problem:

$$\max_{\mathcal{A}_1\subseteq\mathcal{V}_1}\ \min_{\mathcal{B}_1\subseteq\mathcal{A}_1}\cdots\max_{\mathcal{A}_T\subseteq\mathcal{V}_T}\ \min_{\mathcal{B}_T\subseteq\mathcal{A}_T}\ f(\mathcal{A}_1\setminus\mathcal{B}_1,\ldots,\mathcal{A}_T\setminus\mathcal{B}_T),$$
$$\text{s.t.}\quad |\mathcal{A}_t|=\alpha_t,\ \ |\mathcal{B}_t|\leq\beta_t,\quad t=1,\ldots,T. \tag{2}$$

where β_t is a given number of possible failures (generally, $\beta_t\in[0,\alpha_t]$); and $\mathcal{B}_t$ is the failure against $\mathcal{A}_t$.

By solving `RSM`, our goal is to maximize f despite worst-case failures that occur at each maximization step, as captured by the intermediate/subsequent minimization steps. Evidently, since `RSM` considers worst-case failures, it is suitable when there is no prior on the failure mechanism, or when protection against worst-case failures is essential, such as in safety-critical target tracking and costly experiment designs.

`RSM` can be interpreted as a T-stage perfect information sequential game between a "maximization" player (defender) and a "minimization" player (attacker) [26, Chapter 4]. The defender starts the game and the players act sequentially, having perfect knowledge of each others' actions: at each t, the defender selects an $\mathcal{A}_t$, and then the attacker responds with a worst-case removal $\mathcal{B}_t$ from $\mathcal{A}_t$, while both players account for the history of all actions up to $t-1$. In this context, the defender finds an optimal sequence $\mathcal{A}_1,\ldots,\mathcal{A}_T$ by accounting at each t (i) for the history of responses $\mathcal{B}_1,\ldots,\mathcal{B}_{t-1}$, (ii) for the subsequent response $\mathcal{B}_t$, and (iii) for all remaining future responses $\mathcal{B}_{t+1},\ldots,\mathcal{B}_T$. This is an additional computational challenge in comparison to the failure-free eq. (1), which is already computationally hard.

No scalable algorithms exists for `RSM`. *In this paper, to provide the first scalable algorithm, we develop an adaptive algorithm* that at each t accounts only (i) for the history of responses up to $t-1$ and (ii) for the subsequent response $\mathcal{B}_t$ (but not for the remaining future responses up to $t=T$), and as a result is scalable, but which still can guarantee a performance close to the optimal.

Related work in combinatorial optimization. The majority of the related work has focused on the failure-free eq. (1), when f is either monotone and submodular or only monotone. In more detail, Fisher et al. [18] focused on f being monotone and submodular, and proposed offline and online greedy algorithms that both guarantee the constant $1/2$ suboptimality bound. Similarly, Conforti and Cornuéjols [27], Iyer et al. [28], and Sviridenko et al. [23] focused again on f being monotone and sumodular but provided instead per-instance, curvature-depended bounds. The bounds generally tighten the ones in [18]. Finally, Krause et al. [29], Das and Kempe [21], Wang et al. [22], and Sviridenko et al. [23] (see also the earlier [30]) focused on f being only monotone, and proved per-instance, curvature-depended bounds for the greedy algorithms in [18], using notions of curvature —also referred to as "submodularity ratio"— they introduced.

Recent work has also studied failure-robust reformulations of eq. (1), typically per `RSM`'s framework *but only for $T=1$, where no adaptiveness is required*. Specifically, when f is monotone and submodular, Orlin et al. [31] and Bogunovic et al. [32] provided greedy algorithms with constant suboptimality bounds. However, the algorithms are valid only for limited numbers of failures (for $\beta_1\leq\sqrt{\alpha_1}$ in [31] and $\beta_1\leq\alpha_1/(\log\alpha_1)^3$ in [32]). In contrast, Tzoumas et al. [33] provided a greedy algorithm with per-instance bounds for any number of failures (β_1 can take any value in $[0,\alpha_1]$). Also, Rahmattalabi et al. [34] developed a mixed-integer linear program approach for a locations monitoring problem. More recently, Tzoumas et al. [35] and Bogunovic et al. [36] extended the previous works on the $T=1$ case by focusing on f being only monotone, and proved per-instance, curvature-dependent bounds for the algorithm introduced in [33]. In more detail, Bogunovic et al. [36] focuses on cardinality constraints, whereas Tzoumas et al. [35] on the more general matroid constraints, but, still, for the case where $T=1$. The latter framework enabled applications of failure-robust multi-robot robot planning, and particularly of active information gathering [37] and target tracking [38]. Other relevant work is that of Mitrovic et al. [39], where a memoryless failure-robust reformulation of eq. (1) is considered, instead of the sequential framework of `RSM`, which takes into account the history of past selections/failures. Finally, Mirzasoleiman et al. [40] and Kazemi et al. [41] adopted a robust optimization framework against non worst-case failures, in contrast to `RSM` which is against worst-case failures.

All in all, in comparison to all prior research, in this paper we analyze `RSM`'s multistep case $T>1$ for the first time, and consider adaptive algorithms.

Related work in control. In the robust/secure control literature, various approaches have been proposed towards fault-tolerant control, secure control, as well as secure state estimation, against random failures, data injection and DoS failures/attacks [42]–[61]. In contrast to `RSM`'s resource-constrained framework, [42]–[61] focus in resource abundant environments where all sensors and actuators stay always active under normal operation. For example, [59]–[61] focus on DoS failures/attacks from the perspective of packet loss and intermittent network connectivity, which can result to system destabilization. Generally, [42]–[61] focus on failure/attack detection and identification, and/or secure estimator/controller design, *instead of the adaptive activation of a few sensors/actuators against worst-case DoS failures/attacks per* RSM.

Contributions. We introduce the novel RSM problem of robust sequential maximization against DoS failures/attacks. We develop the first scalable algorithm, named *Robust and Adaptive Maximization* (RAM), that has the properties:

• *Adaptiveness:* At each time $t = 1, 2, \ldots$, RAM selects a robust solution $\mathcal{A}_t$ in an online fashion, accounting for the history of failures $\mathcal{B}_1, \ldots, \mathcal{B}_{t-1}$ and of actions $\mathcal{A}_1, \ldots, \mathcal{A}_{t-1}$, as well as, for all possible failures at t from $\mathcal{A}_t$.

• *System-wide robustness:* RAM is valid for any number of failures; that is, for any $\beta_t \in [0, \alpha_t]$, $t = 1, 2, \ldots$.

• *Polynomial running time:* RAM has the same order of running time as the polynomial time greedy algorithm proposed in [18, Section 4] for the failure-free eq. (1).

• *Provable approximation performance:* RAM has provable per-instance suboptimality bounds that quantify RAM's near-optimality at each problem instance at hand.[3] Particularly, we provide both a priori and a posteriori per-instance bounds. The a priori bounds quantify RAM's near-optimality before RAM has run. In contrast, the a posteriori bounds are computable online (as RAM runs), once the failures at each current step have been observed. The a posteriori bounds are tight and/or optimal.[4] Finally, we present approximations of the a posteriori bounds that are computable before each failure occurs. To quantify the bounds, we use curvature notions by Conforti and Conruéjols [27], for monotone and submodular functions, and Sviridenko et al. [23], for monotone functions.

We demonstrate RAM's effectiveness in applications of sensor scheduling, and of target tracking with wireless sensor networks. We present a Monte Carlo analysis, where we vary the failure types from worst-case to greedily and randomly selected failures, and compare RAM against a brute-force optimal algorithm (viable only for small-scale instances), the greedy algorithm in [18], and a random algorithm. In the results, we observe RAM's near-optimality against worst-case failures, its robustness against non worst-case failures, and its superior performance against the compared algorithms.

Comparison with the preliminary results in [62], which coincides with preprint [63]: This paper extents the results in [62], considers new simulations, and includes the proofs that were all omitted from [62]. Particularly, most of the technical results herein, including Theorem 13, Theorem 14, Corollary 21, and Algorithm 3, are novel and have not been

Algorithm 1: Robust adaptive maximization (RAM).

Input: RAM receives the inputs:
- *Offline*: integer T; function $f : 2^{\mathcal{V}_1} \cup \cdots \cup 2^{\mathcal{V}_T} \mapsto \mathbb{R}$ such that f is non-decreasing and $f(\emptyset) = 0$; integers α_t, β_t such that $0 \leq \beta_t \leq \alpha_t \leq |\mathcal{V}_t|$, for all $t = 1, \ldots, T$;
- *Online*: at each $t = 2, \ldots, T$, observed removal $\mathcal{B}_{t-1}$ from RAM's output $\mathcal{A}_{t-1}$.

Output: At each step $t = 1, \ldots, T$, set $\mathcal{A}_t$.

```
1:  for all $t = 1, \ldots, T$ do
2:    $\mathcal{S}_{t,1} \leftarrow \emptyset$;   $\mathcal{S}_{t,2} \leftarrow \emptyset$;
3:    Sort elements in $\mathcal{V}_t$ s.t. $\mathcal{V}_t \equiv \{v_{t,1}, \ldots, v_{t,|\mathcal{V}_t|}\}$ and
      $f(v_{t,1}) \geq \ldots \geq f(v_{t,|\mathcal{V}_t|})$;
4:    $\mathcal{S}_{t,1} \leftarrow \{v_{t,1}, \ldots, v_{t,\beta_t}\}$;
5:    while $|\mathcal{S}_{t,2}| < \alpha_t - \beta_t$ do
6:      $x \in \arg\max_{y \in \mathcal{V}_t \setminus (\mathcal{S}_{t,1} \cup \mathcal{S}_{t,2})} f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus$
        $\mathcal{B}_{t-1}, \mathcal{S}_{t,2} \cup \{y\})$;
7:      $\mathcal{S}_{t,2} \leftarrow \{x\} \cup \mathcal{S}_{t,2}$;
8:    end while
9:    $\mathcal{A}_t \leftarrow \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$.
10: end for
```

previously published. Also, the simulation scenarios are new and include a sensitivity analysis of RAM against various failure types (in [62] we tested RAM only against worst-case failures, and in different scenarios). Finally, all proofs in [62] were omitted and are now included here.

Organization of the rest of the paper. Section II presents RAM, and quantifies its minimal running time. Section III presents RAM's suboptimality bounds. Section IV presents RAM's numerical evaluations. Section V concludes the paper. All proofs are found in the appendix.

II. An Adaptive Algorithm: RAM

We present RAM, the first scalable algorithm for RSM, formulated in eq. (2). RAM's pseudo-code is given in Algorithm 1. Below, we first give an intuitive description of RAM, and then a step-by-step description. Also, we quantify its running time. RAM's suboptimality bounds are given in Section III.

A. Intuitive description

RSM aims to maximize f through a sequence of steps despite compromises to each step. Specifically, at each $t = 1, 2, \ldots$, RSM selects an $\mathcal{A}_t$ towards a maximal f despite the fact that $\mathcal{A}_t$ will be compromised by a worst-case removal $\mathcal{B}_t$, resulting to f being evaluated at $\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_T \setminus \mathcal{B}_T$ instead of $\mathcal{A}_1, \ldots, \mathcal{A}_T$. In this context, RAM aims to achieve RSM's goal by selecting $\mathcal{A}_t$ as the union of two sets $\mathcal{S}_{t,1}$, and $\mathcal{S}_{t,2}$ (RAM's line 9), whose role we describe intuitively below:

a) $\mathcal{S}_{t,1}$ approximates (aims to guess the) worst-case removal from $\mathcal{A}_t$: With $\mathcal{S}_{t,1}$, RAM aims to capture the worst-case removal of β_t elements from $\mathcal{A}_t$. Intuitively, $\mathcal{S}_{t,1}$ is aimed to act as a "bait" to a worst-case attacker that selects the best β_t elements to remove from $\mathcal{A}_t$ at time t (*best* with respect to their contribution towards RSM's goal). RAM aims to approximate them by letting $\mathcal{S}_{t,1}$ be the set of β_t elements

[3]Similarly to eq. (1), RSM is generally inapproximable: no polynomial time algorithm guarantees a constant suboptimality bound across all problem instances. For example, it is inapproximable for fundamental applications in control and machine learning such as *sensor selection for optimal Kalman filtering* [20], and *feature selection for sparse model fitting* [19]. Thus, in this paper we focus our analysis in per-instance suboptimality bounds.

[4]A suboptimality bound is called *optimal* when it is the tightest achievable bound among all polynomial time algorithms, given a worst-case family of f.

with the largest marginal contributions to f (RAM's lines 3-4). As such, each $\mathcal{S}_{t,1}$ is independent of the history of actual removals $\mathcal{B}_1, \ldots, \mathcal{B}_{t-1}$ and can be computed offline, before any of the $\mathcal{B}_1, \ldots, \mathcal{B}_T$ has been realized. In contrast, $\mathcal{S}_{t,2}$ can only be computed online, as we describe below.

b) $\mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$ approximates optimal solution to RSM's t-th step: To complete $\mathcal{A}_t$'s construction, RAM needs to select a set $\mathcal{S}_{t,2}$ of $\alpha_t - \beta_t$ elements (since $|\mathcal{A}_t| = \alpha_t$ and $|\mathcal{S}_{t,1}| = \beta_t$), and return $\mathcal{A}_t = \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$ (RAM's line 9). Assuming $\mathcal{S}_{t,1}$'s removal from $\mathcal{A}_t$, for $\mathcal{A}_t$ to be an optimal solution to RSM's t-th maximization step, RAM 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, this problem is NP-hard [17]. Thereby, RAM approximates such a best set, using the greedy procedure in RAM's lines 5-8. Particularly, RAM's line 6 adapts $\mathcal{S}_{t,2}$ to the history of removals $\mathcal{B}_1, \ldots, \mathcal{B}_{t-1}$ and selections $\mathcal{A}_1, \ldots, \mathcal{A}_{t-1}$, since it constructs $\mathcal{S}_{t,2}$ given $\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus \mathcal{B}_{t-1}$. As such, each $\mathcal{S}_{t,2}$, in contrast to $\mathcal{S}_{t,1}$, can be computed only online, only once the history of removals $\mathcal{B}_1, \ldots, \mathcal{B}_{t-1}$ has been realized.

Overall, RAM adaptively constructs an $\mathcal{A}_t$ to approximate an optimal solution to RSM's t-th maximization step.

Remark 1 (Further intuition on why $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$ are selected as in RAM)**.** *We first discuss why RAM (i) selects $\mathcal{A}_t$ as the union of $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$, and (ii) selects $\mathcal{S}_{t,2}$ as a greedily picked subset of $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$. We then focus on $\mathcal{S}_{t,1}$.*

If $\mathcal{S}_{t,1}$ guesses correctly the removal $\mathcal{B}_t$ from $\mathcal{A}_t = \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$, then all elements in $\mathcal{S}_{t,2}$ remain intact ($\mathcal{A}_t \setminus \mathcal{B}_t = \mathcal{S}_{t,2}$). Therefore, since $\mathcal{S}_{t,2}$ has been selected using the greedy algorithm in RAM's lines 5-8, which is an optimal approximation algorithm for maximizing monotone functions subject to cardinality constraints [23],[5] $\mathcal{A}_t$ is an optimal approximation to RSM's t-th maximization step. This explains why RAM selects $\mathcal{A}_t$ as the union of two sets ($\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$), and why RAM selects $\mathcal{S}_{t,2}$ greedily from $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$ given $\mathcal{S}_{t,1}$. Generally, if $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ guess $\mathcal{B}_1, \ldots, \mathcal{B}_T$ correctly, then RSM becomes equivalent to the attack-free eq. (1)*, and RAM becomes equivalent to the optimal greedy algorithm for eq.* (1)*.*

But if $\mathcal{S}_{t,1}$ guesses incorrectly $\mathcal{B}_t$, then some of $\mathcal{S}_{t,1}$'s elements will survive, and, instead, some of $\mathcal{S}_{t,2}$'s elements will be removed. The question arising now is: Can the elements that survived in $\mathcal{S}_{t,1}$ compensate for the removed elements from $\mathcal{S}_{t,2}$? In this paper, we develop tools to prove that if the elements of $\mathcal{S}_{t,1}$ are chosen as in RAM's lines 3-4, this is indeed the case (proof of Theorem 10), providing the first provable approximation guarantees for RSM via RAM.

B. Step-by-step description

RAM executes four steps for each $t = 1, \ldots, T$:

a) Initialization (RAM's line 2): RAM defines two auxiliary sets, namely, $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$, and initializes them with the empty set (RAM's line 2).

b) Construction of set $\mathcal{S}_{t,1}$ (RAM's lines 3-4): RAM constructs $\mathcal{S}_{t,1}$ by selecting β_t elements, among all $s \in \mathcal{V}_t$, with the highest values $f(s)$. In detail, $\mathcal{S}_{t,1}$ is constructed by first indexing the elements in $\mathcal{V}_t$ such that $\mathcal{V}_t \equiv \{v_{t,1}, \ldots, v_{t,|\mathcal{V}_t|}\}$ and $f(v_{t,1}) \geq \ldots \geq f(v_{t,|\mathcal{V}_t|})$ (RAM's line 3), and then by including in $\mathcal{S}_{t,1}$ the fist β_t elements (RAM's line 4).

c) Construction of set $\mathcal{S}_{t,2}$ (RAM's lines 5-8): RAM constructs $\mathcal{S}_{t,2}$ by picking greedily $\alpha_t - \beta_t$ elements from $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$, taking also into account the history of selections and removals, that is, $\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus \mathcal{B}_{t-1}$. Specifically, the "while loop" (RAM's lines 5-8) 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 $f(\mathcal{A}_1 \setminus \mathcal{B}_1, \ldots, \mathcal{A}_{t-1} \setminus \mathcal{B}_{t-1}, \mathcal{S}_{t,2} \cup \{y\})$.

d) Construction of set $\mathcal{A}_t$ (RAM's line 9): RAM constructs $\mathcal{A}_t$ as the union of $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$.

The above steps are valid for any number of failures β_t.

C. Running time

We now analyze the computational complexity of RAM.

Proposition 2. *At each $t = 1, 2, \ldots$, RAM runs in $O[|\mathcal{V}_t|(\alpha_t - \beta_t)\tau_f]$ time, where τ_f is f's evaluation time.*

Remark 3 (Minimal running time)**.** *Even though RAM robustifies the traditional, failure-free sequential optimization in eq.* (1)*, RAM has the same order of running time as the state-of-the-art algorithms for eq.* (1) *[18, Section 4] [23, Section 8].*

In summary, RAM selects adaptively a solution for RSM, in minimal running time, and is valid for any number of failures. We quantify its approximation performance next.

III. Suboptimality Guarantees

We present RAM's suboptimality bounds. We first present RAM's a priori bounds, and, then, the a posteriori bounds. Finally, we present the latter's pre-failure approximations.

A. Curvature and total curvature

To present RAM's suboptimality bounds we use the notions of *curvature* and *total curvature*. To this end, we start by recalling the definitions of *modularity* and *submodularity*, where we consider the notation:

- $\mathcal{V} \triangleq \bigcup_{i=1}^{T} \mathcal{V}_t$; i.e., $\mathcal{V}$ is the union across the horizon T of all the available elements to choose from;

Definition 4 (Modularity [64])**.** *$f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is modular if and only if $f(\mathcal{A}) = \sum_{v \in \mathcal{A}} f(v)$, for any $\mathcal{A} \subseteq \mathcal{V}$.*

Therefore, if f is modular, then $\mathcal{V}$'s elements complement each other through f. Particularly, Definition 4 implies $f(\{v\} \cup \mathcal{A}) - f(\mathcal{A}) = f(v)$, for any $\mathcal{A} \subseteq \mathcal{V}$ and $v \in \mathcal{V} \setminus \mathcal{A}$.

Definition 5 (Submodularity [64])**.** *$f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is* submodular *if and only if $f(\mathcal{A} \cup \{v\}) - f(\mathcal{A}) \geq f(\mathcal{B} \cup \{v\}) - f(\mathcal{B})$, for any $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$ and $v \in \mathcal{V}$.*

The definition implies f is submodular if and only if the return $f(\mathcal{A} \cup \{v\}) - f(\mathcal{A})$ diminishes as $\mathcal{A}$ grows, for any v. In contrast to f being modular, if f is submodular, then $\mathcal{V}$'s elements substitute each other. Specifically, without loss of generality, consider f to be non-negative: then, Definition 5 implies $f(\{v\} \cup \mathcal{A}) - f(\mathcal{A}) \leq f(v)$. That is, in the presence of $\mathcal{A}$, v's contribution to $f(\{v\} \cup \mathcal{A})$'s value is diminished.

[5]An approximation algorithm is called *optimal* when it achieves the tightest possible achievable suboptimality bound among all polynomial time algorithms, given a worst-case family of functions f.

Algorithm 2: Online greedy algorithm [18, Section 4].

```
Input: Integer $T > 0$; $f : 2^{\mathcal{K}_1} \cup \cdots \cup 2^{\mathcal{K}_T} \mapsto \mathbb{R}$ such that
    $f$ is non-decreasing and $f(\emptyset) = 0$; integers $\delta_t$ such that
    $0 \leq \delta_t \leq |\mathcal{K}_t|$, for all $t = 1, \ldots, T$.
Output: At each step $t = 1, \ldots, T$, set $\mathcal{M}_t$.

1: for all $t = 1, \ldots, T$ do
2:     $\mathcal{M}_t \leftarrow \emptyset$;
3:     while $|\mathcal{M}_t| < \delta_t$ do
4:         $x \in \arg\max_{y \in \mathcal{K}_t \setminus \mathcal{M}_t} f(\mathcal{M}_1, \ldots, \mathcal{M}_{t-1}, \mathcal{M}_t \cup \{y\})$;
5:         $\mathcal{M}_t \leftarrow \{x\} \cup \mathcal{M}_t$;
6:     end while
7: end for
```

Definition 6. (Curvature [27]) *Consider a non-decreasing submodular $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that $f(v) \neq 0$, for any $v \in \mathcal{V}$, without loss of generality. Then, f's* curvature *is defined as*

$$\kappa_f \triangleq 1 - \min_{v \in \mathcal{V}} \frac{f(\mathcal{V}) - f(\mathcal{V} \setminus \{v\})}{f(v)}. \tag{3}$$

Definition 6 implies $\kappa_f \in [0, 1]$. Particularly, κ_f measures how far f is from modularity: if $\kappa_f = 0$, then $f(\mathcal{V}) - f(\mathcal{V} \setminus \{v\}) = f(v)$, for all $v \in \mathcal{V}$; that is, f is modular. In contrast, if $\kappa_f = 1$, then there exist $v \in \mathcal{V}$ such that $f(\mathcal{V}) = f(\mathcal{V} \setminus \{v\})$; that is, v has no contribution to $f(\mathcal{V})$ in the presence of $\mathcal{V} \setminus \{v\}$. Therefore, κ_f can also been interpreted as a measure of how much $\mathcal{V}$'s elements complement/substitute each other.

Definition 7 (Total curvature [23], [30]). *Consider a monotone $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$. Then, f's total curvature is defined as*

$$c_f \triangleq 1 - \min_{v \in \mathcal{V}} \min_{\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}} \frac{f(\{v\} \cup \mathcal{A}) - f(\mathcal{A})}{f(\{v\} \cup \mathcal{B}) - f(\mathcal{B})}. \tag{4}$$

Similarly to κ_f, it also is $c_f \in [0, 1]$. Remarkably, when f is submodular, then $c_f = \kappa_f$. Generally, if $c_f = 0$, then f is modular, while if $c_f = 1$, then eq. (4) implies the assumption that f is non-decreasing. In [65], any monotone f with total curvature c_f is called c_f-submodular, as repeated below.[6]

Definition 8 (c_f-submodularity [65]). *Any monotone function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ with total curvature c_f is called c_f*-submodular.

Remark 9 (Dependence on the size of $\mathcal{V}$ and length of horizon T). *Evidently, both κ_f and c_f are non-decreasing as $\mathcal{V}$ grows (cf. Definition 6 and Definition 7). Therefore, κ_f and c_f are also non-decreasing as T increases, since $\mathcal{V} \equiv \bigcup_{i=1}^{T} \mathcal{V}_t$.*

B. A priori suboptimality bounds

We present RAM's a priori suboptimality bounds, using the above notions of curvature. We use also the notation:

- $f^\star$ is the optimal value of `RSM`;
- $\mathcal{A}_{1:t} \triangleq (\mathcal{A}_1, \ldots, \mathcal{A}_t)$, where $\mathcal{A}_t$ is the selected set by RAM at $t = 1, 2, \ldots$;
- $(\mathcal{B}_1^\star, \ldots, \mathcal{B}_T^\star)$ is an optimal removal from $\mathcal{A}_{1:T}$;
- $\mathcal{B}_{1:t}^\star \triangleq (\mathcal{B}_1^\star, \ldots, \mathcal{B}_t^\star)$;
- $\mathcal{A}_{1:t} \setminus \mathcal{B}_{1:t}^\star \triangleq (\mathcal{A}_1 \setminus \mathcal{B}_1^\star, \ldots, \mathcal{A}_t \setminus \mathcal{B}_t^\star)$.

Theorem 10 (A priori bounds). RAM *selects $\mathcal{A}_{1:T}$ such that $|\mathcal{A}_t| \leq \alpha_t$, and if f is submodular, then*

$$\frac{f(\mathcal{A}_{1:T} \setminus \mathcal{B}_{1:T}^\star)}{f^\star} \geq \begin{cases} \frac{1 - e^{-\kappa_f}}{\kappa_f}(1 - \kappa_f), & T = 1; \\ (1 - \kappa_f)^4, & T > 1; \end{cases} \tag{5}$$

whereas, if f is c_f-submodular, then

$$\frac{f(\mathcal{A}_{1:T} \setminus \mathcal{B}_{1:T}^\star)}{f^\star} \geq \begin{cases} (1 - c_f)^3, & T = 1; \\ (1 - c_f)^5, & T > 1. \end{cases} \tag{6}$$

Evidently, Theorem 10's bounds are a priori, since ineqs. (5)'s and (6)'s right-hand-sides are independent of the selected $\mathcal{A}_{1:T}$ by RAM, and the incurred failures $\mathcal{B}_{1:T}^\star$.

Importantly, the bounds compare RAM's selection $\mathcal{A}_{1:T}$ against an optimal one that knows a priori all future failures (and achieves that way the value $f^\star$). Instead, RAM's has no knowledge of the future failures. Within this challenging setting, Theorem 10 nonetheless implies: for functions f with $\kappa_f < 1$ or $c_f < 1$, RAM's selection $\mathcal{A}_{1:T}$ is finitely close to the optimal, instead of arbitrarily suboptimal. Indeed, then Theorem 10's bounds are non-zero. We discuss functions with $\kappa_f < 1$ or $c_f < 1$ below, along with relevant applications.

Remark 11 (Functions with $\kappa_f < 1, c_f < 1$, and applications). *Functions with $\kappa_f < 1$ are the concave over modular functions [28, Section 2.1] and the* $\log\det$ *of positive-definite matrices [69]. Also, functions with $c_f < 1$ are the support selection functions [66], the average minimum square error of the Kalman filter (trace of error covariance) [70, Section IV], and the LQG cost as a function of the active sensors [10, Theorem 4]. The aforementioned functions appear in control and machine learning applications such as feature selection [21], [71], and actuator and sensor scheduling [5]–[13], [70].*

Evidently, when κ_f and c_f tend to 0, then RAM becomes optimal, since all bounds in Theorem 10 tend to 1; for example, $1/\kappa_f(1 - e^{-\kappa_f})(1 - \kappa_f)$ increases as κ_f decreases, and its limit is equal to 1 for $\kappa_f \to 0$. Application examples of this sort involve the regression of Gaussian processes with RBF kernels [69, Theorem 5], such as in sensor selection for temperature monitoring [72].

Finally, since both κ_f and c_f are non-decreasing in T and $\mathcal{V}$ (Remark 9), the bounds are non-increasing in T and $\mathcal{V}$.

Tightness and optimality (towards a posteriori bounds): RAM's curvature-dependent bounds are the first suboptimality bounds for `RSM`, and make a first step towards separating the classes of monotone functions into functions for which `RSM` can be approximated well (low curvature functions), and functions for which it cannot (high curvature functions). Moreover, although for the failure-free eq. (1) the a priori bounds $1/\kappa_f(1 - e^{-\kappa_f})$ and $1/(1 + \kappa_f)$ (where f is submodular) are known to be tight [27, Theorem 2.12, Theorem 5.4], the tightness of ineq. (5) is an open problem. Similarly, although for eq. (1) the a priori bound $1 - c_f$ (where f is c_f-submodular) is known to be optimal (the tightest possible in polynomial time in a worst-case) [23, Theorem 8.6], the

[6]Lehmann et al. [65] defined c_f-submodularity by considering in eq. (4) $\mathcal{A} \subseteq \mathcal{B}$ instead of $\mathcal{A} \subseteq \mathcal{V}$. Generally, non submodular but monotone functions have been referred to as *approximately* or *weakly* submodular [29], [66], names that have also been adopted for the definition of c_f in [65], e.g., in [67], [68].

optimality of ineq. (6) is an open problem. Notably, in the latter case (f is c_f-submodular) both $1-c_f$ and the bound in ineq. (6) are 0 for $c_f=1$, which is in agreement with the inapproximability of both eq. (1) and RSM in the worst-case.

In contrast to Theorem 10's a priori bounds, we next present tight and/or optimal a posteriori bounds.

C. A posteriori suboptimality bounds

We now present RAM's a posteriori bounds, which are computable once all failures up to step t have been observed. Henceforth, we use the notation:

- $f_t^\star$ is the optimal value of RSM for $T=t$;
- $\mathcal{M}_t$ is the set returned by the online, failure-free greedy Algorithm 2 at $t=1,\ldots,T$,[7] when we consider therein $\delta_t=\alpha_t-\beta_t$ and $\mathcal{K}_t=\mathcal{V}_t\setminus\mathcal{S}_{1,t}$;
- $\mathcal{M}_{1:t}\triangleq\{\mathcal{M}_1,\ldots,\mathcal{M}_t\}$.

Remark 12 (Interpretation of $\mathcal{M}_{1:t}$)**.** *Since each $\mathcal{S}_{1,t}$ is the expected future failures ("baits") selected in RAM's lines 3-4 (see Section II), $\mathcal{M}_{1:t}$ are the sets one would greedily select per Algorithm 2 if it was known a priori that indeed the future failures are the $\mathcal{S}_{1,t}$, $t=1,\ldots,T$.*

Theorem 13 (A posteriori bounds)**.** *For all $t=1,\ldots,T$, given the observed history $\mathcal{B}_{1:t}^\star$, RAM selects $\mathcal{A}_t$ such that $|\mathcal{A}_t|\leq\alpha_t$, and if f is submodular, then*

$$\frac{f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)}{f_t^\star}\geq\begin{cases}\frac{1-e^{-\kappa_f}}{\kappa_f}\frac{f(\mathcal{A}_1\setminus\mathcal{B}_1^\star)}{f(\mathcal{M}_1)}, & t=1;\\ \frac{1}{1+\kappa_f}\frac{f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)}{f(\mathcal{M}_{1:t})}, & t>1;\end{cases}\tag{7}$$

whereas, if f is c_f-submodular, then

$$\frac{f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)}{f_t^\star}\geq(1-c_f)\frac{f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)}{f(\mathcal{M}_{1:t})}.\tag{8}$$

Theorem 14 (Tightness and optimality)**.** *There exist families of f such that the suboptimality bounds in ineq.* (7) *are tight. Also, there exist families of f such that the suboptimality bounds in eq.* (8) *are optimal (the tightest possible) across all algorithms that evaluate f a polynomial number of times.*[8]

The bounds break down into the a priori κ_f- and c_f-depended parts, and the a posteriori $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)/f(\mathcal{M}_{1:t})$. We refer to the latter as a posteriori since it is computable *after* $\mathcal{B}_t^\star$ has been observed. Intuitively, the a posteriori part captures how successful the "bait" $\mathcal{S}_{1,t}$ has been in approximating the anticipated worst-case failure $\mathcal{B}_t^\star$. Indeed, if $\mathcal{B}_t^\star=\mathcal{S}_{1,t}$ for all $t=1,2,\ldots$, then $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)/f(\mathcal{M}_{1:t})=1$ and Theorem 13's bounds become the tight/optimal a priori bounds $1/\kappa_f(1-e^{-\kappa_f})$, $1/(1+\kappa_f)$ and $1-c_f$,[9] and, as such, they are also tighter than Theorem 10's a priori bounds.

In general, Theorem 13's a posteriori bounds may be looser than Theorem 10's a priori bounds; yet, they are tighter when $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)/f(\mathcal{M}_{1:t})$ is close enough to 1: e.g., for f being c_f-submodular and $T>1$, if $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)/f(\mathcal{M}_{1:t})>(1-c_f)^4$, then the a posteriori bound in eq. (8) is tighter than the a priori in eq. (6). Indeed, in the numerical evaluations of Section IV, $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)/f(\mathcal{M}_{1:t})$ is nearly 1, whereas $(1-c_f)^4\leq .0001$; thus, eq. (8) is 3 orders tighter than eq. (6).

Notably, the a priori parts $1/\kappa_f(1-e^{-\kappa_f})$, $1/(1+\kappa_f)$ are non-zero for any values of κ_f. In more detail, $1/\kappa_f(1-e^{-\kappa_f})\geq 1-1/e$ and $1/(1+\kappa_f)\geq 1/2$ for all $\kappa_f\in[0,1]$; particularly, $1/\kappa_f(1-e^{-\kappa_f})$ increases as κ_f decreases, and its limit is equal to 1 for $\kappa_f\to 0$. Therefore, in contrast to the a priori bound in eq. (5), which for $\kappa_f=1$ becomes 0, eq. (7) for $\kappa_f=1$ becomes instead

$$\frac{f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)}{f_t^\star}\geq\begin{cases}(1-1/e)\frac{f(\mathcal{A}_1\setminus\mathcal{B}_1^\star)}{f(\mathcal{M}_1)}, & t=1;\\ \frac{f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)}{2f(\mathcal{M}_{1:t})}, & t>1.\end{cases}\tag{9}$$

Nevertheless, such simplification for eq. (8) is not evident, a fact that is in agreement with both (i) RSM's inapproximability when f is *not* submodular, necessitating per-instance suboptimality bounds for any polynomial time algorithm, and (ii) eq. (8)'s optimality per Theorem 14.

Overall, Theorem 13's bounds are computable online, at each $t=1,2,\ldots$, *after* failure $\mathcal{B}_t^\star$ has been observed. We next approximate the bounds *before* $\mathcal{B}_t^\star$ occurs.

D. Pre-failure approximations of post-failure bounds

We present pre-failure approximations to Theorem 13's post-failure bounds. In particular, we propose a method to lower bound $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)$ by a value $\hat{f}_t$, at each $t=1,\ldots,T$, given $\mathcal{B}_{1:t-1}^\star$ (but before $\mathcal{B}_t^\star$ occurs).

In more detail, we recall $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)$ is the value of the constrained optimization problem

$$f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)\equiv\min_{\mathcal{B}_t\subseteq\mathcal{A}_t,\;|\mathcal{B}_t|\leq\beta_t} f(\mathcal{A}_{1:t-1}\setminus\mathcal{B}_{1:t-1}^\star,\mathcal{A}_t\setminus\mathcal{B}_t).\tag{10}$$

Computing $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)$ is NP-hard, even if f is submodular [73]. But lower bounding $f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)$ can be efficient. Specifically, the non-constrained reformulation of eq. (10) in eq. (11) below is efficiently solvable (see [73]–[76] for f being submodular; and [77] for f being c_f-submodular):

$$\min_{\mathcal{B}_t\subseteq\mathcal{A}_t} f(\mathcal{A}_{1:t-1}\setminus\mathcal{B}_{1:t-1}^\star,\mathcal{A}_t\setminus\mathcal{B}_t)+\lambda_t|\mathcal{B}_t|,\tag{11}$$

where $\lambda_t\geq 0$ and constant (λ_t acts similarly to a Lagrange multiplier [78]). Evidently, Lemma 15 below holds true, where $\hat{\mathcal{B}}_t(\lambda_t)$ denotes an optimal solution to eq. (11), i.e.,

$$\hat{\mathcal{B}}_t(\lambda_t)\in\arg\min_{\mathcal{B}_t\subseteq\mathcal{A}_t} f(\mathcal{A}_{1:t-1}\setminus\mathcal{B}_{1:t-1}^\star,\mathcal{A}_t\setminus\mathcal{B}_t)+\lambda_t|\mathcal{B}_t|,\tag{12}$$

and where $\hat{f}_t(\lambda_t)$ denotes the value of $f(\mathcal{A}_{1:t-1}\setminus\mathcal{B}_{1:t-1}^\star,\mathcal{A}_t\setminus\mathcal{B}_t)$ when $\mathcal{B}_t=\hat{\mathcal{B}}_t(\lambda_t)$, i.e.,

$$\hat{f}_t(\lambda_t)\triangleq f(\mathcal{A}_{1:t-1}\setminus\mathcal{B}_{1:t-1}^\star,\mathcal{A}_t\setminus\hat{\mathcal{B}}_t(\lambda_t)).\tag{13}$$

Lemma 15. *There exists $\lambda_t^\star$ such that $\hat{f}_t(\lambda_t)\leq f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)$ and $|\hat{\mathcal{B}}_t(\lambda_t)|>\beta_t$ for $\lambda_t<\lambda_t^\star$; whereas, $\hat{f}_t(\lambda_t)\geq f(\mathcal{A}_{1:t}\setminus\mathcal{B}_{1:t}^\star)$ and $|\hat{\mathcal{B}}_t(\lambda_t)|\leq\beta_t$ for $\lambda_t\geq\lambda_t^\star$.*

[7]We refer to Algorithm 2 as "online" since each $\mathcal{M}_t$ can be chosen in real time (at time t) sequentially, i.e., given the history of past selections $\mathcal{M}_1,\ldots,\mathcal{M}_{t-1}$. Observe, however, that if one wishes so one can also execute all steps of Algorithm 2 offline at time $t=0$.

[8]Theorem 14's function families are the same as those in the proofs of [27, Theorem 2.12, Theorem 5.4] and [23, Theorem 8.6], which prove the tightness and optimality of $1/\kappa_f(1-e^{-\kappa_f})$, $1/(1+\kappa_f)$ and $(1-c_f)$ for eq. (1).

[9]Theorem 14 is proved based on this observation.

Algorithm 3: Bisection.

Input: Integer β_t per RSM; function f per RSM; histories $\mathcal{A}_{1:t}$ and $\mathcal{B}^\star_{1:t-1}$; $u_0 > 0$ such that $|\hat{\mathcal{B}}_t(u_0)| < \beta_t$, where $\hat{\mathcal{B}}_t(\cdot)$ is defined in eq. (12); $\epsilon > 0$, which defines bisection's stopping condition (accuracy level).

Output: $\lambda_t \geq 0$ such that $\hat{f}_t(\lambda_t) \leq f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})$, where λ_t and $\hat{f}_t(\lambda_t)$ are defined in eq. (11) and eq. (13).

```
1: $l \leftarrow 0$;   $u \leftarrow u_0$;   $\lambda_t \leftarrow (l+u)/2$;
2: while $u - l > \epsilon$ do
3:     Find $\hat{\mathcal{B}}_t(\lambda_t)$ by solving eq. (12);
4:     if $|\hat{\mathcal{B}}_t(\lambda_t)| < \beta_t$ then
5:         $u \leftarrow \lambda_t$; {$u$ always satisfies $|\hat{\mathcal{B}}_t(u)| < \beta_t$}
6:     else
7:         $l \leftarrow \lambda_t$; {$l$ always satisfies $|\hat{\mathcal{B}}_t(l)| \geq \beta_t$}
8:     end if
9:     $\lambda_t \leftarrow (l+u)/2$;
10: end while
11: $\lambda_t \leftarrow l$; {$l$ is $\epsilon$-close to $\lambda_t^\star$ ($\lambda_t^\star$ is defined in Lemma 15) and satisfies $|\hat{\mathcal{B}}_t(l)| \geq \beta_t$}
12: return $\lambda_t$.
```

To observe such a value $\lambda_t^\star$ exists, it suffices to observe: (i) for $\lambda_t = 0$, the cardinality of $\hat{\mathcal{B}}_t$ in eq. (11) is unconstrained, and, thus, the optimal solution in eq. (11) is to remove all $\mathcal{A}_t$, which implies $|\hat{\mathcal{B}}_t(\lambda_t)| = \alpha_t \geq \beta_t$; (ii) more generally, for $\lambda_t > 0$, the cardinality of $\hat{\mathcal{B}}_t$ in eq. (11) is increasingly penalized as λ_t increases, and, thus, $|\hat{\mathcal{B}}_t(\lambda_t)|$ is a decreasing function of λ_t (in particular, if $\lambda_t \to +\infty$, then $|\hat{\mathcal{B}}_t(\lambda_t)| \to 0$). Now, given (i)-(ii), denote by $\lambda_t^\star$ the first value of λ_t such that $|\hat{\mathcal{B}}_t(\lambda_t)| \leq \beta_t$, when λ_t is initially set to 0 and then continuously increases: then, for $\lambda_t < \lambda_t^\star$, it is $|\hat{\mathcal{B}}_t(\lambda_t)| > \beta_t$, and, thus, $\hat{f}_t(\lambda_t) \leq f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})$, since $|\hat{\mathcal{B}}_t(\lambda_t)| > \beta_t = |\mathcal{B}_t^\star|$ and $\hat{\mathcal{B}}_t(\lambda_t)$ is an optimal solution to eq. (11); whereas, for $\lambda_t \geq \lambda_t^\star$, it is $|\hat{\mathcal{B}}_t(\lambda_t)| \leq \beta_t$, and, thus, $\hat{f}_t(\lambda_t) \geq f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})$.

Although $\lambda_t^\star$ is unknown, it can be approximated by using bisection. For example, Algorithm 3 uses bisection with accuracy level $\epsilon > 0$ (Algorithm 3's lines 2-10) to find a λ_t that is ϵ-close to $\lambda_t^\star$ and for which $\hat{f}_t(\lambda_t) \leq f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})$. To start the bisection, Algorithm 3 assumes a large enough $u_0 \geq 0$ such that $|\hat{\mathcal{B}}_t(u_0)| < \beta_t$; such a u_0 can be found since $|\hat{\mathcal{B}}_t(u_0)| \to 0$ for $u_0 \to +\infty$. Next, at each "while loop" (lines 2-10 of Algorithm 3), $\lambda_t^\star \in [l, u]$, since $|\hat{\mathcal{B}}_t(u)| < \beta_t$ and $|\hat{\mathcal{B}}_t(l)| \geq \beta_t$ (cf. line 5 and line 7 of Algorithm 3). Per line 2 of the algorithm, l and u are updated until $u - l \leq \epsilon$. Then, after at most $\log_2[(u-l)/\epsilon]$ iterations, the algorithm terminates by setting λ_t equal to the latest value of l (lines 11-12 of the algorithm). Therefore, λ_t is ϵ-close to $\lambda_t^\star$ and satisfies $|\hat{\mathcal{B}}_t(l)| \geq \beta_t$, which in turn implies $\hat{f}_t(\lambda_t) \leq f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})$, as desired. All in all, given an approximation λ_t to $\lambda_t^\star$, Lemma 15 implies the following approximation of Theorem 13's bounds.

Corollary 16 (Pre-failure approximation of a posteriori bounds)**.** *Let Algorithm 3 return λ_t, for $t = 1, \ldots, T$.* RAM *selects $\mathcal{A}_t$ such that $|\mathcal{A}_t| \leq \alpha_t$, and if f is submodular, then*

$$\frac{f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})}{f_t^\star} \geq \begin{cases} \frac{1-e^{-\kappa_f}}{\kappa_f} \frac{\hat{f}_1(\lambda_1)}{f(\mathcal{M}_1)}, & t = 1; \\ \frac{1}{1+\kappa_f} \frac{\hat{f}_t(\lambda_t)}{f(\mathcal{M}_{1:t})}, & t > 1; \end{cases} \quad (14)$$

whereas if f is c_f-submodular, then

$$\frac{f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})}{f_t^\star} \geq (1 - c_f) \frac{\hat{f}_t(\lambda_t)}{f(\mathcal{M}_{1:t})}. \quad (15)$$

Corollary 16 describes an online mechanism to predict RAM's performance before the upcoming failures, step by step.

Remark 17 (Utility of Corollary 16's bounds)**.** *For $T = 1$, Corollary 16's bounds are computed before $\mathcal{B}_1^\star$ occurs, which implies $\hat{f}_1$ can be computed* offline (*at any time $t < 1$*)*. Therefore, Corollary 16's bounds, when tighter than Theorem 10's a priori bounds, allow for an a priori assessment of* RAM*'s approximation performance* (before RAM *is deployed in the real world*)*. Examples of 1-step design problems where $T = 1$, include* robust actuator and sensor placement *[7], [16], [79],* robust feature selection *[21], [32], [80],* robust graph covering *[81], and* robust server placement *[15], [82].*

For $T > 1$, Corollary 16's bounds can only be computed online*, once $\mathcal{B}^\star_{1:t-1}$ has been observed (but before $\mathcal{B}^\star_t$ has occurred), for each $t = 1, \ldots, T$.*[10] *As such, the bounds can be used to balance the trade-off between (i) computation time requirements (including computation and energy consumption requirements) for solving each step t of RSM, and (ii) approximation performance requirements for solving RSM at each t. For example, if Corollary 16's bounds indicate a good performance by* RAM *at t (e.g., the bounds are above a given threshold),* RAM *is used to select $\mathcal{A}_t$, since* RAM *is computation time inexpensive, being a polynomial time algorithm. However, if the bounds indicate poor performance by* RAM *at t (less than the given threshold), then an optimal algorithm can be used instead at t (such as the one proposed in [34]), but at the expense of higher computation time, since any optimal algorithm is non-polynomial in the worst-case.*[11]

IV. Applications

We evaluate RAM's performance in applications. We start by assessing its near-optimality against worst-case failures. We continue by testing its sensitivity against *non* worst-case failures, particularly, random and greedily selected failures. For such failures, one would expect RAM's performance to be the same, or improve, since RAM is designed to withstand the worst-case. To these ends, we consider two applications from the introduction: *sensor scheduling for autonomous navigation*, and *target tracking with wireless sensor networks*.

[10]Corollary 16's bounds can only be computed *online* since RAM itself is an online algorithm, computing $\mathcal{A}_t$ only once $\mathcal{B}^\star_{1:t-1}$ has been observed (yet before $\mathcal{B}^\star_t$ has occurred), for each $t = 1, \ldots, T$.

[11]Even when RAM is used in combination with another algorithm to choose $\mathcal{A}_{1:t}$, Corollary 16's bounds are still applicable since they are *algorithm agnostic* (cf. proof of Theorem 13).

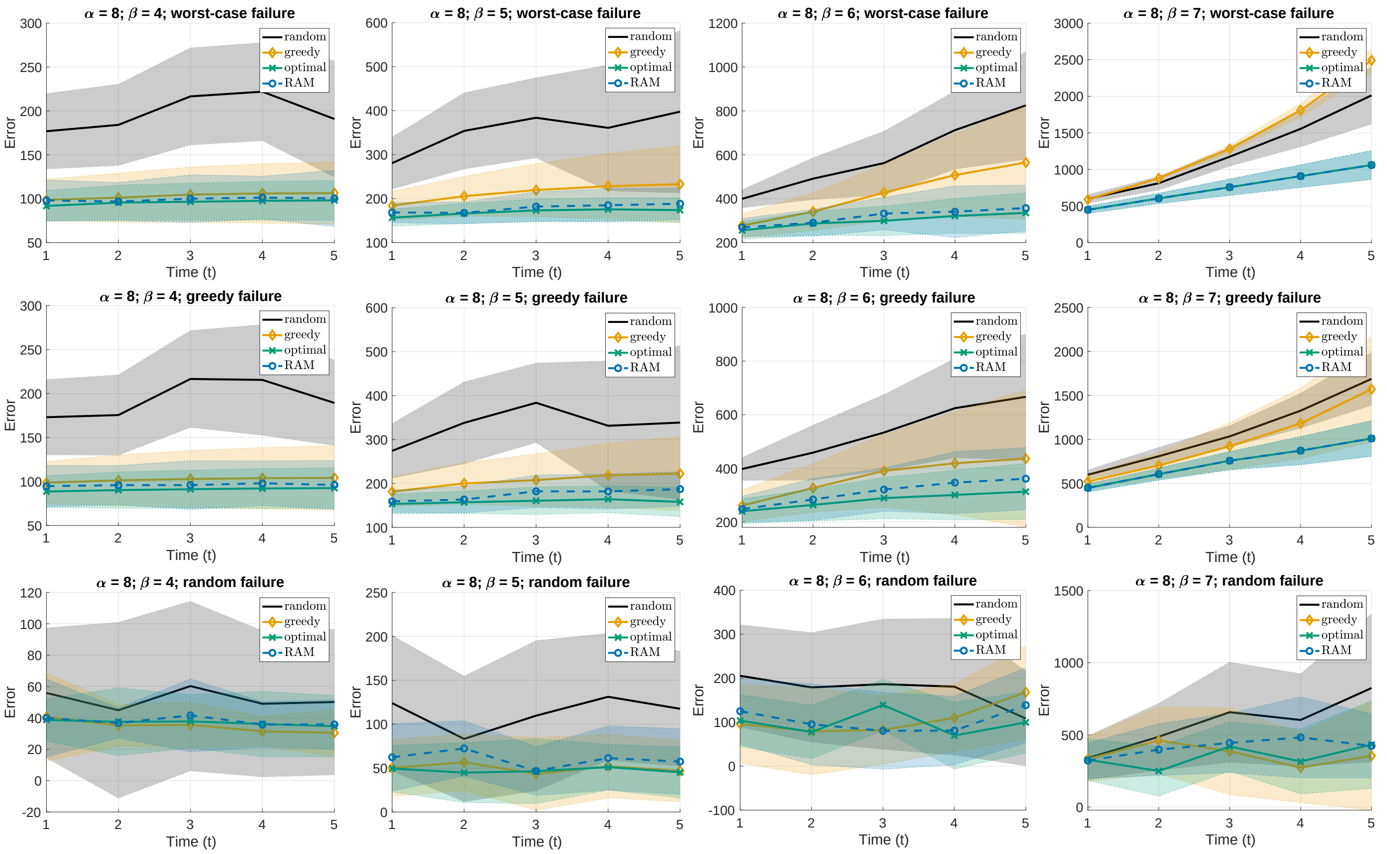
Fig. 1. Representative simulation results for the application *sensor scheduling for autonomous navigation*. Results are averaged across 100 Monte Carlo runs. Depicted is the estimation error for increasing time t, per eq. (16), where $\alpha_t = \alpha = 8$ across all subfigures, whereas $\beta_t = \beta$ where β varies across subfigures column-wise. Finally, the failure type also varies, but row-wise. *Each subfigure has different scale.*

A. Sensor scheduling for autonomous navigation

We demonstrate RAM's performance in autonomous navigation scenarios, in the presence of sensing failures. We focus on small-scale instances, to enable RAM's comparison with a brute-force algorithm attaining the optimal to RSM. Instead, in Section IV-B we consider larger-scale instances.

A UAV moves in a 3D space, starting from a randomly selected initial location. Its objective is to land at $[0,\ 0,\ 0]$ with zero velocity. The UAV is modeled as a double-integrator with state $x_t = [p_t\ v_t]^\top \in \mathbb{R}^6$, where $t = 1, 2, \ldots$ is the time index, p_t is the UAV's position, and v_t is its velocity. The UAV controls its acceleration. The process noise has covariance $\mathbf{I}_6$.

The UAV is equipped with two on-board sensors: a GPS, measuring the UAV's position p_t with a covariance $2 \cdot \mathbf{I}_3$, and an altimeter, measuring p_t's altitude component with standard deviation $0.5\mathrm{m}$. Also, the UAV can communicate with 10 linear ground sensors. These sensors are randomly generated at each Monte Carlo run, along with their noise covariance.

The UAV has limited on-board battery power and measurement-processing bandwidth. Hence, it uses only a few sensors at each t. Particularly, among the 12 available sensors, the UAV uses at most α, where α varies from 1 to 12 in the Monte Carlo analysis (per RSM's notation, $\alpha_t = \alpha$ for all $t = 1, 2, \ldots$). The UAV selects the sensors to minimize the cumulative batch-state error over a horizon $T = 5$, captured by

$$c(\mathcal{A}_{1:t}) = \log\det[\Sigma_{1:t}(\mathcal{A}_{1:t})], \qquad (16)$$

where $\Sigma_{1:t}(\mathcal{A}_{1:t})$ is the error covariance of the minimum variance estimator of $(x_1, \ldots, x_t)$ given the used sensors up to t [83]. Notably, $f(\mathcal{A}_{1:t}) = -c(\mathcal{A}_{1:t})$ is non-decreasing and submodular, in congruence to RSM's framework [8].

Finally, we consider that at most β failures are possible at each t (per RSM's notation, $\beta_t = \beta$ for all t). In the Monte Carlo analysis, β varies from 0 to $\alpha - 1$.

Baseline algorithms. We compare RAM with three algorithms. The first algorithm is a brute-force, optimal algorithm, denoted as optimal. Evidently, optimal is viable only for small-scale problem instances, such as herein where the available sensors are 12. The second algorithm performs random selection and is denoted as random. The third algorithm, denoted as greedy, greedily selects sensors to optimize eq. (16) per the failure-free optimization setup in eq. (1).

Results. The results are averaged over 100 Monte Carlo runs. For $\alpha = 8$ and $\beta = 4, 5, 6, 7$, they are reported in Fig. 1. For the remaining α and β values, the qualitative results are the same. From Fig. 1, the following observations are due:

a) Near-optimality against worst-case failures: We focus on Fig. 1's first row of subfigures, where β varies from 4 to 7 (from left to right). Across all β, RAM nearly matches optimal. In contrast, greedy nearly matches optimal only for $\beta = 4$ (and, generally, for $\beta \le \alpha/2$, taking into account the simulation results for the remaining values of α). Expectedly, random is always the worst among all compared algorithms. Importantly, as β tends to α, greedy's performance tends to random's. The observation exemplifies the insufficiency of the traditional optimization paradigm in eq. (1) against failures.

Across all values of α and β in the Monte Carlo analysis, the suboptimality bound in Theorem 13's eq. (7) is at least .59, informing RAM performs at least 50% the optimal (κ_f remains always less than .93, while $f(\mathcal{A}_{1:t} \setminus \mathcal{B}^{\star}_{1:t})/f(\mathcal{M}_{1:t})$ is close to .95). In contrast, in Fig. 1 we observe an almost optimal performance. This is an example where the actual performance of the algorithm is significantly closer to the optimal than what is indicated by the algorithm's suboptimality bound. Indeed, this is a common observation for greedy-like algorithms: for the failure-agnostic greedy in [18] see, e.g., [14].

b) Robustness against non worst-case failures: We compare Fig. 1's subfigures column-wise, where the failure type varies among worst-case, greedy, and random (from top to bottom).[12] Particularly, RAM's performance remains the same, or improves, against non worst-case failures, and the best performance is being observed against random failures, as expected. For example, if we focus on the rightmost column (where $\alpha = 8; \beta = 7$), at $t = 5$, then we observe: for worst-case failures, RAM achieves error 1061; instead, for greedy failures, RAM achieves the reduced error 1010; while for random failures, RAM achieves even less error (less than 500). Finally, against greedy failures, RAM is still superior to greedy, while against random failures, they fare similarly.

Overall, the above numerical simulations demonstrate both the necessity for failure-robust optimization (RSM), as well as the near-optimality of RAM, even for increasing number of failures (system-wide failures). Similar conclusions we make over the second application scenario below.

B. Target tracking with wireless sensor networks

We demonstrate RAM's performance in adversarial target tracking scenarios. Particularly, we consider a mobile target who aims to escape detection from a wireless sensor network (WSN). To this end, the agent causes failures to the network.

A UAV (the target) is moving in a 3D, cubic shaped space. The UAV moves on a straight line, across two opposite boundaries of the cube, keeping constant altitude and speed. The line's start and end points are randomly generated at each Monte Carlo run. The UAV's model is as in the autonomous navigation scenario in Section IV-A.

The WSN is composed of 100 ground sensors. It is aware of the UAV's model, but can only noisily observe its state. The sensors are randomly generated at each Monte Carlo run.

Due to power consumption and bandwidth limitations, only a few sensors can be active at each $t = 1, 2, \ldots$. Particularly, we assume $\alpha = 10$ active sensors at each t. Also, we assume the sensors are activated so the cumulative Kalman filtering error over a horizon $T = 5$ is minimized, as prescribed by

$$c(\mathcal{A}_{1:t}) = \sum_{t=1}^{T} \text{trace}[\Sigma_{t|t}(\mathcal{A}_{1:t})], \qquad (17)$$

where $\Sigma_{t|t}(\mathcal{A}_{1:t})$ is the Kalman filtering error covariance. Noticeably, $f(\mathcal{A}_{1:t}) = -c(\mathcal{A}_{1:t})$ is non-decreasing and c_f-submodular, in agreement with RSM's framework [70].

Finally, at most β failures are possible at each t. In the Monte Carlo analysis, β varies from 1 to $\alpha - 1 = 9$.

Baseline algorithms. We compare RAM with random, and greedy. We cannot compare with optimal, since the network's large-scale size makes optimal unfeasible.

Results. The simulation results are averaged over 100 Monte Carlo runs. For $\beta = 3, 5, 7, 9$, they are reported in Fig. 2, where random is excluded since it results to exceedingly larger errors. For the remaining β values, the qualitative results remain the same. From Fig. 2, we make the observations:

a) Superiority against worst-case failures: We focus on Fig. 2's first row, where β takes the values 3, 5, 7, and 9 (from left to right). For $\beta = 3$ (also, for $\beta = 1, 2$, accounting for the remaining, non depicted simulations), RAM fares similar to greedy. In contrast, for the remaining values of β, RAM dominates greedy, achieving significantly lower error (observe the different scales among the subfigures for $\beta = 5, 7, 9$).

Across all β values in the Monte Carlo analysis (including those in Fig. 2), the suboptimality bound in eq. (8) ranges from .02 to .10, informing that RAM performs at least 2% to 10% the optimal. Specifically, c_f ranges from .89 to .98, whereas $f(\mathcal{A}_{1:t} \setminus \mathcal{B}^{\star}_{1:t})/f(\mathcal{M}_{1:t})$ remains again close to .95. Hence, the possible conservativeness of the bound stems from the conservativeness of its term $1 - c_f$.

b) Robustness against non worst-case failures: We compare Fig. 2's subfigures column-wise. Similarly to the autonomous navigation scenarios, RAM's performance remains the same, or improves, against non worst-case failures, and the lowest error is being observed against random failures. For example, if we focus on the rightmost column (where $\alpha = 10; \beta = 9$), at $t = 5$, then: for worst-case failures, RAM achieves error 611; in contrast, for greedy failures, RAM achieves the lower error 526; and for random failures, RAM achieves the even lower error 456. Generally, against greedy failures, RAM is again still superior to greedy; while against random failures, both have similar performance.

In summary, RAM remains superior even against system-wide failures, and even if the failures are non worst-case.

V. Conclusion

We made the first step to adaptively protect critical control, estimation, and machine learning applications against sequential failures. Particularly, we focused on scenarios requiring the robust discrete optimization of systems per RSM. We provided RAM, the first online algorithm, which adapts to the history of failures, and guarantees a near-optimal performance even against system-wide failures despite its minimal running time. To quantify RAM's performance, we provided per-instance a priori bounds and tight, optimal a posteriori bounds. To this end, we used curvature notions, and contributed a first step towards characterizing the curvature's effect on the per-instance approximability of RSM. Our curvature-dependent bounds complement the current knowledge on the curvature's effect on the approximability of the failure-free optimization

[12]We refer to a failure $\mathcal{B}_t$ as "greedy," when $\mathcal{B}_t$ is selected greedily towards minimizing $f(\mathcal{A}_{1:t-1} \setminus \mathcal{B}_{1:t-1}, \mathcal{A}_t \setminus \mathcal{B}_t)$, where $\mathcal{A}_{1:t}$ and $\mathcal{B}_{1:t-1}$ are given, as in Algorithm 2 but now for minimization instead of maximization.

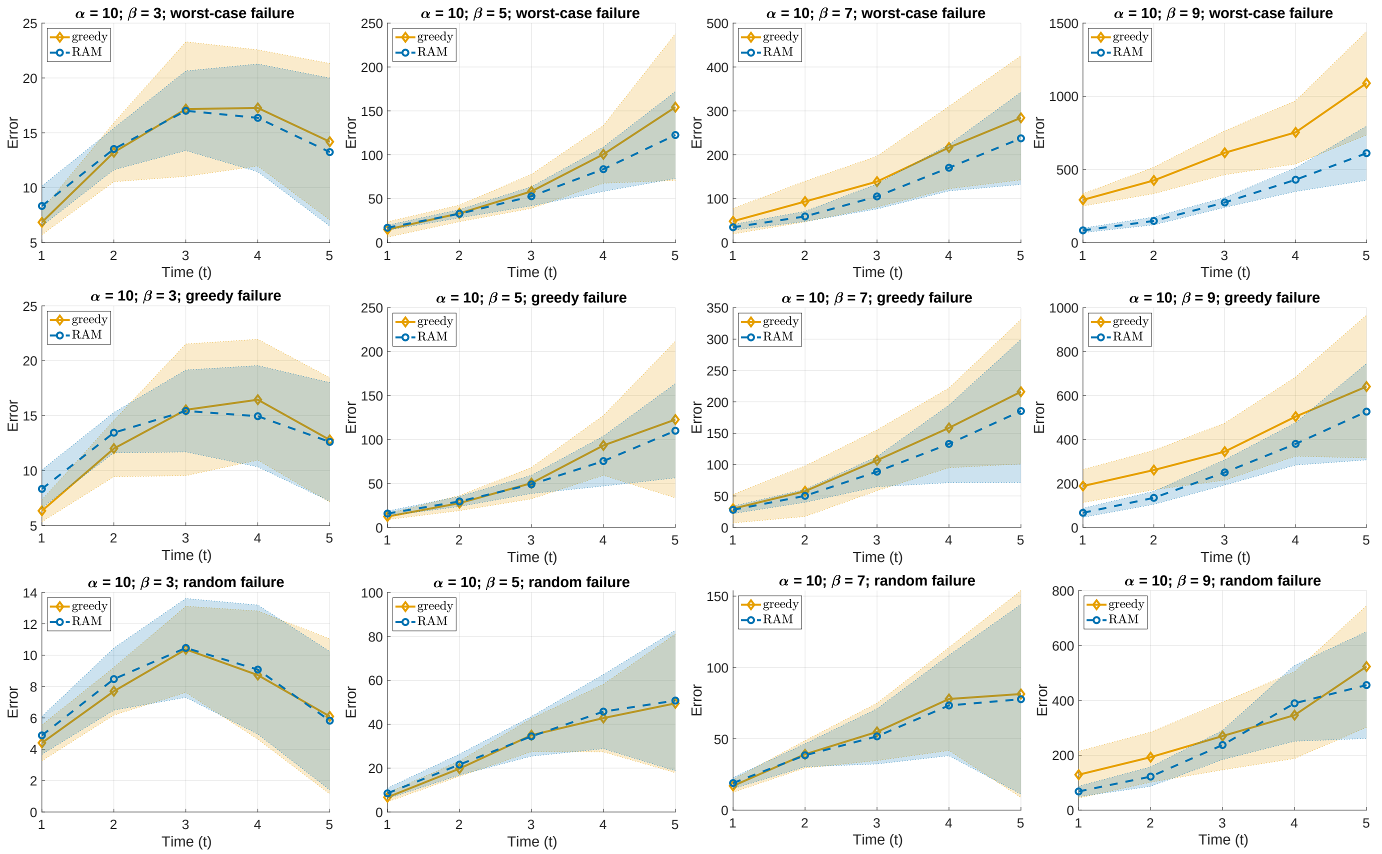

Fig. 2. Representative simulation results for the application *target tracking with wireless sensor networks*. Results are averaged over 100 Monte Carlo runs. Depicted is the estimation error for increasing t, per eq. (17), where $\alpha_t = \alpha = 10$ across all subfigures, whereas $\beta_t = \beta$ where β varies across subfigures column-wise. Finally, the failure type also varies, but row-wise. *Each subfigure has different scale.*

paradigm in eq. (1) [23], [27], [30], [65]. Finally, we supported our theoretical results with numerical evaluations.

The paper opens several avenues for future research. One is the decentralized implementation of RAM towards robust multi-agent autonomy and large-scale network design. And another is the extension of our results to optimization frameworks with general constraints (instead of cardinality, as in RSM), such as observability/controllability requirements, including matroid constraints, towards multi-robot planning.

Appendix

In this appendix, we provide all proofs. We use the notation:

$$f(\mathcal{X} \mid \mathcal{X}') \triangleq f(\mathcal{X} \cup \mathcal{X}') - f(\mathcal{X}'), \tag{18}$$

for any $\mathcal{X}, \mathcal{X}'$. Also, $\mathcal{X}_{1:t} \triangleq (\mathcal{X}_1, \ldots, \mathcal{X}_t)$ for any $\mathcal{X}_1, \ldots, \mathcal{X}_t$ (and $(\mathcal{X}_1, \ldots, \mathcal{X}_t) \equiv \mathcal{X}_1 \cup \ldots \cup \mathcal{X}_t$).

Appendix A: Preliminary Lemmas

We list lemmas that support the proofs.

Lemma 18. *Consider a non-decreasing $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that $f(\emptyset) = 0$. Then, for any $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$,*

$$f(\mathcal{A} \cup \mathcal{B}) \geq (1 - c_f) \left[f(\mathcal{A}) + f(\mathcal{B}) \right].$$

Proof of Lemma 18: Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i \mid \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}). \tag{19}$$

The definition of c_f implies

$$\begin{aligned} f(b_i \mid \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}) &\geq \\ (1 - c_f)\; f(b_i \mid \{b_1, b_2, \ldots, b_{i-1}\}). \end{aligned} \tag{20}$$

The proof is completed by substituting ineq. (20) in eq. (19), along with $f(\mathcal{A}) \geq (1 - c_f)\, f(\mathcal{A})$, since $c_f \leq 1$. ■

Lemma 19. *Consider a non-decreasing $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that $f(\emptyset) = 0$. Then, for any $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A} \cap \mathcal{B} = \emptyset$,*

$$f(\mathcal{A} \cup \mathcal{B}) \geq (1 - c_f) \left[f(\mathcal{A}) + \sum_{b \in \mathcal{B}} f(b) \right].$$

Proof of Lemma 19: Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$. Then,

$$f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i \mid \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}). \tag{21}$$

Now, c_f's Definition 7 implies

$$\begin{aligned} f(b_i \mid \mathcal{A} \cup \{b_1, b_2, \ldots, b_{i-1}\}) &\geq (1 - c_f) f(b_i \mid \emptyset) \\ &= (1 - c_f) f(b_i), \end{aligned} \tag{22}$$

where the latter holds since $f(\emptyset)=0$. The proof is completed by substituting eq. (22) in eq. (21), along with $f(\mathcal{A}) \geq (1-c_f)f(\mathcal{A})$, since $c_f \leq 1$. ∎

Lemma 20. *Consider a non-decreasing $f: 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that $f(\emptyset)=0$. Then, for any $\mathcal{A},\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A}\setminus\mathcal{B} \neq \emptyset$,*

$$f(\mathcal{A}) + (1-c_f)f(\mathcal{B}) \geq (1-c_f)f(\mathcal{A}\cup\mathcal{B}) + f(\mathcal{A}\cap\mathcal{B}).$$

Proof of Lemma 20: Let $\mathcal{A}\setminus\mathcal{B} = \{i_1, i_2, \ldots, i_r\}$, where $r = |\mathcal{A}-\mathcal{B}|$. c_f's Definition 7 implies $f(i_j \mid (\mathcal{A}\cap\mathcal{B}) \cup \{i_1, i_2, \ldots, i_{j-1}\}) \geq (1-c_f)f(i_j \mid \mathcal{B}\cup\{i_1, i_2, \ldots, i_{j-1}\})$, for any $i = 1, \ldots, r$. Summing the 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 21. *Consider a non-decreasing $f: 2^{\mathcal{V}} \mapsto \mathbb{R}$ such that $f(\emptyset)=0$. Then, for any $\mathcal{A},\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{A}\cap\mathcal{B} = \emptyset$,*

$$f(\mathcal{A}) + \sum_{b\in\mathcal{B}} f(b) \geq (1-c_f)f(\mathcal{A}\cup\mathcal{B}).$$

Proof of Corollary 21: Let $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$.

If $\mathcal{A} \neq \emptyset$, then

$$f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i) \geq (1-c_f)f(\mathcal{A}) + \sum_{i=1}^{|\mathcal{B}|} f(b_i) \tag{23}$$

$$\geq (1-c_f)f(\mathcal{A}\cup\{b_1\}) + \sum_{i=2}^{|\mathcal{B}|} f(b_i)$$

$$\geq (1-c_f)f(\mathcal{A}\cup\{b_1,b_2\}) + \sum_{i=3}^{|\mathcal{B}|} f(b_i)$$

$$\vdots$$

$$\geq (1-c_f)f(\mathcal{A}\cup\mathcal{B}),$$

where ineq. (23) holds since $0 \leq c_f \leq 1$, and the remaining inequalities are implied by applying Lemma 20 multiple times ($\mathcal{A}\cap\mathcal{B}=\emptyset$ implies $\mathcal{A}\setminus\{b_1\} \neq \emptyset$, $\mathcal{A}\cup\{b_1\}\setminus\{b_2\} \neq \emptyset$, …, $\mathcal{A}\cup\{b_1, b_2, \ldots, b_{|\mathcal{B}|-1}\}\setminus\{b_{|\mathcal{B}|}\} \neq \emptyset$).

If $\mathcal{A} = \emptyset$, then the proof follows the same reasoning as above but now we need to start from the following inequality, instead of ineq. (23):

$$\sum_{i=1}^{|\mathcal{B}|} f(b_i) \geq (1-c_f)f(\{b_1\}) + \sum_{i=2}^{|\mathcal{B}|} f(b_i).$$

∎

Lemma 22. *Consider the sets $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ selected by* RAM*'s lines 3-4. Also, for all $t = 1, \ldots, T$, let $\mathcal{O}_t$ be any subset of $\mathcal{V}_t \setminus \mathcal{S}_{t,1}$ such that $|\mathcal{O}_t| \leq \alpha_t - \beta_t$. Then,*

$$f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) \geq (1-c_f)^2 f(\mathcal{O}_{1:T}). \tag{24}$$

Proof of Lemma 22: For all $t = 1, \ldots, T$, let $\mathcal{R}_t \triangleq \mathcal{A}_t \setminus \mathcal{B}_t$; namely, $\mathcal{R}_t$ is the set that remains after the optimal (worst-case) removal $\mathcal{B}_t$ from $\mathcal{A}_t$. Furthermore, let $s_{t,2}^i \in \mathcal{S}_{t,2}$ denote the i-th element added to $\mathcal{S}_{t,2}$ per RAM's lines 5-8; i.e., $\mathcal{S}_{t,2} = \{s_{t,2}^1, \ldots, s_{t,2}^{\alpha_t-\beta_t}\}$. Additionally, 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 set $\mathcal{S}_{t,2}^0 \triangleq \emptyset$. Next, order the elements in each $\mathcal{O}_t$ so that $\mathcal{O}_t = \{o_t^1, \ldots, o_t^{\alpha_t-\beta_t}\}$ and if $o_t^i \in \mathcal{S}_{t,2}$, then $o_t^i = s_{t,2}^i$; i.e., order the elements 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: $\mathcal{O}_{1:t} \triangleq (\mathcal{O}_1, \ldots, \mathcal{O}_t)$; $\mathcal{O}_{1:0} \triangleq \emptyset$; $\mathcal{S}_{1:t,2} \triangleq (\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{t,2})$; and $\mathcal{S}_{1:0,2} \triangleq \emptyset$. Then,

$$f(\mathcal{O}_{1:T}) = \sum_{t=1}^{T}\sum_{i=1}^{\alpha_t-\beta_t} f(o_t^i \mid \mathcal{O}_{1:t-1} \cup \mathcal{O}_t^{i-1}) \tag{25}$$

$$\leq \frac{1}{1-c_f}\sum_{t=1}^{T}\sum_{i=1}^{\alpha_t-\beta_t} f(o_t^i \mid \mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}) \tag{26}$$

$$\leq \frac{1}{1-c_f}\sum_{t=1}^{T}\sum_{i=1}^{\alpha_t-\beta_t} f(s_{t,2}^i \mid \mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}) \tag{27}$$

$$\leq \frac{1}{(1-c_f)^2}\sum_{t=1}^{T}\sum_{i=1}^{\alpha_t-\beta_t} f(s_{t,2}^i \mid \mathcal{S}_{1:t-1,2} \cup \mathcal{S}_{t,2}^{i-1}) \tag{28}$$

$$= \frac{1}{(1-c_f)^2} f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}). \tag{29}$$

where eq. (25) holds due to eq. (18); ineq. (26) due to ineq. (4); ineq. (27) holds since $s_{t,2}^i$ is chosen greedily by the algorithm, given $\mathcal{R}_{1:t-1} \cup \mathcal{S}_{t,2}^{i-1}$; ineq. (28) holds for the same reasons as ineq. (26); eq. (29) holds for the same reasons as eq. (25). ∎

Lemma 23. *Consider the sets $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ selected by* RAM*'s lines 3-4. Also, for all $t = 1, \ldots, T$, let in Algorithm 2 be $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $\delta_t = \alpha_t - \beta_t$. Finally, consider $\mathcal{P}_t$ such that $\mathcal{P}_t \subseteq \mathcal{K}_t$, $|\mathcal{P}_t| \leq \delta_t$, and $f(\mathcal{P}_{1:T})$ is maximal, that is,*

$$\mathcal{P}_{1:T} \in \arg\max_{\bar{\mathcal{P}}_1 \subseteq \mathcal{K}_1, |\bar{\mathcal{P}}_1| \leq \delta_1} \cdots \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{K}_T, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\bar{\mathcal{P}}_{1:T}). \tag{30}$$

Then, $f(\mathcal{M}_{1:T}) \geq (1-c_f)f(\mathcal{P}_{1:T})$.

Proof of Lemma 23: The proof is the same as that of [23, Theorem 6]. ∎

Corollary 24. *Consider the sets $\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}$ selected by* RAM*'s lines 3-4, as well as, the sets $\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}$ selected by* RAM*'s lines 5-8. Finally, consider $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $\delta_t = \alpha_t - \beta_t$, and $\mathcal{P}_t$ per eq. (30). Then,*

$$f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) \geq (1-c_f)^3 f(\mathcal{P}_{1:T}).$$

Proof of Corollary 24: Let $\mathcal{O}_t = \mathcal{M}_t$ in ineq. (24). Then,

$$f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) \geq (1-c_f)^2 f(\mathcal{M}_{1:T}). \tag{31}$$

Using in ineq. (31) Lemma 23, the proof is complete. ∎

Lemma 25. *Per the notation in Corollary 24, for all $t = 1, \ldots, T$, consider $\mathcal{K}_t = \mathcal{V}_t \setminus \mathcal{S}_{t,1}$, $\delta_t = \alpha_t - \beta_t$, and $\mathcal{P}_t$ per eq. (30). Then, it holds true that*

$$f(\mathcal{P}_{1:T}) \geq f^\star. \tag{32}$$

Proof of Lemma 25: We use the notation

$$h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}) \triangleq \max_{\bar{\mathcal{P}}_1 \subseteq \mathcal{V}_1 \setminus \mathcal{S}_{1,1}, |\bar{\mathcal{P}}_1| \leq \delta_1} \cdots \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \mathcal{S}_{T,1}, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\bar{\mathcal{P}}_{1:T}). \tag{33}$$

For any $\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_T$ such that $\hat{\mathcal{P}}_t \subseteq \mathcal{V}_t \setminus \mathcal{S}_{t,1}$ and $|\hat{\mathcal{P}}_t| \leq \delta_t$ (for all $t = 1, \ldots, T$), h's definition in eq. (33) implies

$$h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}) \geq f(\hat{\mathcal{P}}_1, \ldots, \hat{\mathcal{P}}_T). \quad (34)$$

Since ineq. (34) holds for any $\hat{\mathcal{P}}_T$ such that $\hat{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \mathcal{S}_{T,1}$ and $|\hat{\mathcal{P}}_T| \leq \delta_T$, then it also holds for the $\hat{\mathcal{P}}_T$ that maximizes the right-hand-side of ineq. (34), i.e.,

$$h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T,1}) \geq \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \mathcal{S}_{T,1}, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\hat{\mathcal{P}}_{1:T-1}, \bar{\mathcal{P}}_T). \quad (35)$$

Treating in ineq. (35) the set $\mathcal{S}_{T,1}$ as a free variable, since (35) holds for any $\mathcal{S}_{T,1} \subseteq \mathcal{V}_T$ such that $|\mathcal{S}_{T,1}| \leq \beta_T$, then

$$\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}, \bar{\mathcal{B}}_T) \geq \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} \max_{\bar{\mathcal{P}}_T \subseteq \mathcal{V}_T \setminus \bar{\mathcal{B}}_T, |\bar{\mathcal{P}}_T| \leq \delta_T} f(\hat{\mathcal{P}}_{1:T-1}, \bar{\mathcal{P}}_T). \quad (36)$$

Specifically, ineq. (36) holds true for the same reason the following holds true: given a set function $f_1 : \mathcal{I} \mapsto \mathbb{R}$ (representing the left-hand-side of ineq. (35)) and a set function $f_2 : \mathcal{I} \mapsto \mathbb{R}$ (representing the right-hand-side of ineq. (35)), where $\mathcal{I} = \{\mathcal{S} : \mathcal{S} \subseteq \mathcal{V}_T, |\mathcal{S}| \leq \beta_T\}$, if $f_1(\mathcal{S}) \geq f_2(\mathcal{S})$ for every $\mathcal{S} \in \mathcal{I}$ (as ineq. (35) defines), then also the minimum of f_1 must be greater or equal to the minimum of f_2, i.e., $\min_{\mathcal{S} \in \mathcal{I}} f_1(\mathcal{S}) \geq \min_{\mathcal{S} \in \mathcal{I}} f_2(\mathcal{S})$. The reason: if $\mathcal{S}_1^\star \in \arg\min_{\mathcal{S} \in \mathcal{I}} f_1(\mathcal{S})$, then $f_1(\mathcal{S}_1^\star) \geq f_2(\mathcal{S}_1^\star)$, since $f_1(\mathcal{S}) \geq f_2(\mathcal{S})$ for every $\mathcal{S} \in \mathcal{I}$; but also $f_2(\mathcal{S}_1^\star) \geq \min_{\mathcal{S} \in \mathcal{I}} f_2(\mathcal{S})$, and, as a result, indeed, $\min_{\mathcal{S} \in \mathcal{I}} f_1(\mathcal{S}) \geq \min_{\mathcal{S} \in \mathcal{I}} f_2(\mathcal{S})$. Changing the symbol of the dummy variable $\mathcal{S}$ to $\bar{\mathcal{B}}_T$, we get ineq. (36).

Denote now the right-hand-side of ineq. (36) by $z(\hat{\mathcal{P}}_{1:T-1})$. Since $\delta_T = \alpha_T - \beta_T$, and for $\bar{\mathcal{P}}_T$ in ineq. (36) 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:

$$z(\hat{\mathcal{P}}_{1:T-1}) = \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} f(\hat{\mathcal{P}}_{1:T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \quad (37)$$

Let in ineq. (37) $w(\bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T) \triangleq f(\hat{\mathcal{P}}_{1:T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T)$; we prove that the following holds true:

$$z(\hat{\mathcal{P}}_{1:T-1}) \geq \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} w(\bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \quad (38)$$

Particularly, for any $\hat{\mathcal{A}}_T \subseteq \mathcal{V}_T$ such that $|\hat{\mathcal{A}}_T| \leq \alpha_T$, and any $\hat{\mathcal{S}}_{T,1} \subseteq \mathcal{V}_T$ such that $|\hat{\mathcal{S}}_{T,1}| \leq \beta_T$, it is

$$\max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} w(\bar{\mathcal{A}}_T \setminus \hat{\mathcal{S}}_{T,1}) \geq w(\hat{\mathcal{A}}_T \setminus \hat{\mathcal{S}}_{T,1}). \quad (39)$$

From ineq. (39), following the same reasoning as for the derivation of ineq. (36) from ineq. (35), considering in ineq. (39) the $\hat{\mathcal{S}}_{T,1}$ to be the free variable, we get

$$\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} w(\bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T) \geq \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} w(\hat{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \quad (40)$$

Now, ineq. (40) implies ineq. (38). The reason: (40) holds for any $\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T$ such that $|\bar{\mathcal{A}}_T| \leq \alpha_T$, while the left-hand-side of (40) is equal to $z(\hat{\mathcal{P}}_{1:T-1})$, which is independent of $\hat{\mathcal{A}}_T$; therefore, if we maximize the right-hand-side of (40) with respect to $\bar{\mathcal{A}}_T$, then indeed we get (38).

All in all, due to ineq. (38), ineq. (36) becomes:

$$\min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}, \bar{\mathcal{B}}_T) \geq \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} f(\hat{\mathcal{P}}_{1:T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T). \quad (41)$$

The left-hand-side of ineq. (41) 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. (41) is a function of $\hat{\mathcal{P}}_{1:T-1}$; denote it as $f'(\hat{\mathcal{P}}_{1:T-1})$. Given these notations, ineq. (41) is equivalently written as

$$h'(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-1,1}) \geq f'(\hat{\mathcal{P}}_{1:T-1}), \quad (42)$$

which has the same form as ineq. (34). Therefore, by following the same steps as those we used from ineq. (34) and onward, we similarly get

$$\min_{\bar{\mathcal{B}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{B}}_{T-1}| \leq \beta_{T-1}} h'(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-2,1}, \bar{\mathcal{B}}_{T-1}) \geq \max_{\bar{\mathcal{A}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{A}}_{T-1}| \leq \alpha_{T-1}} \min_{\bar{\mathcal{B}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{B}}_{T-1}| \leq \beta_{T-1}} f'(\hat{\mathcal{P}}_{1:T-2}, \bar{\mathcal{A}}_{T-1} \setminus \bar{\mathcal{B}}_{T-1}), \quad (43)$$

which, given the definitions of $h'(\cdot)$ and $f'(\cdot)$, is equivalent to

$$\min_{\bar{\mathcal{B}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{B}}_{T-1}| \leq \beta_{T-1}} \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{T-2,1}, \bar{\mathcal{B}}_{T-1}, \bar{\mathcal{B}}_T) \geq \max_{\bar{\mathcal{A}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{A}}_{T-1}| \leq \alpha_{T-1}} \min_{\bar{\mathcal{B}}_{T-1} \subseteq \mathcal{V}_{T-1}, |\bar{\mathcal{B}}_{T-1}| \leq \beta_{T-1}} \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} f(\hat{\mathcal{P}}_{1:T-2}, \bar{\mathcal{A}}_{T-1} \setminus \bar{\mathcal{B}}_{T-1}, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T).$$

Eq. (43) has the same form as ineq. (41). Therefore, repeating the same steps as above for another $T-2$ times (starting now from (43) instead of (41)), we get

$$\min_{\bar{\mathcal{B}}_1 \subseteq \mathcal{V}_1, |\bar{\mathcal{B}}_1| \leq \beta_1} \cdots \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\bar{\mathcal{B}}_{1:T}) \geq \max_{\bar{\mathcal{A}}_1 \subseteq \mathcal{V}_1, |\bar{\mathcal{A}}_1| \leq \alpha_1} \min_{\bar{\mathcal{B}}_1 \subseteq \mathcal{V}_1, |\bar{\mathcal{B}}_1| \leq \beta_1} \cdots \max_{\bar{\mathcal{A}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{A}}_T| \leq \alpha_T} \min_{\bar{\mathcal{B}}_1 \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} f(\bar{\mathcal{A}}_1 \setminus \bar{\mathcal{B}}_1, \ldots, \bar{\mathcal{A}}_T \setminus \bar{\mathcal{B}}_T), \quad (44)$$

which implies ineq. (32) since the right-hand-side of ineq. (44) is equal to the right-hand-side of ineq. (32), while for the left-hand-side of ineq. (44) the following holds:

$$\min_{\bar{\mathcal{B}}_1 \subseteq \mathcal{V}_1, |\bar{\mathcal{B}}_1| \leq \beta_1} \cdots \min_{\bar{\mathcal{B}}_T \subseteq \mathcal{V}_T, |\bar{\mathcal{B}}_T| \leq \beta_T} h(\bar{\mathcal{B}}_{1:T}) \leq f(\mathcal{P}_{1:T}).$$ ■

Appendix B: Proof of Proposition 2

We compute the running time of RAM's line 3 and lines 5-8. Line 3 needs $|\mathcal{V}_t|\tau_f + |\mathcal{V}_t|\log(|\mathcal{V}_t|) + |\mathcal{V}_t| + O(\log(|\mathcal{V}_t|))$ time: 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: the while loop is repeated $\alpha_t - \beta_t$ times, and during each loop at most $|\mathcal{V}_t|$ evaluations of f are needed (line 5), plus at most $|\mathcal{V}_t|$ steps for a maximal element to be found (line 6). Hence, RAM runs at each t 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.

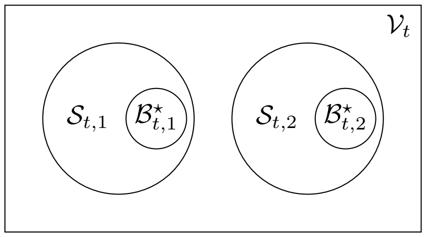

Fig. 3. Venn diagram, where the sets $\mathcal{S}_{t,1}, \mathcal{S}_{t,2}, \mathcal{B}^\star_{t,1}, \mathcal{B}^\star_{t,2}$ are as follows: per RAM, $\mathcal{S}_{t,1}$ and $\mathcal{S}_{t,2}$ are such that $\mathcal{A}_t = \mathcal{S}_{t,1} \cup \mathcal{S}_{t,2}$. Additionally, due to their construction, $\mathcal{S}_{t,1} \cap \mathcal{S}_{t,2} = \emptyset$. Next, $\mathcal{B}^\star_{t,1}$ and $\mathcal{B}^\star_{t,2}$ are such that $\mathcal{B}^\star_{t,1} = \mathcal{B}^\star_{1:T} \cap \mathcal{S}_{t,1}$, and $\mathcal{B}^\star_2 = \mathcal{B}^\star_{1:T} \cap \mathcal{S}_{t,2}$; therefore, $\mathcal{B}^\star_{t,1} \cap \mathcal{B}^\star_{t,2} = \emptyset$ and $\mathcal{B}^\star_{1:T} = (\mathcal{B}^\star_{1,1} \cup \mathcal{B}^\star_{1,2}) \cup \cdots \cup (\mathcal{B}^\star_{T,1} \cup \mathcal{B}^\star_{T,2})$.

Appendix C: Proof of Theorem 10

We first prove ineq. (6) and then (5). We use the notation:

- $\mathcal{S}^+_{t,1} \triangleq \mathcal{S}_{t,1} \setminus \mathcal{B}^\star_t$, i.e., $\mathcal{S}^+_{t,1}$ is the remaining set after the optimal (worst-case) removal $\mathcal{B}^\star_t$;
- $\mathcal{S}^+_{t,2} \triangleq \mathcal{S}_{t,2} \setminus \mathcal{B}^\star_t$;
- $\mathcal{P}_{1:T}$ be a solution to eq. (30).

Proof of ineq. (6)*:* For $T > 1$, we have:

$$\begin{aligned}
&f(\mathcal{A}_{1:T} \setminus \mathcal{B}^\star_{1:T})\\
&\quad= f(\mathcal{S}^+_{1,1} \cup \mathcal{S}^+_{1,2}, \ldots, \mathcal{S}^+_{T,1} \cup \mathcal{S}^+_{T,2}) && (45)\\
&\quad\geq (1-c_f)\sum_{t=1}^{T}\sum_{v \in \mathcal{S}^+_{t,1}\cup\mathcal{S}^+_{t,2}} f(v) && (46)\\
&\quad\geq (1-c_f)\sum_{t=1}^{T}\sum_{v \in \mathcal{S}_{t,2}} f(v) && (47)\\
&\quad\geq (1-c_f)^2 f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) && (48)\\
&\quad\geq (1-c_f)^5 f(\mathcal{P}_{1:T}) && (49)\\
&\quad\geq (1-c_f)^5 f^\star, && (50)
\end{aligned}$$

where eq. (45) follows from the definitions of $\mathcal{S}^+_{t,1}$ and $\mathcal{S}^+_{t,2}$; ineq. (46) follows from ineq. (45), due to Lemma 19; ineq. (47) follows from ineq. (46), because: for all $v \in \mathcal{S}^+_{t,1}$ and $v' \in \mathcal{S}_{t,2} \setminus \mathcal{S}^+_{t,2}$ it is $f(v) \geq f(v')$, and $\mathcal{S}_{t,2} = (\mathcal{S}_{t,2} \setminus \mathcal{S}^+_{t,2}) \cup \mathcal{S}^+_{t,2}$; ineq. (48) follows from ineq. (47) due to Corollary 21; ineq. (49) follows from ineq. (48) due to Corollary 24; finally, ineq. (50) follows from ineq. (49) due to Lemma 25.

For $T = 1$, the proof follows the same steps up to ineq. (48), at which point $f(\mathcal{S}_{1,2}) \geq (1-c_f) f(\mathcal{P}_1)$ instead, due to Lemma 23 (since $\mathcal{S}_{1,2} = \mathcal{M}_1$). ■

Proof of ineq. (5)*:* For $T > 1$ we follow similar steps:

$$\begin{aligned}
&f(\mathcal{A}_{1:T} \setminus \mathcal{B}^\star_{1:T})\\
&\quad= f(\mathcal{S}^+_{1,1} \cup \mathcal{S}^+_{1,2}, \ldots, \mathcal{S}^+_{T,1} \cup \mathcal{S}^+_{T,2}) && (51)\\
&\quad\geq (1-\kappa_f)\sum_{t=1}^{T}\sum_{v \in \mathcal{S}^+_{t,1}\cup\mathcal{S}^+_{t,2}} f(v) && (52)\\
&\quad\geq (1-\kappa_f)\sum_{t=1}^{T}\sum_{v \in \mathcal{S}_{t,2}} f(v) && (53)\\
&\quad\geq (1-\kappa_f) f(\mathcal{S}_{1,2}, \ldots, \mathcal{S}_{T,2}) && (54)\\
&\quad\geq (1-\kappa_f)^4 f(\mathcal{P}_{1:T}) && (55)\\
&\quad\geq (1-\kappa_f)^4 f^\star, && (56)
\end{aligned}$$

where eq. (51) follows from the definitions of $\mathcal{S}^+_{t,1}$ and $\mathcal{S}^+_{t,2}$; ineq. (52) follows from ineq. (51) due to Lemma 18 and the fact that $c_f = \kappa_f$ for f being submodular; ineq. (53) follows from ineq. (52) because for all $v \in \mathcal{S}^+_{t,1}$ and $v' \in \mathcal{S}_{t,2} \setminus \mathcal{S}^+_{t,2}$ it is $f(v) \geq f(v')$, while $\mathcal{S}_{t,2} = (\mathcal{S}_{t,2} \setminus \mathcal{S}^+_{t,2}) \cup \mathcal{S}^+_{t,2}$; ineq. (54) follows from ineq. (53) because f is submodular and, as a result, $f(\mathcal{S}) + f(\mathcal{S}') \geq f(\mathcal{S} \cup \mathcal{S}')$, for any $\mathcal{S}, \mathcal{S}' \subseteq \mathcal{V}$ [64, Proposition 2.1]; ineq. (55) follows from ineq. (54) due to Corollary 24, along with the fact that since f is monotone submodular it is $c_f = \kappa_f$; finally, ineq. (56) follows from ineq. (55) due to Lemma 25.

For $T = 1$, the proof follows the same steps up to ineq. (54), at which point $f(\mathcal{S}_{1,2}) \geq 1/\kappa_f(1 - e^{-\kappa_f}) f(\mathcal{P}_1)$, due to [27, Theorem 5.4]. ■

Appendix D: Proof of Theorem 13

To prove ineq. (7), we have

$$\begin{aligned}
&f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})\\
&\quad= f(\mathcal{M}_{1:t})\frac{f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})}{f(\mathcal{M}_{1:t})}\\
&\quad\geq \begin{cases} \frac{1-e^{-\kappa_f}}{\kappa_f}\frac{f(\mathcal{A}_1 \setminus \mathcal{B}^\star_1)}{f(\mathcal{M}_1)} f(\mathcal{P}_1), & t = 1;\\ \frac{1}{1+\kappa_f}\frac{f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})}{f(\mathcal{M}_{1:t})} f(\mathcal{P}_{1:t}), & t > 1, \end{cases} && (57)\\
&\quad\geq \begin{cases} \frac{1-e^{-\kappa_f}}{\kappa_f}\frac{f(\mathcal{A}_1 \setminus \mathcal{B}^\star_1)}{f(\mathcal{M}_1)} f^\star_1, & t = 1;\\ \frac{1}{1+\kappa_f}\frac{f(\mathcal{A}_{1:t} \setminus \mathcal{B}^\star_{1:t})}{f(\mathcal{M}_{1:t})} f^\star_t, & t > 1, \end{cases} && (58)
\end{aligned}$$

where ineq. (57) holds since [27, Theorem 5.4] implies $f(\mathcal{M}_1) \geq 1/\kappa_f(1 - e^{-\kappa_f}) f(\mathcal{P}_1)$, while [27, Theorem 2.3] implies $f(\mathcal{M}_{1:t}) \geq 1/(1+\kappa_f) f(\mathcal{P}_{1:t})$. Finally, ineq. (58) is proved following the same steps as in Lemma 25's proof.

The proof of ineq. (8) follows similar steps as above but it is based instead on Lemma 23.

Appendix E: Proof of Theorem 14

It can be verified that for $t = 1$ eq. (7) is tight for any $\beta_t \leq \alpha_t$ for the families of functions in [27, Theorem 5.4], and for $t > 1$ it is tight for the families of functions in [27, Theorem 2.12]. Similarly, it can be verified eq. (8) is optimal for the families of functions in [23, Theorem 8] for $\alpha_t = |\mathcal{V}_t|^{1/2}$ and any $\beta_t \leq \alpha_t - |\mathcal{V}_t|^{1/3}$.

Acknowledgements

We thank Rakesh Vohra of the University of Pennsylvania, and Luca Carlone of the Massachusetts Institute of Technology for inspiring comments and discussions.

References

[1] T. Abdelzaher, N. Ayanian, T. Basar, S. Diggavi, J. Diesner, D. Ganesan, R. Govindan, S. Jha, T. Lepoint, B. Marlin *et al.*, "Will distributed computing revolutionize peace? The emergence of Battlefield IoT," in *IEEE 38th International Conference on Distributed Computing Systems*, 2018, pp. 1129–1138.

[2] L. Carlone and S. Karaman, "Attention and anticipation in fast visual-inertial navigation," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 1–20, 2018.

[3] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. A. Stankovic, and T. Abdelzaher, "Achieving real-time target tracking usingwireless sensor networks," in *IEEE Real-Time and Embedded Technology and Applications Symposium*, 2006, pp. 37–48.

[4] H. H. Yang, Z. Liu, T. Q. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *arXiv preprint:1908.06287*, 2019.

[5] 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.

[6] 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.

[7] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, *Submodularity in dynamics and control of networked systems*. Springer, 2016.

[8] 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.

[9] 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.

[10] V. Tzoumas, L. Carlone, G. J. Pappas, and A. Jadbabaie, "LQG Control and Sensing Co-design," *arXiv preprints:1802.08376*, 2018.

[11] 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.

[12] 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.

[13] T. Ikeda and K. Kashima, "Sparsity-constrained controllability maximization with application to time-varying control node selection," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 321–326, 2018.

[14] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, C. Faloutsos, J. Van-Briesen, and N. Glance, "Cost-effective outbreak detection in networks," in *ACM SIGKDD international conference on Knowledge discovery and data mining*, 2007, pp. 420–429.

[15] 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.

[16] 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.

[17] U. Feige, "A threshold of $ln(n)$ for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[18] 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.

[19] D. Foster, H. Karloff, and J. Thaler, "Variable selection is hard," in *Conference on Learning Theory*, 2015, pp. 696–709.

[20] L. Ye, S. Roy, and S. Sundaram, "On the complexity and approximability of optimal sensor selection for Kalman filtering," in *American Control Conference*, 2018, pp. 5049–5054.

[21] 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.

[22] Z. Wang, B. Moran, X. Wang, and Q. Pan, "Approximation for maximizing monotone non-decreasing set functions with a greedy method," *Journal of Combinatorial Optimization*, vol. 31, no. 1, pp. 29–43, 2016.

[23] 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.

[24] T. Abdelzaher, N. Ayanian, T. Basar, S. Diggavi, J. Diesner, D. Ganesan, R. Govindan, S. Jha, T. Lepoint, B. Marlin *et al.*, "Toward an Internet of Battlefield Things: A resilience perspective," *Computer*, vol. 51, no. 11, pp. 24–36, 2018.

[25] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks," *Computer*, vol. 35, no. 10, pp. 54–62, 2002.

[26] R. B. Myerson, *Game theory*. Harvard University Press, 2013.

[27] M. Conforti and G. Cornuéjols, "Submodular set functions, matroids and the greedy algorithm," *Discrete Applied Mathematics*, vol. 7, no. 3, pp. 251 – 274, 1984.

[28] 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.

[29] A. Krause and V. Cevher, "Submodular dictionary selection for sparse representation," in *International Conference on Machine Learning*, 2010.

[30] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," *arXiv preprint:1311.4728*, 2013.

[31] J. B. Orlin, A. S. Schulz, and R. Udwani, "Robust monotone submodular function maximization," in *International Conference on Integer Programming and Combinatorial Optimization*, 2016, pp. 312–324.

[32] I. Bogunovic, S. Mitrović, J. Scarlett, and V. Cevher, "Robust submodular maximization: A non-uniform partitioning approach," in *International Conference on Machine Learning*, 2017, pp. 508–516.

[33] 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.

[34] A. Rahmattalabi, P. Vayanos, and M. Tambe, "A robust optimization approach to designing near-optimal strategies for constant-sum monitoring games," in *International Conference on Decision and Game Theory for Security*, 2018, pp. 603–622.

[35] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Resilient non-submodular maximization over matroid constraints," *arXiv preprint:1804.01013*, 2018.

[36] I. Bogunovic, J. Zhao, and V. Cevher, "Robust maximization of non-submodular objectives," in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 890–899.

[37] B. Schlotfeldt, V. Tzoumas, D. Thakur, and G. J. Pappas, "Resilient active information gathering with mobile robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 4309–4316.

[38] L. Zhou, V. Tzoumas, G. J. Pappas, and P. Tokekar, "Resilient active target tracking with multiple robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 129–136, 2018.

[39] 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.

[40] 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.

[41] E. Kazemi, M. Zadimoghaddam, and A. Karbasi, "Deletion-robust submodular maximization at scale," *ArXiv preprints:1711.07112*, 2017.

[42] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and fault-tolerant control*. Springer, vol. 2.

[43] L. F. Cómbita, A. A. Cárdenas, and N. Quijano, "Mitigating sensor attacks against industrial control systems," *IEEE Access*, vol. 7, pp. 92 444–92 455, 2019.

[44] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security*, vol. 14, no. 1, p. 13, 2011.

[45] M. Jin, J. Lavaei, and K. H. Johansson, "Power grid ac-based state estimation: Vulnerability analysis against cyber attacks," *IEEE Transactions on Automatic Control*, vol. 64, no. 5, pp. 1784–1799, 2018.

[46] A. Clark and L. Niu, "Linear quadratic gaussian control under false data injection attacks," in *Annual American Control Conference*, 2018, pp. 5737–5743.

[47] Q. Zhu and T. Basar, "Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: Games-in-games principle for optimal cross-layer resilient control systems," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 46–65, 2015.

[48] X. Jin, W. M. Haddad, and T. Yucelen, "An adaptive control architecture for mitigating sensor and actuator attacks in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 11, pp. 6058–6064, 2017.

[49] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE transactions on automatic control*, vol. 58, no. 11, pp. 2715–2729, 2013.

[50] L. S. Perelman, W. Abbas, X. Koutsoukos, and S. Amin, "Sensor placement for fault location identification in water networks: A minimum test cover approach," *Automatica*, vol. 72, pp. 166–176, 2016.

[51] Y. Shoukry, P. Nuzzo, A. Puggelli, A. L. Sangiovanni-Vincentelli, S. A. Seshia, and P. Tabuada, "Secure state estimation for cyber-physical systems under sensor attacks: A satisfiability modulo theory approach,"

IEEE Transactions on Automatic Control, vol. 62, no. 10, pp. 4917–4932, 2017.

[52] M. Pajic, J. Weimer, N. Bezzo, P. Tabuada, O. Sokolsky, I. Lee, and G. J. Pappas, "Robustness of attack-resilient state estimators," in *ACM/IEEE 5th International Conference on Cyber-Physical Systems*, 2014, pp. 163–174.

[53] A. Kanellopoulos and K. G. Vamvoudakis, "A moving target defense control framework for cyber-physical systems," *IEEE Transactions on Automatic Control*, pp. 1–1, 2019, in press.

[54] Y. Mo, S. Weerakkody, and B. Sinopoli, "Physical authentication of control systems: Designing watermarked control inputs to detect counterfeit sensor outputs," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 93–109, 2015.

[55] J. Usevitch and D. Panagou, "Resilient leader-follower consensus to arbitrary reference values in time-varying graphs," *IEEE Transactions on Automatic Control*, pp. 1–1, 2019, in press.

[56] G. De La Torre, T. Yucelen, and J. D. Peterson, "Resilient networked multiagent systems: A distributed adaptive control approachy," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 5367–5372.

[57] L. Su and S. Shahrampour, "Finite-time guarantees for Byzantine-resilient distributed state estimation with noisy measurements," *arXiv preprint:1810.10086*, 2018.

[58] Y. Chen, S. Kar, and J. Moura, "Resilient distributed estimation: Sensor attacks," *IEEE Transactions on Automatic Control*, 2018.

[59] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, "Foundations of control and estimation over lossy networks," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 163–187, 2007.

[60] S. Amin, A. A. Cárdenas, and S. S. Sastry, "Safe and secure networked control systems under denial-of-service attacks," in *International Workshop on Hybrid Systems: Computation and Control*, 2009, pp. 31–45.

[61] A.-Y. Lu and G.-H. Yang, "Input-to-state stabilizing control for cyber-physical systems with multiple transmission channels under denial of service," *IEEE Transactions on Automatic Control*, vol. 63, no. 6, pp. 1813–1820, 2017.

[62] V. Tzoumas, A. Jadbabaie, and G. J. Pappas, "Resilient monotone sequential maximization," in *IEEE Conference on Decision and Control*, 2018, pp. 7261–7268.

[63] ——, "Resilient monotone sequential maximization," *arXiv preprint:1803.07954*, 2018.

[64] 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.

[65] B. Lehmann, D. Lehmann, and N. Nisan, "Combinatorial auctions with decreasing marginal utilities," *Games and Economic Behavior*, vol. 55, no. 2, pp. 270–296, 2006.

[66] E. R. Elenberg, R. Khanna, A. G. Dimakis, S. Negahban *et al.*, "Restricted strong convexity implies weak submodularity," *The Annals of Statistics*, vol. 46, no. 6B, pp. 3539–3568, 2018.

[67] L. F. Chamon and A. Ribeiro, "Near-optimality of greedy set selection in the sampling of graph signals," in *IEEE Global Conference on Signal and Information Processing*, 2016, pp. 1265–1269.

[68] B. Guo, O. Karaca, T. Summers, and M. Kamgarpour, "Actuator placement for optimizing network performance under controllability constraints," *arXiv preprint:1903.08120*, 2019.

[69] D. Sharma, A. Kapoor, and A. Deshpande, "On greedy maximization of entropy," in *Inter. Conf. on Machine Learning*, 2015, pp. 1330–1338.

[70] L. F. Chamon, G. J. Pappas, and A. Ribeiro, "The mean square error in kalman filtering sensor selection is approximately supermodular," in *IEEE 56th Annual Conference on Decision and Control*, 2017, pp. 343–350.

[71] 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.

[72] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, pp. 235–284, 2008.

[73] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time," *Journal of Combinatorial Theory, Series B*, vol. 80, no. 2, pp. 346–355, 2000.

[74] S. Iwata and J. B. Orlin, "A simple combinatorial algorithm for submodular function minimization," in *ACM-SIAM symposium on Discrete algorithms*, 2009, pp. 1230–1237.

[75] Y. T. Lee, A. Sidford, and S. C.-w. Wong, "A faster cutting plane method and its implications for combinatorial and convex optimization," in *IEEE 56th Annual Symposium on Foundations of Computer Science*, 2015, pp. 1049–1065.

[76] D. Chakrabarty, Y. T. Lee, A. Sidford, and S. C.-W. Wong, "Subquadratic submodular function minimization," in *49th Annual ACM SIGACT Symposium on Theory of Computing*, 2017, pp. 1220–1231.

[77] M. E. Halabi and S. Jegelka, "Minimizing approximately submodular functions," *arXiv preprint:1905.12145*, 2019.

[78] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[79] L. Ye, S. Roy, and S. Sundaram, "Resilient sensor placement for kalman filtering in networked systems: Complexity and algorithms," *IEEE Transactions on Control of Network Systems*, 2020.

[80] A. Krause, H. B. McMahan, C. Guestrin, and A. Gupta, "Robust submodular observation selection," *Journal of Machine Learning Research*, vol. 9, pp. 2761–2801, 2008.

[81] A. Rahmattalabi, P. Vayanos, A. Fulginiti, E. Rice, B. Wilder, A. Yadav, and M. Tambe, "Exploring algorithmic fairness in robust graph covering problems," in *Advances in Neural Information Processing Systems*, 2019, pp. 15 776–15 787.

[82] D. Lu, Y. Qu, F. Wu, H. Dai, C. Dong, and G. Chen, "Robust server placement for edge computing," in *IEEE International Parallel and Distributed Processing Symposium*, 2020, pp. 285–294.

[83] 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.

Vasileios Tzoumas received his Ph.D. in Electrical and Systems Engineering at the University of Pennsylvania (2018). He holds a Master of Arts in Statistics from the Wharton School of Business at the University of Pennsylvania (2016); a Master of Science in Electrical Engineering from the University of Pennsylvania (2016); and a diploma in Electrical and Computer Engineering from the National Technical University of Athens (2012). Vasileios is as an Assistant Professor in the Department of Aerospace Engineering, University of Michigan, Ann Arbor. Previously, he was at the Massachusetts Institute of Technology (MIT), in the Department of Aeronautics and Astronautics, and in the Laboratory for Information and Decision Systems (LIDS), were he was a research scientist (2019-2020), and a post-doctoral associate (2018-2019). Vasileios was a visiting Ph.D. student at the Institute for Data, Systems, and Society (IDSS) at MIT during 2017. Vasileios works on control, learning, and perception, as well as combinatorial and distributed optimization, with applications to robotics, cyber-physical systems, and self-reconfigurable aerospace systems. He aims for trustworthy collaborative autonomy. His work includes foundational results on robust and adaptive combinatorial optimization, with applications to multi-robot information gathering for resiliency against robot failures and adversarial removals. Vasileios is a recipient of the Best Paper Award in Robot Vision at the 2020 IEEE International Conference on Robotics and Automation (ICRA), and was a Best Student Paper Award Finalist at the 2017 IEEE Conference in Decision and Control (CDC).

Ali Jadbabaie (S'99-M'08-SM'13-F'15) is the JR East Professor of Engineering and Associate Director of the Institute for Data, Systems and Society at MIT, where he is also on the faculty of the department of civil and environmental engineering and a principal investigator in the Laboratory for Information and Decision Systems (LIDS). He is the director of the Sociotechnical Systems Research Center, one of MIT's 13 laboratories. He received his Bachelors (with high honors) from Sharif University of Technology in Tehran, Iran, a Masters degree in electrical and computer engineering from the University of New Mexico, and his Ph.D. in control and dynamical systems from the California Institute of Technology. He was a postdoctoral scholar at Yale University before joining the faculty at Penn in July 2002. Prior to joining MIT faculty, he was the Alfred Fitler Moore a Professor of Network Science and held secondary appointments in computer and information science and operations, information and decisions in the Wharton School. He was the inaugural editor-in-chief of IEEE Transactions on Network Science and Engineering, a new interdisciplinary journal sponsored by several IEEE societies. He is a recipient of a National Science Foundation Career Award, an Office of Naval Research Young Investigator Award, the O. Hugo Schuck Best Paper Award from the American Automatic Control Council, and the George S. Axelby Best Paper Award from the IEEE Control Systems Society. His students have been winners and finalists of student best paper awards at various ACC and CDC conferences. He is an IEEE fellow and a recipient of the Vannevar Bush Fellowship from the office of Secretary of Defense. His current research interests include the interplay of dynamic systems and networks with specific emphasis on multi-agent coordination and control, distributed optimization, network science, and network economics.

George J. Pappas (S'90-M'91-SM'04-F'09) received the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA, in 1998. He is currently the Joseph Moore Professor and Chair of the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA. He also holds a secondary appointment with the Department of Computer and Information Sciences and the Department of Mechanical Engineering and Applied Mechanics. He is a Member of the GRASP Lab and the PRECISE Center. He had previously served as the Deputy Dean for Research with the School of Engineering and Applied Science. His research interests include control theory and, in particular, hybrid systems, embedded systems, cyberphysical systems, and hierarchical and distributed control systems, with applications to unmanned aerial vehicles, distributed robotics, green buildings, and biomolecular networks. Dr. Pappas has received various awards, such as the Antonio Ruberti Young Researcher Prize, the George S. Axelby Award, the Hugo Schuck Best Paper Award, the George H. Heilmeier Award, the National Science Foundation PECASE award and numerous best student papers awards.