Artificial Intelligence 297 (2021) 103486

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

A unifying look at sequence submodularity ☆

Sara Bernardini [a,*], Fabio Fagnani [b], Chiara Piacentini [c]

[a] Department of Computer Science, Royal Holloway University of London, Egham, Surrey, TW20 0EX, UK
[b] Department of Mathematical Sciences, Politecnico di Torino, Torino, 10129, Italy
[c] Augmenta Inc., Toronto, M5A 1E1, Canada

ARTICLE INFO

Article history:
Received 11 September 2020
Received in revised form 9 January 2021
Accepted 16 February 2021
Available online 24 February 2021

Keywords:
Submodularity
Sequence submodularity
Greedy algorithms
Suboptimal algorithms
Detection problems
Search-and-tracking
Environmental monitoring
Scheduling
Recommender systems

ABSTRACT

Several real-world problems in engineering and applied science require the selection of *sequences* that maximize a given reward function. Optimizing over sequences as opposed to sets requires exploring an exponentially larger search space and can become prohibitive in most cases of practical interest. However, if the objective function is submodular (intuitively, it exhibits a diminishing return property), the optimization problem becomes more manageable. Recently, there has been increasing interest in *sequence submodularity* in connection with applications such as recommender systems and online ad allocation. However, mostly ad hoc models and solutions have emerged within these applicative contexts. In consequence, the field appears fragmented and lacks coherence. In this paper, we offer a unified view of sequence submodularity and provide a generalized greedy algorithm that enjoys strong theoretical guarantees. We show how our approach naturally captures several application domains, and our algorithm encompasses existing methods, improving over them.

1. Introduction

Many real-world applications in engineering and applied science have at their core the selection of *sequences* of objects that maximize a reward. In information gathering missions, for example, the objects are observations and the goal is to select a sequence of them that maximizes the information gain [1,2]. In a similar fashion, a movie recommender system aims to provide its users with sequences of items that maximize relevance [3,4]. The crucial point in these applications is that the value of the sequence depends not only on the objects belonging to it, but also on their relative *order*. This is because the value of each object changes based on its position in the sequence.

If optimizing over sets is already a daunting task, optimizing over sequences quickly becomes unmanageable when the problem at hand grows. However, the identification of special properties in the objective function helps in making the task more approachable. *Submodularity*, in particular, has emerged as a powerful feature that can be leveraged to control complexity in the maximization of both set and sequence functions. Submodularity can be understood intuitively as a *diminishing return* condition. Consider again an information-gathering mission. Each new observation increases the information gain, but it does it to a smaller extent than the previous observations, with gain vanishing at infinity.

In areas as variegated as optimization, machine learning, economics, medicine and sensor networks, there has been a vast amount of work on the maximization of submodular set functions (see Section 2). Only recently, the scientific commu-

☆ This paper is an invited revision of a paper which first appeared at the 2020 International Conference Automated Planning and Scheduling (ICAPS-20).

* Corresponding author.
E-mail address: sara.bernardini@rhul.ac.uk (S. Bernardini).

https://doi.org/10.1016/j.artint.2021.103486

nity has started to pay closer attention to *sequence submodularity* prompted by applications such as online ad allocation [5] and recommendations in online shopping [6], entertainment [3] and courses [7]. However, having arisen in specific applicative contexts, the proposed models as well as the corresponding algorithms lack generality and require making restrictive assumptions on the objective function to maintain efficiency.

In this paper, to remedy the current ad hoc approach and lack of coherence in the field, we offer a unified view of sequence submodularity. By abstracting away specific applicative details, we show that the optimization problem that lies behind several applications can be captured by a particular type of *recursive submodular function*. We study its structure and, based on its properties, we propose a *generalized greedy algorithm* that has theoretical guarantees as strong as its classical counterpart on set functions but does not require unrealistic restrictive assumptions. Our generalized algorithm encompasses and improves the specific algorithms that have been developed for several practical applications. Another property that confers *flexibility* to our approach is that we can easily enforce constraints on the cardinality of the elements in the sequence (e.g. all elements must be distinct) in the domain description, which is particularly useful in applicative problems.

The paper is organized as follows. After discussing related work in Section 2, we state the problem formally and introduce our running example in Section 3. In Section 4, we recall the concept of submodularity for sequence functions and show how, in general, a simple generalization of the classical greedy algorithm from sets to sequences fails to achieve good performance for several optimization problems of practical relevance. Subsequently, in Section 5, we propose and analyze a new greedy algorithm that is proven to achieve the same performance as the classical one for submodular set functions (Theorem 1). In Section 6, we study how this result can be applied to the general class of problems that we are interested in solving (Theorem 2 and Corollary 1) and, in Sections 7 and 8, we present several different application domains, which demonstrate the expressiveness and generality of our approach. Finally, Section 9 provides explicit numerical simulations for two of the applicative setups discussed in the previous two sections, while Sections 10 offers conclusive thoughts.

2. Related work

Work on submodularity spreads across multiple fields, including optimization [8,9], machine learning [10,11], economics [12,13], medicine [14] and sensor networks [15,16]. This body of work focuses on *set functions* and, as most of the problems considered are NP-complete, revolves around finding good approximations of the optimal solution via greedy approaches, which are very effective for non-decreasing, submodular functions [9]. We do not review this literature here as set functions are not our focus. For a comprehensive review on this topic, we refer the readers to the literature [17].

Only recently, work on sequence submodularity has emerged. Streeter and Golovin [18] first considered this problem in the context of online resource allocation applications. Shortly after, Alaei and Malekian [5] introduced the term *sequence submodularity* and showed that if the submodular function is non-decreasing and differentiable, a greedy approach always achieves a solution that is at least $1 - \frac{1}{e}$ of the optimal one for the maximization problem.

Zhang et al. [15] consider *string submodularity*, which is a weaker concept as the submodularity holds for the prefix relationship instead of for any type of subsequence relationship. They improve on Alaei and Malekian's approximation by introducing additional constraints on the degree of string submodularity *(curvature)* of the objective function.

Other authors have defined sequence submodularity within a *graph*-based setting. Tschiatschek et al. [4] consider cases in which dependencies between elements of a sequence can be captured via directed acyclic graphs (DAGs) and present an algorithm with theoretical guarantees for them. However, repetitions in the sequence are not allowed and DAG submodular functions are not necessarily string or sequence submodular.

Mitrovic et al. [7] extend this graph-based framework to graphs and hypergraphs with bounded in or out degrees.

Finally, Qian et al. [19] take a departure from the greedy approach and propose a Pareto optimization method for sequence selection. They show that, for any class of submodular functions previously studied, their approach can always reach the best known approximation guarantee.

Mitrovic et al. [20], on the other hand, consider the case in which the value of a sequence depends not only on the items selected and their order but also on the states of the items, which might be initially unknown (*adaptive* submodularity).

Against the backdrop of this body of work, we aim to show that the submodular functions appearing in practical applications do not satisfy the constraints imposed by the approaches highlighted here. However, they do present a common structure that can be exploited to equip a suitably modified greedy algorithm with strong theoretical guarantees.

3. Problem statement

In this section, we formally introduce the optimization problems that we study in this paper. Let Ω be a set and $\mathbb{H}(\Omega)$ be the language over Ω, i.e. the set of sequences of elements in Ω of any length including the empty sequence $\varnothing$. Let $\mathbb{H}^d(\Omega)$ denote the sub-language consisting of all sequences in $\mathbb{H}(\Omega)$ with distinct elements. If $S = (S_1, \dots, S_n) \in \mathbb{H}(\Omega)$, with S_i being the element of sequence S in position i, we denote with $|S| = n$ the length of the sequence S. Given $R, S \in \mathbb{H}(\Omega)$, we say that R is a *subsequence* of S (denoted $R \leq S$) if R is obtained from S by eliminating some of its elements, i.e. if there exists a strictly increasing function $\rho : \{1, \dots, |R|\} \to \{1, \dots, |S|\}$ such that $R_i = S_{\rho(i)}$ for every $i = 1, \dots, |R|$. We use the following convention to indicate a specific type of subsequences: if $S = (S_1, \dots, S_n) \in \mathbb{H}(\Omega)$ and $1 \leq a \leq b \leq n$, we write $S|_a^b = (S_a, S_{a+1}, \dots, S_b)$. We put $S|_a^b = \varnothing$, if $a > b$.

In this paper, we focus on greedy algorithms for maximizing functions defined on $\mathbb{H}(\Omega)$ that present the following *recursive* form:

$$F_g(S) = \sum_{k=1}^{n} g(S_k)\left[F(S|_1^k) - F(S|_1^{k-1})\right] \tag{1}$$

for $S = (S_1, \dots, S_n) \in \mathbb{H}(\Omega)$. In Eq. (1), $g : \Omega \to \mathbb{R}^+$ is any function and $F : \mathbb{H}(\Omega) \to \mathbb{R}$ is a function independent from the specific order of the elements in S, monotonic and submodular (formal definitions are given in the next section).

The problem of maximizing these functions (typically on finite sequences with length below a given value) is significant because lies at the heart of several practical applications, ranging from jobs scheduling to web recommendation systems, as we will see in what follows. Note that, for a general g, the functions F_g depends on the specific order of the elements of the sequence, with the consequence that the classical results on submodularity of set functions cannot be applied.

3.1. Running example

We now discuss an illustrative scenario that we will use throughout the paper as a running example. Consider the monitoring of the level of a river subject to flooding. In case of danger, a set of movable floodgates are used to protect roads, bridges and other critical points whose impairment could lead to catastrophic events. Actioning and using these barriers over time and at different points on the river can be costly. A sensing system, e.g. a drone, can be used to observe a specific point more closely to establish whether a floodgate is needed, with a consequent reduction in cost if the drone concludes that the gate is unnecessary to protect that point. We are interested in the problem of finding the sequence of drone observations that allows the maximum reduction in cost. We formalize this scenario as follows.

Given a time horizon $[0, T]$, consider the monitoring of an environment subject to several catastrophic events, which are represented as the elements of a set $\mathcal{D}$. Each of these possible events requires to keep a safety infrastructure system in place (the movable floodgates, in the example above). For simplicity, given an event d in $\mathcal{D}$, we set the cost of the infrastructure to prevent d to one per time unit. In the absence of other information, the total cost of monitoring each event over the time horizon $[0, T]$ is T and, as there are $|\mathcal{D}|$ events, the total cost of monitoring the environment amounts to $T|\mathcal{D}|$. However, over time, the monitoring system acquires information that can be used to rule out the happening of some of the catastrophic events in $\mathcal{D}$. Specifically, we assume that the system can perform a set Ω of experiments. Each $\sigma \in \Omega$ has an associated cost $c > 0$ (in the example above, the cost is associated to the use of a drone to make the observation σ) and an associated time $t(\sigma) \in [0, T]$, representing the moment at which the experiment can be performed. Each experiment $\sigma \in \Omega$ is also associated with a subset of the events $\mathcal{D}_\sigma \subseteq \mathcal{D}$ with the following meaning: if σ gives a negative result, it can be inferred that none of the events in $\mathcal{D}_\sigma$ will take place. In consequence, the system can stop using the safety infrastructure meant to prevent the events in $\mathcal{D}_\sigma$ from the time of the experiment $t(\sigma)$ to the end of the horizon, with a consequence reduction of $(T - t(\sigma))|\mathcal{D}_\sigma|$ from the monitoring cost. A positive result of the experiment σ, instead, does not allow the system to rule out any possible event so the monitoring process does not undergo any change.

Over time, the monitoring system performs a sequence of experiments $S = (S_1, \dots, S_n)$, which are ordered in such a way that $t(S_1) < t(S_2) < \dots < t(S_n)$. Let us now calculate the total monitoring cost when no catastrophic event takes place, assuming that, in this case, all experiments will give a negative result. We first define $d(S) = |\cup_{i=1}^{n} \mathcal{D}_{S_i}|$, which is the number of events ruled out by the sequence of experiments and $d = |\mathcal{D}|$. The total cost is given by the following expression:

$$\Gamma(S) = \sum_{k=1}^{n} (t(S_k) - t(S_{k-1}))(d - d(S_1^{k-1})) + (T - t(S_n))(d - d(S_1^n)) + cn \tag{2}$$

interpreting $t(S_0) = 0$. The terms in Eq. (2) have the following interpretation. The expression $d - d(S_1^{k-1})$ is the number of events not yet ruled out at time $t(S_{k-1})$. Since the system needs to keep monitoring those events, we have to account for the cost per time they generate over the interval $[t(S_{k-1}), t(S_k)]$. The first term is then the monitoring cost up to the last experiment S_n. The second term, $(T - t(S_n))(d - d(S_1^n))$, is the remaining cost up to the time horizon T. Finally, the third term, cn, is the cost for performing n experiments.

Eq (2) can be rewritten as follows:

$$\Gamma(S) = -\sum_{k=1}^{n} (T - t(S_k))(d(S_1^k) - d(S_1^{k-1})) + Td + cn$$

If we now put $F(S) = d(S)$ and $g(\sigma) = T - t(\sigma)$, we have that $\Gamma(S) = -F_g(S) + Td + cn$, and the minimization of the cost for sequences of a given length is equivalent to the maximization of $F_g(S)$. The general minimization problem can then be solved as follows:

$$\min \Gamma(S) = \min_{n=0}^{|\Omega|} [Td + cn - F_g(S^{(n)})]$$

where $S^{(n)}$ is a maximum of $F_g(S)$ for the sequences of length n.

4. Preliminary results on sequence submodularity

In this section, we formally define the concepts of monotonicity and sequence submodularity of a function and describe the greedy algorithm for the maximization of sequence submodular functions originally proposed by Alaei and Malekian [5]. We then discuss some preliminary results concerning the functions of interest in this paper, i.e. those of the type in Eq. (1), and show that Alaei and Malekian's algorithm does not perform well on them (Example 2). In the next section, we propose a new greedy algorithm, which overcomes the limitations of the original one on such functions.

Consider the language $\mathbb{H}(\Omega)$ over a set Ω. If $S = (S_1, \dots, S_n)$ and $S' = (S'_1, \dots, S'_m)$ are two elements in $\mathbb{H}(\Omega)$, their concatenation is defined as:

$$S \perp S' = (S_1, \dots, S_n, S'_1, \dots, S'_m)$$

For the sake of notational simplicity, concatenations with sequences of length 1 (σ) will be denoted simply by $S \perp \sigma$ and $\sigma \perp S$, dropping the parentheses.

Definition 1. A function $J : \mathbb{H}(\Omega) \to \mathbb{R}$ is called **forward/backward monotonic** if, respectively,

$$J(S \perp \sigma) \geq J(S), \quad J(\sigma \perp S) \geq J(S) \quad \forall S \in \mathbb{H}(\Omega),\, \sigma \in \Omega$$

We use instead the term **anti-monotonic** if the inequalities are inverted.

Definition 2. A function $J : \mathbb{H}(\Omega) \to \mathbb{R}$ is called **forward/backward (sequence) submodular** if, for every $S, R \in \mathbb{H}(\Omega)$, $\sigma \in \Omega$, respectively,

$$J(S \perp R \perp \sigma) - J(S \perp R) \leq J(S \perp \sigma) - J(S)$$
$$J(\sigma \perp R \perp S) - J(R \perp S) \leq J(\sigma \perp S) - J(S)$$

For brevity, we drop *sequence* as we are only concerned about those functions in this paper. On the subset $\Omega^n \subseteq \mathbb{H}(\Omega)$ of sequences of length exactly n, there is a natural action of the permutation group $\mathcal{P}_n$ (i.e. the set of bijections from $\{1, \dots, n\}$ to itself):

$$S = (S_1, \dots, S_n),\ \theta \in \mathcal{P}_n \longrightarrow \theta S := (S_{\theta(1)}, \dots, S_{\theta(n)})$$

Definition 3. We define the concept of permutation invariance for sets and functions as follows:

- A subset $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ is said to be **permutation invariant** if for every $S \in \mathcal{I}$ and for every $\theta \in \mathcal{P}_{|S|}$, it holds $\theta S \in \mathcal{I}$.
- A function $J : \mathbb{H}(\Omega) \to \mathbb{R}$ is **permutation invariant** if, for every $R \in \mathbb{H}(\Omega)$ and for every $\theta \in \mathcal{P}_{|R|}$, it holds $J(\theta R) = J(R)$.

For permutation invariant functions, the backward and forward notions of monotonicity and submodularity always coincide and, in that case, we will refer to them as monotonic, anti-monotonic, and submodular functions.

Example 1. We now illustrate the concepts presented in this section in the context of our running example, presented in Section 3.1. In that case, we can confer the right interpretation to the function $F_g(S)$ only for sequences S such that t_{S_i} is monotonically increasing and, in consequence, $g(S_i) = T - t_{S_i}$ is monotonically decreasing.

For the sake of illustration, we consider now the extension of $F_g(S)$ to the entire language $\mathbb{H}(\Omega)$. More precisely, we consider the special case in which the experiments in $\Omega = \{\sigma_1 \dots, \sigma_r\}$ can be labeled so that the information they convey is monotonically increasing, i.e. $\mathcal{D}_{\sigma_1} \subseteq \mathcal{D}_{\sigma_2} \subseteq \cdots \subseteq \mathcal{D}_{\sigma_r}$. Given $\sigma \in \Omega$, we denote by $k(\sigma) = 1, \dots, r$ its index in this ordering, namely $k(\sigma)$ is such that $\sigma_{k(\sigma)} = \sigma$. We put $d_k = |\mathcal{D}_{\sigma_k}|$ for $k = 1, \dots, r$. Given $S \in \mathbb{H}(\Omega)$, we further put $k(S) = \max\{k(S_i) \mid i = 1, \dots, |S|\}$, and we notice that

$$d(S) = \left| \bigcup_{k=1}^{|S|} \mathcal{D}_{\sigma_k} \right| = |\mathcal{D}_{\sigma_{k(S)}}| = d_{k(S)} \tag{3}$$

If $S \in \mathbb{H}(\Omega)$ and $\sigma \in \Omega$, thanks to (3), we have that,

$$F_g(S \perp \sigma) - F_g(S) = g(\sigma)[d(S \perp \sigma) - d(S)] = \begin{cases} g(\sigma) d_{k(\sigma)} & \text{if } k(S) < k(\sigma) \\ 0 & \text{if } k(S) \geq k(\sigma) \end{cases} \tag{4}$$

Notice that $k(S)$ and $d(S)$ are both permutation invariant and monotonic. From expression (4), we obtain that F_g is forward monotonic and, using the monotonicity of $k(S)$, that is also forward submodular. Also note that

$$F_g(\sigma_i \perp \sigma_j) - F_g(\sigma_j) = g(\sigma_i)d_i + g(\sigma_j)d_j[\mathbb{1}_{i<j} - 1]$$

where $\mathbb{1}_{i<j}$ is 1 if $i < j$ and is 0 otherwise. The expression above shows that in general F_g is not backward monotonic: the right hand side can be negative if $i > j$ and g is such that $g(\sigma_i)d_i < g(\sigma_j)d_j$.

The following result shows how the function F_g behaves with respect to a transposition of two consecutive elements of a sequence S.

Lemma 1. *Given a permutation invariant and submodular function F and a function $g : \Omega \to \mathbb{R}^+$, consider the function F_g as defined in Eq. (1). Let $S \in \mathbb{H}(\Omega)$ and $k < |S|$ be such that $g(S_k) \le g(S_{k+1})$. Let $\tilde{S}$ be the sequence obtained from S by swapping S_k with S_{k+1}. Then,*

$$F_g(\tilde{S}) \ge F_g(S)$$

Proof. Thanks to the permutation invariance of the function F, we have that:

$$\begin{aligned}F_g(S) - F_g(\tilde{S}) &= g(S_k)[F(S|_1^k) - F(S|_1^{k-1})] + g(S_{k+1})[F(S|_1^{k+1}) - F(S|_1^k)] - g(S_{k+1})[F(S|_1^{k-1} \perp S_{k+1}) - F(S|_1^{k-1})] \\ &\quad - g(S_k)[F(S|_1^{k+1}) - F(S|_1^{k-1} \perp S_{k+1})] \\ &= [g(S_{k+1}) - g(S_k)][F(S|_1^{k+1}) - F(S|_1^k) - F(S|_1^{k-1} \perp S_{k+1}) + F(S|_1^{k-1})]\end{aligned}$$

We conclude the proof by observing that the last term is non-positive since $g(S_k) \le g(S_{k+1})$ and F is submodular. □

Remark 1. Notice that, under the stronger assumption that $g(S_k) = g(S_{k+1})$, Lemma 1 yields $F_g(\tilde{S}) = F_g(S)$.

Lemma 1 implies that if we want to maximize functions of the type in Eq. (1) over a set of sequences $\mathcal{I}$ that is permutation invariant, we can always restrict to those sequences $S \in \mathcal{I}$ for which $g(S_k) \ge g(S_{k+1})$ for every k. To formalize this fact, we consider a total ordering $\prec$ of the elements of Ω for which g is non-decreasing: $\sigma \prec \sigma'$ implies $g(\sigma) \le g(\sigma')$. We call this a *g-ordering* and note that the g-ordering is not unique when g is non-injective.

We now present a number of useful concepts related to any fixed total ordering $\prec$ on Ω. We use the notation $\sigma \succeq \sigma'$ to indicate that $\sigma' \prec \sigma$ or $\sigma' = \sigma$. A sequence $S \in \mathbb{H}(\Omega)$ is called *$\prec$-ordered* if $S_1 \succeq \cdots \succeq S_{|S|}$. A subset $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ is called *$\prec$-ordered* if each $S \in \mathcal{I}$ is $\prec$-ordered. Given any subset $\mathcal{I} \subseteq \mathbb{H}(\Omega)$, we indicate the $\prec$-ordered subset as follows:

$$\mathcal{I}(\prec) = \{S \in \mathcal{I} \mid S \text{ is } \prec\text{-ordered}\} \tag{5}$$

When $\mathcal{I} = \mathbb{H}(\Omega)$ or $\mathcal{I} = \mathbb{H}^d(\Omega)$, we will use the notation $\mathbb{H}(\Omega, \prec)$ and $\mathbb{H}^d(\Omega, \prec)$, respectively, for $\mathcal{I}(\prec)$. For the sake of simplicity, a subset $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ that is $\prec$-ordered with respect to a g-ordering $\prec$, where g is a function $g : \Omega \to \mathbb{R}^+$, will simply be called g-ordered.

The following is a direct consequence of Lemma 1.

Proposition 1. *Consider a permutation invariant and submodular function F and a function $g : \Omega \to \mathbb{R}^+$. Let $\prec$ be a g-ordering. Then, given any permutation invariant set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ and $T \in \mathbb{N}$, it holds that:*

$$\max_{\substack{S \in \mathcal{I} \\ |S| = T}} F_g(S) = \max_{\substack{S \in \mathcal{I}(\prec) \\ |S| = T}} F_g(S) \tag{6}$$

Remark 2. If we maximize $F_g(S)$ over $\mathbb{H}^d(\Omega)$ with sequences of maximal length $T = |\Omega|$, the only sequence S of length T belonging to $\mathbb{H}^d(\Omega, \prec)$ is a maximum. Because of Lemma 1 and Remark 1, all possible maxima can be obtained from S by arbitrarily permuting the elements of any subsequence $(S_h, S_{h+1}, \dots, S_k)$ for which $g(S_h) = g(S_k)$. In this case, the maximization problem boils down to a sort problem.

In our work, we are interested in investigating the maximization problem of the function $F_g(S)$ for the general case when sequences do not necessarily consist of distinct elements or have maximal length. In many applications, this is indeed the case.

Let us now fix a value $T \in \mathbb{N}$ and consider the problem of maximizing a function $J : \mathbb{H}(\Omega) \to \mathbb{R}$ on the sequences of fixed length T. A popular, simple, suboptimal algorithm for such maximization problems is the greedy algorithm by Alaei and Malekian [5], which generalizes the classical result in Nemhauser and Wolsey [9] to sequence functions. This algorithm produces recursively a sequence $S = (S_1, \dots, S_T)$ by adding new elements on the right side of the sequence so that, for every $k = 0, \dots, T - 1$,

$$J(S|_1^k \perp S_{k+1}) \ge J(S|_1^k \perp \sigma) \quad \forall \sigma \in \Omega \tag{7}$$

Alaei and Malekian [5] give a lower bound on the performance of the greedy algorithm in the presence of monotonicity and submodularity of J. In particular, fix a value $T \in \mathbb{N}$ and let S^T be the sequence generated by the greedy algorithm stopped at step T and $O^T \in \mathbb{H}(\Omega)$ any maximizing sequence of J restricted to sequences in $\mathbb{H}(\Omega)$ of length T. Now, assume that J is backward monotonic and forward submodular, then,

$$J(S^T) \geq \left(1 - \frac{1}{e}\right) J(O^T) \tag{8}$$

Considering now our function F_g, it is simple to see that it is forward monotonic and forward submodular, while, in general, it does not possess the other two complementary properties. Hence, the result in Eq. (8) cannot be applied to it. In Example 2, we show that the classical greedy algorithm can perform arbitrarily bad on such functions.

Example 2. Considering Example 1 again, we specify the values of the parameters as follows. We put $d_k = 2^{k-1}$ if $k = 1, \ldots, n-1$, while $d_n = 2^n$. We also assume that $g(\sigma_k) = 2^{n-k}$. Then, the value of the function over the single elements of Ω is given by the following expression:

$$F_g(\sigma_k) = g(\sigma_k)d(\sigma_k) = \begin{cases} 2^{n-1} & \text{if } k \leq n-1 \\ 2^n & \text{if } k = n \end{cases}$$

Given the assumptions made, this expression reaches its maximum for $k = n$. Consequently, the greedy solution S of length n will necessarily be such that $S_1 = \sigma_n$. This implies that

$$F_g(S) = g(\sigma_n)d(\sigma_n) = 2^n$$

as the remaining term will not give any further contribution. A direct check shows that, instead, the optimum among the sequences of length n is reached by $O = (\sigma_1, \ldots, \sigma_n)$. We can compute as follows:

$$F_g(O) = 2^{n-1}d_1 + \sum_{k=2}^{n} 2^{n-k}(d_k - d_{k-1}) = 2^{n-1} + \tfrac{(n-2)2^n}{4} + 2^n \tfrac{3}{4} = 2^n \tfrac{n+3}{4}$$

Therefore,

$$\frac{F_g(O)}{F_g(S)} = \frac{n+3}{4}$$

Hence, no bound of the form in Eq. (8) can possibly hold in this case.

5. A new greedy algorithm on fully extendable sets

We now present a new greedy algorithm for the maximization of sequence submodular functions that allows each new element of the sequence under construction to be placed in any position among the elements already in it. This approach differs from Alaei and Malekian's algorithm [5], which adds new elements only of the right-hand side of the sequence. We show that the new algorithm maintains the same theoretical guarantees as the original one and performs well on functions of the type in Eq. (1).

Compared to previous work, our optimization approach is more general as it allows problems to be defined not only over $\mathbb{H}(\Omega)$ (elements can be repeated) and $\mathbb{H}^d(\Omega)$ (elements are all distinct) but also over sets in which the number of repetitions of each element can be constrained to be below a certain value. To allow for such generality, in Section 5.1, we introduce the key concept of *fully extendable* set of sequences, and we generalize the notions of monotonicity and submodularity by adapting them to fully extendable sets. Our new greedy algorithm, described in Section 5.2, exploits such notions.

5.1. Monotonic and submodular functions on fully extendable sets

Definition 4. A subset $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ is called **fully extendable** if the following conditions are satisfied.

1. For every $\sigma \in \Omega$, $(\sigma) \in \mathcal{I}$;
2. If $R \in \mathcal{I}$ and $Q \preceq R$, then $Q \in \mathcal{I}$;
3. If $Q, R \in \mathcal{I}$, there exists $U \in \mathcal{I}$ such that $Q, R \preceq U$ and $|U| \leq |Q| + |R|$.

The third property says that, given two sequences $Q, R \in \mathcal{I}$, there must exist another sequence $U \in \mathcal{I}$ of which both are subsequences and whose length is at most the sum of the two lengths. If Q and R do not have any element in common, the only possibility is that U is obtained by intertwining Q and R and then $|U| = |Q| + |R|$.

We denote by $\mathcal{I}(Q, R)$ the subset of sequences U satisfying property 3 defined above. Given $Q \in \mathcal{I}$, we also denote

$$\mathcal{I}^{+}(Q) := \{U \in \mathcal{I} \mid Q \leq U,\ |U| = |Q| + 1\}$$

In other words, $\mathcal{I}^{+}(Q)$ consists of the sequences in $\mathcal{I}$ that are obtained from Q by adding one element. It follows from properties 1 and 3 and the considerations above that if there exist elements in Ω not appearing in Q, surely $\mathcal{I}^{+}(Q) \neq \varnothing$.

We now give some examples of fully extendable sets.

Example 3. $\mathbb{H}(\Omega)$ is a fully extendable set.

Example 4. Consider any total ordering $\prec$ in the set Ω. Then, the set of $\prec$-ordered sequences $\mathbb{H}(\Omega, \prec)$ (formally defined in Eq. (5)) is a fully extendable set.

Finally, we construct a family of fully extendable sets that subsumes the examples above and will be useful in the applications presented in Sections 7 and 8. Those are sets in which the number of repetitions of each element can be constrained to be below a certain value.

Example 5. We fix a total ordering $\prec$ on the set Ω. Given $S \in \mathbb{H}(\Omega)$, we denote by $n_\sigma(S)$ the number of times the element σ appears in the sequence S. The following two properties are a direct consequence of the definition of $\prec$-ordered sequences:

(i) Given non negative integer numbers n_σ for every $\sigma \in \Omega$, there exists exactly one $\prec$-ordered sequence S such that $n_\sigma(S) = n_\sigma$ for every $\sigma \in \Omega$.
(ii) Given two $\prec$-ordered sequences Q, R, we have that $Q \leq R$ if and only if $n_\sigma(Q) \leq n_\sigma(R)$ for every $\sigma \in \Omega$.

For every $\sigma \in \Omega$, fix a number $n_\sigma \in \{1, 2, \dots\} \cup \{+\infty\}$ and consider the set of sequences

$$\mathcal{I} = \{S \in \mathbb{H}(\Omega, \prec) \mid n_\sigma(S) \leq n_\sigma\ \forall \sigma \in \Omega\} \tag{9}$$

Note that $\mathbb{H}(\Omega, \prec)$ and $\mathbb{H}^d(\Omega, \prec)$ are special cases of $\mathcal{I}$, obtained when, respectively, $n_\sigma = +\infty$ and $n_\sigma = 1$ for every $\sigma \in \Omega$.

We have the following result:

Proposition 2. *The set of sequences $\mathcal{I}$ defined in Eq. (9) is fully extendable.*

Proof. All singleton sequences $S = (\sigma)$ are $\prec$-ordered and respect the repetition constraint (since $n_\sigma \geq 1$). Therefore, they are in $\mathcal{I}$ and property 1 in Definition 4 holds. Property 2 also holds because any subsequence Q of a sequence in $R \in \mathcal{I}$ is necessarily g-ordered and satisfies, thanks to property (ii) above, the constraints $n_\sigma(Q) \leq n_\sigma(R) \leq n_\sigma$ for every $\sigma \in \Omega$. To check property 3, consider now two sequences $Q, R \in \mathcal{I}$ and put, for every $\sigma \in \Omega$,

$$\bar{n}_\sigma = \max\{n_\sigma(Q), n_\sigma(R)\}$$

Let U be the only $\prec$-ordered sequence such that $n_\sigma(U) = \bar{n}_\sigma$ (see property (i)). Since by construction $\bar{n}_\sigma \leq n_\sigma$ for all $\sigma \in \Omega$, we have that $U \in \mathcal{I}$. Notice now that both Q and R are subsequences of U because of property (ii). Finally,

$$|U| = \sum_{\sigma \in \Omega} n_\sigma(U) \leq \sum_{\sigma \in \Omega} [n_\sigma(Q) + n_\sigma(R)] = |Q| + |R|$$

That completes the proof. $\square$

It follows from Proposition 2 that $\mathbb{H}(\Omega, \prec)$ and $\mathbb{H}^d(\Omega, \prec)$ are both fully extendable.

Example 6. If $|\Omega| > 1$, the set $\mathbb{H}^d(\Omega)$ is not fully extendable. Indeed, let $\sigma_1, \sigma_2 \in \Omega$ be two distinct elements and consider the sequences $R = (\sigma_1, \sigma_2)$ and $S = (\sigma_2, \sigma_1)$. Any sequence U of which both R and S are subsequences must contains either two copies of σ_1 or two copies of σ_2 and thus cannot belong to $\mathbb{H}^d(\Omega)$. This reveals that the condition 3 in Definition 4 does not hold true.

Let us now see how the notions of monotonicity and submodularity change when adapted to fully extendable sets. The novelty is that, in the definitions below, the new element σ that is added to the sequences can appear not only at the beginning and at the end of them, as with the standard notions of monotonicity and submodularity, but also in between the sequences.

Definition 5. Given a fully extendable set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$, a function $F : \mathcal{I} \to \mathbb{R}$ is called $\mathcal{I}$-**monotonic** if for every $Q, R \in \mathcal{I}$ and $\sigma \in \Omega$ such that $Q \perp \sigma \perp R \in \mathcal{I}$, it holds:

$$F(Q \perp \sigma \perp R) \geq F(Q \perp R) \tag{10}$$

Definition 6. Given a fully extendable set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$, a function $F : \mathcal{I} \to \mathbb{R}$ is called $\mathcal{I}$-**submodular** if for every $Q, R, S \in \mathcal{I}$ and $\sigma_1, \sigma_2 \in \Omega$ such that $Q \perp \sigma_1 \perp R \perp \sigma_2 \perp S \in \mathcal{I}$, it holds:

$$F(Q \perp \sigma_1 \perp R \perp \sigma_2 \perp S) - F(Q \perp \sigma_1 \perp R \perp S) \leq F(Q \perp R \perp \sigma_2 \perp S) - F(Q \perp R \perp S) \tag{11}$$

Relation (11) is a way to express the diminishing return property that characterizes the definition of submodularity. Adding an extra element σ_2 to a sequence produces a smaller impact on the growth of the function F calculated over the sequence as more elements are added to its prefix, as long as the constructed sequences remain elements of the set $\mathcal{I}$.

Remark 3. If $\mathcal{I} = \mathbb{H}(\Omega)$, $\mathcal{I}$-monotonic functions are backward and forward monotonic and $\mathcal{I}$-submodular functions are forward submodular.

Remark 4. If $F : \mathbb{H}(\Omega) \to \mathbb{R}$ is permutation invariant, monotonic and submodular, then, for every fully extendable set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$, the restriction of F to $\mathcal{I}$ is $\mathcal{I}$-monotonic and $\mathcal{I}$-submodular.

5.2. A new greedy algorithm

We now introduce a generalized greedy algorithm and show that, for functions that are $\mathcal{I}$-monotonic and $\mathcal{I}$-submodular, this new algorithm ensures the same performance as in expression (8). In the next section, we will then show how to apply this result to our problems.

Take a function $J : \mathbb{H}(\Omega) \to \mathbb{R}$ and a fully extendable set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$. We fix a value $T \in \mathbb{N}$: the goal is to maximize J over the subset of $\mathcal{I}$ of the sequences of length T. Put O^T to be any such maximizing sequence for J.

We now consider a variation of the greedy algorithm to approximately solve this maximization problem. The algorithm produces recursively an ordered sequence $S^T = (S_1^T, \ldots, S_T^T) \in \mathcal{I}$ in the following way:

- $S^1 = (S_1^1)$ where $S_1^1 \in \underset{\sigma \in \Omega}{\operatorname{argmax}}\, J(\sigma)$;
- Given $S^k = (S_1^k, \ldots, S_k^k) \in \mathcal{I}$, we define

$$S^{k+1} \in \underset{U \in \mathcal{I}^+(S^k)}{\operatorname{argmax}}\, J(U) \tag{12}$$

In other words, instead of simply augmenting the sequence on the right hand side as the traditional greedy algorithm does, we allow each new element to be placed in any position among the elements of the previous sequence. We note that, when the argmax in expression (12) is not a singleton, the choice of S^{k+1} can be any arbitrary element of it. The performance of the algorithm will not be affected by this choice.

The following result shows that this new algorithm satisfies the same performance bound than the classical one (see expression (8)).

Theorem 1. *Consider a function $J : \mathbb{H}(\Omega) \to \mathbb{R}$ and a fully extendable set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ and assume that J is $\mathcal{I}$-monotonic and $\mathcal{I}$-submodular. Let O^T be a maximizing sequence for J among the sequences in $\mathcal{I}$ of length T and let S^T be the result of the previous algorithm. Then,*

$$J(S^T) \geq \left(1 - \frac{1}{e}\right) J(O^T)$$

Proof. For simplicity of notation, in the proof, we put $O = O^T$. Fix $k < T$ and consider

$$\Lambda = (\lambda_1, \ldots, \lambda_n) \in \underset{U \in \mathcal{I}(S^k, O)}{\operatorname{argmax}}\, J(U)$$

We consider a partition of the indices

$$\{1, 2, \ldots, n\} = \{i_1, i_2, \ldots, i_k\} \cup \{j_1, j_2, \ldots, j_m\}$$

where $i_1 < i_2 < \cdots < i_k$ are such that $S_l^k = \lambda_{i_l}$ for $l = 1, \ldots, k$ and $j_1 < j_2 < \cdots < j_m$ with $m = n - k$ are the remaining indices.

We now consider, for $0 \le t \le m$, the sequence $\Lambda^{(t)}$ obtained from Λ by removing the elements $\lambda_{j_m}, \lambda_{j_{m-1}}, \dots, \lambda_{j_{t+1}}$. Note that, by property 2 of fully extendable sets, $\Lambda^{(t)} \in \mathcal{I}$ for every t and that $\Lambda^{(m)} = \Lambda$ and $\Lambda^{(0)} = S^k$. We can write

$$J(\Lambda) - J(S^k) = \sum_{t=1}^{m} \left[J(\Lambda^{(t)}) - J(\Lambda^{(t-1)}) \right] \tag{13}$$

Using the property of $\mathcal{I}$-submodularity and removing the elements $\lambda_{j_{t-1}}, \dots, \lambda_{j_0}$ from $\Lambda^{(t-1)}$ and $\Lambda^{(t)}$, we obtain that

$$J(\Lambda^{(t)}) - J(\Lambda^{(t-1)}) \le J(U^{(t)}) - J(S^k) \tag{14}$$

for some $U^{(t)} \in \mathcal{I}^+(S^k)$ (a sequence obtained from S^k adding in some position the element λ_{j_t}). Given the definition of the extended greedy solution S^k, it follows that $J(U^{(t)}) \le J(S^{k+1})$. This fact together with expressions (13) and (14) yields:

$$J(\Lambda) - J(S^k) \le T \left[J(S^{k+1}) - J(S^k) \right] \tag{15}$$

The assumption of $\mathcal{I}$-monotonicity and the choice of Λ to maximize J on $\mathcal{I}(S^k, O)$ ensure that $J(\Lambda) \ge J(O)$. Using this fact inside expression (15) gives:

$$J(S^{k+1}) \ge \frac{1}{T} J(O) + \left(1 - \frac{1}{T}\right) J(S^k)$$

for every $k = 0, \dots, T-1$. Applying recursively this relation, we obtain that

$$J(S^T) \ge \quad \frac{1}{T} \sum_{i=0}^{T-1} \left(1 - \frac{1}{T}\right)^i J(O) = \left[1 - \left(1 - \frac{1}{T}\right)^T\right] J(O) \ge \left(1 - \frac{1}{e}\right) J(O) \quad \square$$

6. A detailed analysis of the function F_g

We now go back to our original optimization problem on functions of the type of Eq. (1) and study the conditions under which we can apply the theory laid out in the previous section to it. Our aim is to show that, under suitable assumptions, the function F_g satisfies the conditions of Theorem 1 on fully extendable g-ordered sets, which – thanks to Proposition 1 – will allow us to obtain a solution with bounded suboptimality to our maximization problem.

We start by introducing some additional notation that will become handy in the proof of the main result of this section.

$$\begin{aligned} \Delta F(Q, \sigma, R) &= F(Q \perp \sigma \perp R) - F(Q \perp R) \\ \Delta^2 F(Q, \sigma_1, R, \sigma_2, S) &= \Delta F(Q \perp \sigma_1 \perp R, \sigma_2, S) - \Delta F(Q \perp R, \sigma_2, S) \\ &= F(Q \perp \sigma_1 \perp R \perp \sigma_2 \perp S) - F(Q \perp \sigma_1 \perp R \perp S) - F(Q \perp R \perp \sigma_2 \perp S) + F(Q \perp R \perp S) \end{aligned} \tag{16}$$

The first expression, $\Delta F(Q, \sigma, R)$, is the 'first' variation of F obtained starting from the sequence $Q \perp R$ and adding one more element σ between the subsequences Q and R. The second expression, $\Delta^2 F(Q, \sigma_1, R, \sigma_2, S)$, is instead a 'second' variation, namely a variation of the first variation, which goes from $Q \perp R$ to $Q \perp \sigma_1 \perp R$, relative to an added element σ_2. They play an analogous role to, respectively, the discrete derivative and the discrete second derivative of a function; this analogy will become apparent below when we discuss an example of a submodular functions obtained through convex functions.

Note that the $\mathcal{I}$-monotonicity of F is equivalent to the requirement that $\Delta F(Q, \sigma, R) \ge 0$ for every choice of Q, R and σ such that $Q \perp \sigma \perp R \in \mathcal{I}$, while the $\mathcal{I}$-submodularity of F is equivalent to the requirement that $\Delta^2 F(Q, \sigma_1, R \sigma_2, S) \le 0$ under the assumption that $Q \perp \sigma_1 \perp R \perp \sigma_2 \perp S \in \mathcal{I}$.

We are now ready to state and prove the main result of this section. It asserts that the $\mathcal{I}$-monotonicity and $\mathcal{I}$-submodularity of a function F are transferred to a function F_g if the fully extendable set $\mathcal{I}$ is g-ordered, namely is $\prec$-ordered with respect to any g-ordering $\prec$.

Theorem 2. *Let $g : \Omega \to \mathbb{R}^+$ and let $\mathcal{I} \subseteq \mathbb{H}(\Omega)$ be a g-ordered fully extendable set. Given any $F : \mathcal{I} \to \mathbb{R}$, it holds that*

1. *If F is $\mathcal{I}$-monotonic, then F_g is $\mathcal{I}$-monotonic;*
2. *If F is $\mathcal{I}$-submodular, then F_g is $\mathcal{I}$-submodular.*

Proof. Assume that F is $\mathcal{I}$-monotonic. Fix $Q, R \in \mathcal{I}$ and $\sigma \in \Omega$ such that $Q \perp \sigma \perp R \in \mathcal{I}$. Put $n = |R|$. From the definition of F_g, we obtain that

$$\begin{aligned}
\Delta F_g(Q,\sigma,R) &= F_g(Q) + g(\sigma)\Delta F(Q,\sigma,\emptyset) \\
&\quad + \sum_{k=1}^{n} g(R_k)(F(Q\perp\sigma\perp R|_1^k) - F(Q\perp\sigma\perp R|_1^{k-1}) - F_g(Q) \\
&\quad - \sum_{k=1}^{n} g(R_k)(F(Q\perp R|_1^k) - F(Q\perp R|_1^{k-1})) \\
&= g(\sigma)\Delta F(Q,\sigma,\emptyset) + \sum_{k=1}^{n} g(R_k)\Delta F(Q,\sigma,R|_1^k) - \sum_{k=0}^{n-1} g(R_{k+1})\Delta F(Q,\sigma,R|_1^k) \\
&= [g(\sigma)-g(R_1)]\Delta F(Q,\sigma,\emptyset) + g(R_n)\Delta F(Q,\sigma,R) + \sum_{k=1}^{n-1}[g(R_k)-g(R_{k+1})]\Delta F(Q,\sigma,R|_1^k)
\end{aligned} \tag{17}$$

By the assumption on $\mathcal{I}$, we have that

$$[g(\sigma)-g(R_1)] \geq 0, \;\; g(R_n) \geq 0, \;\; [g(R_k)-g(R_{k+1})] \geq 0 \;\forall k \tag{18}$$

On the other hand, $\mathcal{I}$-submonotonicity of F yields that all the first variation terms appearing in the expression above are non-negative. It thus follow that $\Delta F_g(Q,\sigma,R) \geq 0$. This proves that F_g is $\mathcal{I}$-monotonic.

Assume that F is $\mathcal{I}$-submodular. Consider now $Q, R, S \in \mathcal{I}$ and $\sigma_1, \sigma_2 \in \Omega$ such that $Q\perp\sigma_1 R\perp\sigma_2\perp S \in \mathcal{I}$. Put $m = |S|$. Then, from the expressions (16) and (17), we obtain that

$$\begin{aligned}
\Delta^2 F_g(Q,\sigma_1,R,\sigma_2,S) &= \Delta F_g(Q\perp\sigma_1\perp R,\sigma_2,S) - \Delta F_g(Q\perp R,\sigma_2,S) \\
&= [g(\sigma_2)-g(S_1)]\cdot[\Delta F(Q\perp\sigma_1\perp R,\sigma_2,\emptyset) - \Delta F(Q\perp R,\sigma_2,\emptyset)] \\
&\quad + g(S_n)[\Delta F(Q\perp\sigma_1\perp R,\sigma_2,S) - \Delta F(Q\perp R,\sigma_2,S)] \\
&\quad + \sum_{k=1}^{m-1}[g(S_k)-g(S_{k+1})][\Delta F(Q\perp\sigma_1\perp R,\sigma_2,S|_1^k) - \Delta F(Q\perp R,\sigma_2,S|_1^k)] \\
&= [g(\sigma_2)-g(S_1)]\Delta^2 F(Q,\sigma_1,R,\sigma_2) + g(S_m)\Delta^2 F(Q,\sigma_1,R,\sigma_2,S) \\
&\quad + \sum_{k=1}^{m-1}[g(S_k)-g(S_{k+1})]\Delta^2 F(Q,\sigma_1,R,\sigma_2,S|_1^k)
\end{aligned} \tag{19}$$

From the inequalities in (18) and the fact that all second variation terms $\Delta^2 F$ appearing above are, as F is $\mathcal{I}$-submodular, non-positive, it follows that $\Delta^2 F_g(Q,\sigma_1,R,\sigma_2,S) \leq 0$ □

A direct consequence of Theorem 2 is the following corollary, which is crucial to analyze all our applicative examples as we will see in the next section.

Corollary 1. *Consider a permutation invariant, monotonic, and submodular function $F : \mathbb{H}(\Omega) \to \mathbb{R}$, a function $g : \Omega \to \mathbb{R}^+$, and a g-ordered fully extendable set $\mathcal{I} \subseteq \mathbb{H}(\Omega)$. Then, F_g is $\mathcal{I}$-monotonic and $\mathcal{I}$-submodular.*

Proof. It is a direct consequence of Theorem 2 and of Remark 4. □

7. Detection and monitoring problems

In this section, we present a few applicative case studies regarding detection and monitoring and show how they can be framed within the theory developed so far. The case studies revolve around the problem of choosing an optimal sequence of experiments to optimize a quantity of interest. We start with a broad formulation of this problem, focusing on the common characteristics of the case studies. We continue by exploring detection problems, which involve minimizing the detection time of a given event, and we then tackle a particular instance of them, i.e. search-and-tracking (S&T), which is an important applicative domain [2,21,22]. We conclude this section by considering environmental monitoring problems in which the goal is to minimize the cost of the monitoring infrastructure. For each example, we show that its formal representation leads to the maximization of a function F_g on a fully extendable set $\mathcal{I}$ for which the properties of Corollary 1 are satisfied and, in consequence, our generalized greedy algorithm can be profitably applied.

Consider a set of experiments Ω. Each experiment $\sigma \in \Omega$ is associated with a random variable X^σ on a finite space $\mathcal{A}$, which expresses the outcome of the experiment σ, and a time stamp $t(\sigma)$, which indicates the time at which that experiment can take place (e.g. because the device to perform the experiment is only available at that time).

We are interested in scenarios where multiple experiments take place, statistically described as follows. For each sequence $S = (S_1, \dots, S_n) \in \mathbb{H}(\Omega)$, we consider a sequence of random variables $X^S = (X_1, \dots, X_n)$ distributed according to a law $P_S : \mathcal{A}^n \to [0, 1]$. $P_S(\omega_1, \dots \omega_n)$ is the probability that the set of experiments S has a global result $(\omega_1, \dots \omega_n) \in \mathcal{A}^n$. Note that the same experiment σ can be repeated within the sequence S.

We make two assumptions concerning the experiments: the specific order of disclosing the outcome of the various experiments does not play any role; and ignoring the result of a subset of the experiments is equivalent to not having performed them at all. Formally:

1. *Permutation covariance:* For every $S = (S_1, \dots, S_n) \in \mathbb{H}(\Omega)$ of length n, $\omega = (\omega_1, \dots, \omega_n) \in \mathcal{A}^n$, and permutation $\theta \in \mathcal{P}_n$, it holds that

   $$P_{\theta S}(\theta\omega) = P_S(\omega)$$

   where $\theta S = (S_{\theta(1)}, \dots, S_{\theta(n)})$ and $\theta\omega = (\omega_{\theta(1)}, \dots, \omega_{\theta(n)})$.
2. *Coherence:* Given $R, S \in \mathbb{H}(\Omega)$ with $R \le S$, the law of X^R is the same as the law of the subsequence of the random variables $(X^S)^{|R}$ that are obtained from X^S considering only those components corresponding to the positions of R inside S. In particular, the i-th component of X^S has the same law than X^{S_i} for every $i = 1, \dots, |S|$.

Against the backdrop of the framework outlined above, we now analyze the different problems, which differ in how they exploit the experiments and in the quantity they want to optimize.

7.1. Detection problems

Given the space of the experiment outcomes, let us consider a specific event $E \subseteq \mathcal{A}$, which we call the *success event*, that represents a desired outcome of one of the experiments (e.g. a positive detection event). The problem is to determine, among all the experiment sequences of a given length, the one that minimizes the expected time to obtain the success event. We now formalize this problem. We fix a $(-t)$-ordering $\prec$ on Ω (e.g. $\sigma_1 \prec \sigma_2$ implies $t(\sigma_1) > t(\sigma_2)$) and consider the fully extendable set of $\prec$-ordered sequences $\mathbb{H}(\Omega, \prec)$ (as defined in Eq. (5)). For every $S \in \mathbb{H}(\Omega, \prec)$ with $|S| = n$, we define the *first detection time* as the random variable $\tau_S : \mathcal{A}^n \to \mathbb{R}$ such that

$$\tau_S(\omega) = \begin{cases} t(S_k) & \text{if } \omega_k \in E,\ \omega_j \notin E \text{ for } j < k \\ T & \text{if } \omega_j \notin E \text{ for every } j \end{cases}$$

where $T \ge \max_{\sigma\in\Omega} t(\sigma)$ is a constant playing the role, as we will see below, of a penalty for the fact that detection has not succeeded within S. The quantity that we aim to minimize is $\mathbb{E}[\tau_S]$, i.e. the expected value of τ_S on the set $\mathbb{H}(\Omega, \prec)$. We now show how such a goal involves the maximization of a function of the type in Eq. (1).

We put $E_n = \{\omega \in \mathcal{A}^n \mid \exists i \text{ s.t. } \omega_i \in E\}$. Given a sequence $S \in \mathbb{H}(\Omega)$ of length $|S| = n$, we define

$$F(S) = P_S(E_n)$$

that is the probability that the success event E has happened within the sequence of experiments S. The expected success time can then be computed as follows:

$$\mathbb{E}[\tau_S] = \sum_{k=1}^{n} t(S_k)(F(S|_1^k) - F(S|_1^{k-1})) + T(1 - F(S)) = T + \sum_{k=1}^{n}(t(S_k) - T)(F(S|_1^k) - F(S|_1^{k-1})) \tag{20}$$

The minimization of the expected success time is then equivalent to the maximization of the function

$$\sum_{k=1}^{n}(T - t(S_k))(F(S|_1^k) - F(S|_1^{k-1})) \tag{21}$$

which is of the form in Eq. (1) with $g(\sigma) = T - t(\sigma)$. We observe that the optimization problem in this case is already defined over the fully extendable set $\mathcal{I} = \mathbb{H}(\Omega, \prec)$. We now show that the properties needed to apply Corollary 1 hold:

- Permutation invariance of F follows from the permutation covariance property of the family of probabilities P_S and the fact that the success events E_n are permutation invariant.
- Monotonicity of F follows from the following computation:

  $$\Delta F(S, \sigma) = P_{S\perp\sigma}(E_{n+1}) - P_S(E_n) = P_{S\perp\sigma}(E_{n+1}) - P_{S\perp\sigma}(E_n \times \mathcal{A}) = P_{S\perp\sigma}(E^c \times \dots \times E^c \times E) \ge 0 \tag{22}$$

  where we denote $E^c = \mathcal{A} \setminus E$.

- Submodularity of F follows from the following computation:

$$\begin{aligned}\Delta^2 F(S,\sigma_1,\sigma_2) &= P_{S\perp\sigma_1\perp\sigma_2}(E^c\times\cdots\times E^c\times E) - P_{S\perp\sigma_2}(E^c\times\cdots\times E^c\times E)\\ &= P_{S\perp\sigma_1\perp\sigma_2}(E^c\times\cdots\times E^c\times E) - P_{S\perp\sigma_1\perp\sigma_2}(E^c\times\cdots\times E^c\times\mathcal{A}\times E)\\ &= -P_{S\perp\sigma_1\perp\sigma_2}(E^c\times\cdots\times E^c\times E\times E)\le 0\end{aligned}\tag{23}$$

  Notice how, in the second equalities of (22) and (23), we use the coherence property.
- By definition, the set $\mathbb{H}(\Omega,\prec)$ is g-ordered with $g(\sigma)=T-t(\sigma)$.

An important extension of this example is the case when there are multiple detection events of interest $E^1,\dots,E^s\subseteq\mathcal{A}$, possibly correlated among themselves. We indicate with τ_S^k the first detection time of the event E^k. In certain applications, it is natural to consider the global average completion time $\mathbb{E}[\max_k\tau_S^k]$ as the function to minimize; this happens when a partial detection of only a subset of the events has no value. This new function has no submodularity properties. However, in those applications where each detection event has an intrinsic value, it is useful to minimize the function $\mathbb{E}[\sum_k\tau_S^k]$. This function fits in our framework. Indeed, if we define $F^k(S)=P_S(E_n^k)$ and

$$F(S)=\sum_{k=1}^{s}F^k(S)$$

we obtain, computing as in Eq. (20), that the minimization of $\mathbb{E}[\sum_k\tau_S^k]$ is equivalent to the maximization of the function in expression (21).

We now present a specific instance of the general detection problem presented above, which corresponds to an important practical application.

7.2. Search-and-tracking

As a specific applicative example of the detection problem presented in Section 7.1, we consider the search-and-tracking (S&T), which is the problem of locating a moving target in a given area and following it to destination. Following a state-of-the-art S&T application in this area [2,22], we assume that the target travels across a large geographical area by following a road network (set of paths Γ), and the observer is a UAV with imperfect sensors. When the UAV loses track of the target, a set of candidate flight search patterns Ω is selected via a Monte Carlo simulation to direct the search towards the areas in which it is more probable to rediscover the target (see [23] for more detail on the Monte Carlo simulation). The UAV, however, has not enough resources to execute all candidates and a subset of patterns needs to be selected and arranged in a feasible sequence for execution.

Each pattern $\sigma\in\Omega$ provides visibility over a family of paths $\Gamma_\sigma\subseteq\Gamma$, i.e. if the target follows a route in Γ_σ, the UAV may be able to detect it while performing pattern σ. Each pattern $\sigma\in\Omega$ is also associated with: (i) a time stamp $t(\sigma)$, indicating the mid-point of a time window during which the target might plausibly be in the area covered by σ; and (ii) a detection probability ϕ_σ with the following meaning: assuming that the target has taken a route in Γ_σ, if the UAV performs the pattern σ at time $t(\sigma)$, detection will be positive with probability ϕ_σ. In all other cases, detection will be negative. We assume an a-priori uniform probability distribution on the routes in Γ, as well as independence of the outcomes of the search experiments conditioned to the fact that the target has chosen a specified route.

In Bernardini et al. [22], the authors study the problem of selecting the sequence $S\in\mathbb{H}(\Omega)$ of length n that minimizes the expected detection time (defined using the reference time stamps $t(\sigma)$ for the various elements of the sequence). They allow repetitions of the same search patterns as they correspond to the UAV repeating the search in the same area. Below, we show how this problem fits in the general detection framework presented in this section, while, in Section 9, we provide numerical simulations that highlight how the generalized greedy approach benefits the solution of the problem.

To every search pattern $\sigma\in\Omega$, we associate a binary random variable X^σ on $\{0,1\}$ that describes the outcome of the search performed at σ, where 1 expresses the positive detection event (target is found). For each sequence $S\in\mathbb{H}(\Omega)$, the probability distribution $P_S:\{0,1\}^n\to[0,1]$ of the outcomes of the corresponding search experiments is constructed on the basis of the topology of the road network and the detection probability of each pattern. More precisely, given $S\in\mathbb{H}(\Omega)$ of length n, we consider a joint probability distribution $\tilde{P}_S$ on $\{0,1\}^n\times\Gamma$ where $\tilde{P}_S(\omega_1,\dots,\omega_n,\gamma)$ denotes the probability that the target has taken the road γ and the outcome of performing the n search patterns $S_1,S_2,\dots,S_n$ have given results, respectively, $\omega_1,\omega_2,\dots,\omega_n$. The probability $\tilde{P}_S$ is univocally described by assuming that its marginal on Γ is the uniform distribution and that

$$\tilde{P}_S(\omega_1,\dots,\omega_n\,|\,\gamma)=\prod_{\substack{i\,:\,\gamma\notin\Gamma_{S_i}\\ \omega_i=1}}(1-\omega_i)\prod_{\substack{i\,:\,\gamma\in\Gamma_{S_i}\\ \omega_i=1}}\phi_{S_i}\prod_{\substack{i\,:\,\gamma\in\Gamma_{S_i}\\ \omega_i=0}}(1-\phi_{S_i})\tag{24}$$

The above equation summarizes our assumptions: given that the target has taken route γ, the outcome of the various experiments in S will give an independent outcome. The patterns S_i not compatible with γ will deterministically give a null result, while the others will be positive with probability ϕ_{S_i} and null with probability $1-\phi_{S_i}$. The probability distribution P_S of $X^S=(X^{S_1},\dots,X^{S_n})$ is obtained from $\tilde{P}_S$ by averaging over $\gamma\in\Gamma$. The properties of permutation covariance and coherence for P_S follow directly from Eq. (24). The detection event is, in this case, $E_n=\{(0,\dots,0,1)\}$. In Bernardini et al. [22], the authors propose an explicit iterative expression for the function $F(S)=P_S(E_n)$, which is useful for computation purposes.

As in the general detection problem, here the optimization problem is naturally defined over the set $\mathbb{H}(\Omega,\prec)$. However, other choices can be of interest: for instance, we can use $\mathbb{H}^d(\Omega,\prec)$, if we want to enforce all search patterns to be distinct or, more generally, we can use the fully extendable sets defined in Eq. (9) to impose specific restrictions on the number of repetitions. Such type of restrictions may reflect the time needed for the UAV to perform a search pattern or the number of UAV's that are simultaneously available for the search.

7.3. Monitoring problems

In this section, we present monitoring applications by going back to Example 1 (Section 3.1) and framing it within a more general and richer context.

Let us consider a system that monitors an environmental phenomenon $\mathcal{Z}$, which is influenced by a set of uncertain factors. As seen in Example 1, the phenomenon could be the level of a river subject to flooding, which is determined by the intensity of the precipitations (among other factors). The phenomenon $\mathcal{Z}$ manifests itself by a set $\mathcal{D}$ of events. In Example 1, such events are given by whether the river will overflow at some established checkpoints over its course.

We now consider a family Ω of Bernoulli experiments, formally modeled as in Section 7.2, that allow the system to observe the events in $\mathcal{D}$. Each experiment $\sigma\in\Omega$ is associated with a random variable X^σ on $\{0,1\}$ and a time stamp $t(\sigma)$. The output of a sequence of experiments $S\in\mathbb{H}(\Omega)$ performed at the prescribed times, $X^S=(X^{S_1},\dots,X^{S_l})$, is governed by the distribution laws $P_S(x)$ as $x\in\{0,1\}^l$. Such distributions are calculated on the basis of historical data on the phenomenon $\mathcal{Z}$ or other information, and we assume that they satisfy the same set of assumptions concerning permutation covariance and coherence that hold true in the case of the detection problems.

Each $\sigma\in\Omega$ is also associated with a subset $\mathcal{D}_\sigma\subseteq\mathcal{D}$ with the following meaning. If the monitoring system performs the experiment σ at its associated time and observes a negative outcome, $X^\sigma=0$, the system is guaranteed that none of the events in $\mathcal{D}_\sigma$ will take place, namely $\mathcal{Z}\neq E$ for any $E\subseteq\mathcal{D}$ such that $E\cap\mathcal{D}_\sigma\neq\emptyset$. In our example, the system can observe the level of the river at some of the checkpoints and, for any observation of the water being below a safety threshold, it can exclude that a flooding will happen at that point.

We model the cost of handling the phenomenon $\mathcal{Z}$ without the aid of the additional experiments in Ω through an additive function $C:2^{\mathcal{D}}\to\mathbb{R}^+$. C is determined by assigning nonnegative numbers c_i to each $i\in\mathcal{D}$ and, then, defining, given $E\subseteq\mathcal{D}$, $C(E)=\sum_{i\in E}c_i$. This is the global cost that the system incurs to handle all the possible events in E. In Example 1, this cost arises from the use of moveable floodgates, which are lifted as soon as the precipitations increase above a certain threshold. On the other hand, each experiment in Ω is associated to a cost c. In Example 1, this cost emerges from the use of a drone to monitor some of the checkpoints and evaluate whether lifting the corresponding floodgates is needed.

Given a sequence S of length l and a binary vector $x\in\{0,1\}^l$, we put

$$\mathcal{D}_{S,x}=\bigcup_{j:x_j=0}\mathcal{D}_{S_j} \tag{25}$$

Let us now assume that we keep monitoring all the events in $\mathcal{D}$ for which the experiments have not given a negative outcome. Over a time horizon $[0,T]$, the total cost that the system incurs upon performing the sequence of experiments $S=(S_1,\dots,S_n)$ with output $X=X^S$ is given by the following expression:

$$\begin{aligned}\Gamma(S,X)&=\sum_{k=1}^{n}(t(S_k)-t(S_{k-1}))C\left(\mathcal{D}\setminus\mathcal{D}_{S_1^{k-1},X_1^{k-1}}\right)+(T-t(S_n))C\left(\mathcal{D}\setminus\mathcal{D}_{S_1^n,X_1^n}\right)+nc\\&=-\sum_{k=1}^{n}(T-t_{S_k})\left(C\left(\mathcal{D}_{S_1^k,X_1^k}\right)-C\left(\mathcal{D}_{S_1^{k-1},X_1^{k-1}}\right)\right)+TC(\mathcal{D})+cn\end{aligned}$$

A natural optimal problem is the minimization of the average global cost $\Gamma(S)=\mathbb{E}[\Gamma(S,X)]$. If we put $F(S)=\mathbb{E}[C(\mathcal{D}_{S,X})]$ and $g(\sigma)=T-t(\sigma)$, we have that

$$\Gamma(S)=-F_g(S)+TC(\mathcal{D})+cn$$

Minimizing $\Gamma(S)$ for sequences of experiments of a given length is equivalent to maximizing $F_g(S)$. Indicated with $S^{(n)}$ a maximum of $F_g(S)$ for sequences of length n, we are finally left with the following problem:

$$\min\Gamma(S)=\min_{n=0}^{|\Omega|}[Td+cn-F_g(S^{(n)})] \tag{26}$$

Note that we can reconstruct the special case considered in Section 3.1 by choosing C as the cardinality function and assuming that deterministically $X_i = 0$ for all i.

Below, we show that the assumptions in Corollary 1 are satisfied and hence our theory can be applied to find approximate maxima for $F_g(S)$ for sequences of experiments of any given length. It is worth noting that the inherent recursive structure of the solution obtained through the greedy algorithm yields a simple iterative solution of the final minimization problem (26).

We now discuss the applicability of Corollary 1. The set of sequences on which this maximization takes place is here the set $\mathbb{H}^d(\Omega, \prec)$, where $\prec$ is any $(-t)$-ordering on the set Ω. Concerning the properties of F, we reason as follows.

- Permutation invariance is obtained via the following computation. For a fixed $S \in \mathbb{H}(\Omega)$ of length n and a permutation $\theta : \{1, \ldots, n\} \to \{1, \ldots, n\}$, we have that

$$F(S^\theta) = \sum_{x\in\{0,1\}^n} P_{S^\theta}(x) C(\mathcal{D}_{S^\theta,x}) = \sum_{x\in\{0,1\}^n} P_{S^\theta}(x^\theta) C(\mathcal{D}_{S^\theta,x^\theta}) = \sum_{x\in\{0,1\}^n} P_S(x) C(\mathcal{D}_{S,x}) = F(S)$$

  where the second equality follows from a relabeling of the running variable x and the third equality from the assumption of permutation covariance for P_S and the definition of $\mathcal{D}_{S,x}$.
- Note that, by the coherence property of P_S, we can write for every $S \in \mathbb{H}(\Omega)$ and $\sigma \in \Omega$,

$$F(S\perp\sigma) - F(S) = \sum_{x\in\{0,1\}^{|S|+1}} P_{S\perp\sigma}(x)[C(\mathcal{D}_{S\perp\sigma,x}) - C(\mathcal{D}_{S,x_1^{|S|}}]$$

  Since, by construction, $\mathcal{D}_{S,x_1^{|S|}} \subseteq \mathcal{D}_{S\perp\sigma,x}$ (see Eq. (25)) and C is additive, it follows that the right hand side above is nonnegative. This proves that F is monotonic.
- We now fix the sequences R and S in $\mathbb{H}(\Omega)$ and $\sigma \in \Omega$ and we put $n = |R|$ and $m = |S|$. Denoting the corresponding running output sequences as x_R, x_S, and x_σ and using again the coherence property, we can write

$$\begin{aligned} F(R\perp S\perp\sigma) - F(R\perp S) - F(S\perp\sigma) + F(S) = &\sum_{x_R\in\{0,1\}^n} \sum_{x_R\in\{0,1\}^m} \sum_{x_\sigma\in\{0,1\}} P_{R\perp S\perp\sigma}(x_R\perp x_S\perp x_\sigma)[C(\mathcal{D}_{R\perp S\perp\sigma,x_R\perp x_S\perp x_\sigma}) \\ &- C(\mathcal{D}_{R\perp S,x_R\perp x_S}) - C(\mathcal{D}_{S\perp\sigma,x_S\perp x_\sigma}) + C(\mathcal{D}_{S,x_S})] \end{aligned} \tag{27}$$

  Notice now that

$$C(\mathcal{D}_{R\perp S\perp\sigma,x_R\perp x_S\perp x_\sigma}) - C(\mathcal{D}_{R\perp S,x_R\perp x_S}) = \begin{cases} 0 & \text{if } x_\sigma = 1 \\ C(\mathcal{D}_\sigma \setminus \mathcal{D}_{R\perp S,x_R\perp x_S}) & \text{if } x_\sigma = 0 \end{cases} \tag{28}$$

  while

$$C(\mathcal{D}_{S\perp\sigma,x_S\perp x_\sigma}) - C(\mathcal{D}_{S,x_S}) = \begin{cases} 0 & \text{if } x_\sigma = 1 \\ C(\mathcal{D}_\sigma \setminus \mathcal{D}_{Sx_S}) & \text{if } x_\sigma = 0 \end{cases} \tag{29}$$

  Since by construction $\mathcal{D}_{S,x_S} \subseteq \mathcal{D}_{R\perp S,x_R\perp x_S}$ and C is additive, it follows comparing (28) and (29) that

$$C(\mathcal{D}_{R\perp S\perp\sigma,x_R\perp x_S\perp x_\sigma}) - C(\mathcal{D}_{R\perp S,x_R\perp x_S}) \le C(\mathcal{D}_{S\perp\sigma,x_S\perp x_\sigma}) - C(\mathcal{D}_{S,x_S})$$

  This implies that the right hand side formula in Eq. (27) is nonpositive and, thus, F is submodular.

8. Other applicative domains: job scheduling and recommender systems

In this section, we present additional applicative examples that can be addressed within our framework.

8.1. Job scheduling

We tackle a job scheduling problem that was first studied by Stadje [24]. Assume that Ω is a set of jobs that need to be processed by a single machine subject to failure, which is modeled stochastically. We associate a number $P(\sigma)$ with each job $\sigma \in \Omega$, which represents the probability that the machine does not fail while performing σ. We assume that the machine is not aging so the probability of not failing while performing a sequence of jobs S is simply $P(S) = \prod_j P(S_j)$. Every job σ is also associated with a reward $R(\sigma) \ge 0$ and a discount $d(\sigma) \in [0, 1[$ (typically, the discount depends on the time t_σ needed to complete job σ, e.g. $d(\sigma) = e^{-at_\sigma}$). The reward of performing the job σ after the sequence of jobs S has been performed is given by $d(S)R(\sigma)$, where $d(S) = \prod_j d(S_j)$. The objective function G on a sequence of jobs S is the expected total reward under the assumption that the machine keeps processing jobs of the sequence S until it fails. Formally, we have

$$G(S) = \sum_{k=1}^{|S|} P(S|_1^{k-1}) d(S|_1^{k-1}) R(S_k) \tag{30}$$

(with the convention that $P(\varnothing)d(\varnothing) = 1$). The function G fits the class of functions in Eq. (1) and is formally equivalent to the function considered in the S&T problem described above. To see this, we put $D(S) = P(S)d(S)$ and we note that, by multiplying and dividing the k-th addend in Eq. (30) by $1 - D(S_k)$, we obtain:

$$G(S) = \sum_{k=1}^{|S|} [D(S|_1^{k-1}) - D(S|_1^{k})] \frac{R(S_k)}{1 - D(S_k)} \tag{31}$$

If we now put $F(S) = 1 - D(S)$ and $g(\sigma) = \frac{R(\sigma)}{1-D(\sigma)}$, we observe that G coincides with F_g as defined in Eq. (1). The main result reported by Stadje [24] is that, restricting G to sequences of distinct jobs of a fixed length n, the optimal solution is a sequence S for which g is decreasing, namely $g(S_1) \geq g(S_2) \geq \cdots \geq g(S_n)$. This conclusion is also implied by the general result expressed in Proposition 1. We now fix a g-ordering $\prec$ on Ω and show that the properties needed to apply Corollary 1 hold on the fully extendable set $\mathbb{H}^d(\Omega, \prec)$:

- $D(S)$ is, by construction, permutation invariant and thus also $F(S) = 1 - D(S)$ is permutation invariant.
- Note that

  $$\Delta D(S, \sigma) = D(S \perp \sigma) - D(S) = D(S)[D(\sigma) - 1]$$

  This implies (since $D(\sigma) \leq 1$ for every σ) that $\Delta D(S, \sigma) \leq 0$ for every S and σ. Hence, D is anti-monotonic and, consequently, F monotonic.
- From the equality

  $$\Delta^2 D(S, \sigma_1, \sigma_2) = D(S \perp \sigma_1 \perp \sigma_2) - D(S \perp \sigma_1) - D(S \perp \sigma_2) + D(S) = D(S)[1 - D(\sigma_1)][1 - D(\sigma_2)]$$

  it now follows that $\Delta^2 D(S, \sigma_1, \sigma_2) \geq 0$ for every S, σ_1, and σ_2. This yields the submodularity of F.
- By definition, $\mathbb{H}^d(\Omega, \prec)$ is g-ordered.

8.2. Recommender systems

Finally, we present and extend a recommender system application presented by Ashkan et al. [3]. Numerical simulations for this example are given in Section 9.

Assume that Ω is a set of movies and the function $g : \Omega \to [0, 1]$ attributes the corresponding satisfaction probability of a default user to each of them. Movies are organized under different genres, i.e. there is a set $\mathcal{T}$ of genres and a function t such that, for each $\sigma \in \Omega$, $t(\sigma) \subseteq \mathcal{T}$ is the subset of the genres covered by σ. The recommender system generates a sequence $S \in \mathbb{H}^d(\Omega)$.

The objective function $G : \mathbb{H}^d(\Omega) \to \mathbb{R}$ is the probability of the user satisfaction assuming the following stochastic model of choice: the user chooses a genre t in $\mathcal{T}$ with a probability r_t and picks the first item S_i in the given sequence for which $t \in t(S_i)$. We use the notation $i(t)$ to indicate such index i. Formally, we have that $i(t) = \min\{i = 1, \ldots, |S| \,|\, t \in t(S_i)\}$. The user will be satisfied with probability $g(S_{i(t)})$.

We can formally compute $G(S)$ as follows:

$$G(S) = \sum_{t \in \mathcal{T}} r_t \mathbb{P}(\text{satisfied} \,|\, t) = \sum_{t \in \mathcal{T}} r_t g(S_{i(t)}) = \sum_{i=1}^{|S|} \left(\sum_{t:\, i(t)=i} r_t \right) g(S_i) \tag{32}$$

Define now $F : \mathbb{H}^d(\Omega) \to \mathbb{R}$ so that

$$F(S) = \sum_{t \in \bigcup_i t(S_i)} r_t \tag{33}$$

$F(S)$ represents the probability that the chosen genre shows up in the sequence S and it holds

$$\sum_{t:\, i(t)=i} r_t = F(S|_1^i) - F(S|_1^{i-1})$$

Substituting the above expression in Eq. (32), we recognize that the function is in the form of Eq. (1).

Ashkan et al. [3] study (for the case when r_t are all equal) the optimality of the function F_g over the set of sequences of distinct items of maximal length $|\Omega|$ and discover that the solution, as in the previous example, is given by any S on which g is monotonically decreasing. In addition, they note that such optimal solution S can be trimmed by iteratively discarding all items S_i for which $F(S|_1^i) - F(S|_1^{i-1}) = 0$. In this way, they obtain the shortest possible recommended sequence of items still maximizing the satisfaction probability.

Similarly to the previous example, we now show that the properties needed to apply Corollary 1 hold on the fully extendable set $\mathbb{H}^d(\Omega, \prec)$, where $\prec$ is any fixed g-ordering. The function $F(S)$ is, by construction, permutation invariant. It is known in the literature as a *weighted coverage function*: interpreting r_t as the weight of genre t, $F(S)$ represents the global weight of the genres covered by the sequence of movies S. Such functions represent a well known example of monotonic submodular functions (see the work by Krause and Golovin [17] for details). Finally, by definition, $\mathbb{H}^d(\Omega, \prec)$ is g-ordered.

In practical applications, as also noted by Ashkan et al. [3], it may be of interest to optimize over sequences that are not necessarily of maximal length. In this direction, we propose a generalization of the above model that also leads to a function of the type of Eq. (1). Instead of assuming that a movie $\sigma \in \Omega$ covers a set of genres $t(\sigma)$, we associate a probability vector p^σ over $\mathcal{T}$ with each movie σ, where $p^\sigma(t)$ indicates to which extent movie σ covers genre t. Hence, we assume that the choice mechanism of the user is now the following: once the genre t has been selected, the user will pick S_1 with probability $p^{S_1}(t)$. If S_1 is not chosen (which will happen with probability $1 - p^{S_1}(t)$), the user will pick S_2 with probability $p^{S_2}(t)$ and so on. If $S_{i(t)}$ is the one chosen, the user will be satisfied with probability $g(S_{i(t)})$. In this case:

$$G(S) = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathbb{P}(\text{sat.}\,|\,t) = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\sum_{i=1}^{|S|} g(S_i)\mathbb{P}(i(t)=i\,|\,t)$$

where

$$\mathbb{P}(i(t)=i\,|\,t) = (1-p^{S_1}(t))\cdots(1-p^{S_{i-1}}(t))p^{S_i}(t)$$

If we now define

$$F(S) = \sum_{i=1}^{|S|}(1-p^{S_1}(t))\cdots(1-p^{S_{i-1}}(t))p^{S_i}(t)$$

as the probability that one of the items of the sequence S is eventually picked, we have that

$$\mathbb{P}(i(t)=i\,|\,t) = F(S|_1^i) - F(S|_1^{i-1})$$

This shows that, in this more general case too, the function G has the same structure of the function in Eq. (1).

In regard to the applicability of Corollary 1, note that the function $F(S)$ is identical to the success probability as defined in the detection problem and is thus permutation invariant, monotonic, and submodular. Finally, the fully extendable set on which the maximization takes place is the set $\mathbb{H}^d(\Omega, \prec)$ introduced above, which is g-ordered by definition.

9. Experimental results

To show the potential of our method, we now provide explicit numerical simulations for the S&T and recommender system applications described above.

9.1. Search and tracking

We show the advantage of using the generalized greedy algorithm over the standard one by running both algorithms on several, randomly generated S&T problems. To highlight when the two algorithms exhibit different behaviors, we consider scenarios in which the detection probability of each pattern depends on the execution time associated with it. If the patterns associated with a lower t have a high detection probability, the standard and the generalized greedy search perform similarly. They prefer these early patterns by placing them at the beginning of the sequence and add the remaining patterns to the end of the sequence. Conversely, if the detection probabilities associated with patterns with a greater t are high enough, the standard greedy immediately places those patterns at the beginning of the sequence, but, as patterns are only added on the right side of the sequence, it never exploits early search patterns. In this way, it constructs short sequences that do not make full advantage of the richness of the set Ω. Instead, the generalized greedy, being free to place patterns in any position, manages to exploit both types of patterns.

We generate 11,000 realistic problems instances, each with 20 candidate patterns and 40 destinations.[1] Each pattern σ is associated with a random sample of destinations. Time stamps are generated sequentially by taking a random sequence

[1] See supplementary material available online at https://doi.org/10.5281/zenodo.3695080.

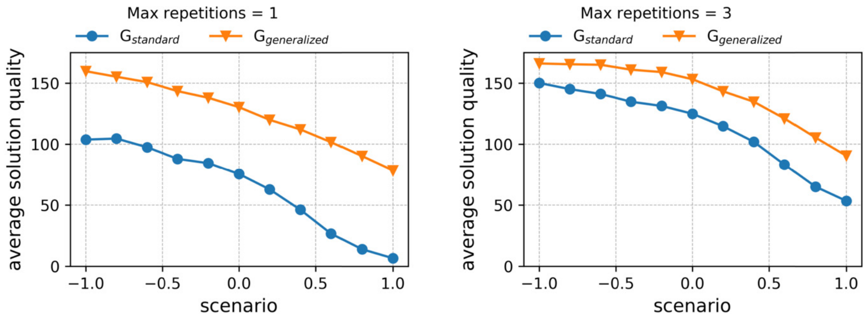

Fig. 1. Average objective values obtained by the standard and the generalized greedy algorithms.

Table 1
Average running time (in milliseconds) of the standard and the generalized greedy algorithms.

	max repetitions = 1	max repetitions = 3
$G_{standard}$	4 ± 1	15 ± 4
$G_{generalized}$	16 ± 6	67 ± 25

of all the search patterns $\bar{S} = (\sigma_1, \ldots, \sigma_n)$ and imposing that $t(\sigma_i) = t(\sigma_{i-1}) + r$, where r is a random number. The detection probability is a linear function of the indexes of the sequence of search patterns in $\bar{S}$ with different angular coefficients: $\phi_{\sigma_i} = m \cdot i + q$, where m is a value between -1 and 1, and q is such that $\sum_i \phi_{\sigma_i}$ is constant across the scenarios. A scenario with $m = -1$ corresponds to the case of patterns with a lower time stamp having higher detection probabilities, while, a scenario with $m = 1$, represents the case of patterns with higher time stamp having higher detection probabilities. When $m = 0$, all the patterns have the same detection probability.

For each problem, we run the generalized and the standard greedy algorithms over sequences of maximal length 10. We consider two types of sequences: in the first, we establish that all patterns must be distinct; in the second, instead, we allow at most 3 repetitions for each pattern.

Fig. 1 shows the average objective values found by the two algorithms for different scenarios: the left plot corresponds to the case of distinct patterns, while the right plot to the case of a maximum of 3 repetitions per pattern. The figure shows that, in all cases, the generalized greedy algorithm dominates the standard algorithm. As expected, the difference in performance is particularly high (considering the ratio) in scenarios where the search patterns associated with a greater execution time have a higher detection probability. The average running time across all scenarios of the two algorithms and across all instances is reported in Table 1. While the generalized greedy algorithm is slightly more time consuming than the standard algorithm, the runtime is acceptable for the real S&T application as the optimization of the objective function is typically performed within a time limit of one minute [2,23].

9.2. Recommender systems

For the recommender systems application, we analyze the performance of our generalized greedy algorithm for the same case study considered by Ashkan et al. in [3], and we make a comparison with the DUM algorithm proposed therein. The general setting is described in Section 8.2. Each movie σ in a set Ω is equipped with a satisfaction probability $g(\sigma)$ and a subset of genres $t(\sigma) \subseteq \mathcal{T}$. For a fixed positive integer K, Ashkan et al. [3] compare sequences of movies S of length at most K from Ω based on two performance indices: the Intra-List Distance metric (ILD) [25] and the normalized Discounted Cumulative Gain (nDCG) [26]. ILD measures the diversity of a sequence and is formally defined, for a sequence S, as

$$ILD(S) = \frac{1}{|S|^2} \sum_{i,j \le |S|} |t(S_i) \Delta t(S_j)|$$

where $t(S_i)\Delta t(S_j)$ indicates the symmetric difference between the two subsets $t(S_i)$ and $t(S_j)$. nDCG is a discounted accumulated measure of the level of satisfaction of a sequence, formally defined, for a sequence S, as

$$nDCG(S) = \frac{1}{C} \sum_{i=1}^{|S|} \frac{g(S_i)}{\log_2(i+1)}$$

where C is a normalization constant defined as the ideal gain obtained for a sequence of the same length and maximum satisfaction probability for all its elements.

The DUM algorithm considered by Ashkan et al. [3] and explained in Section 8.2 consists in sorting the elements of Ω according to their satisfaction probability g and, then, removing from the sequence all those elements that do not increment

Table 2
Comparison of Dum and Greedy.

	$K=5$		$K=10$		$K=15$	
	ILD	nDCG	ILD	nDCG	ILD	nDCG
Dum	1.85	0.78	1.91	0.72	1.84	0.72
Greedy	2.20	0.83	2.12	0.82	2.01	0.82

the value of the function F. The sequence obtained in this way has a length that is guaranteed to be below the total number of genres. Since in this case the authors want to produce a sequence of prescribed length K that is in general smaller than the number of genres ($|\mathcal{T}|=18$ in the considered case study), they apply a modification of their algorithm by substituting the original function F with the following one:

$$F(S)=\sum_{t\in\mathcal{T}}\min\{\sum_{\sigma\in\Omega}|\{\sigma : t\in t(\sigma)\}|, N_t\} \tag{34}$$

where N_t is a prescribed number of movies of genre t that are forced to appear in the sequence. These numbers N_t are chosen so that $\sum_t N_t = K$ and are constructed by making use of the user's preferences r_t of the various genres.

Specifically, the case study considered starts from the *1M MovieLens dataset*,[2] consisting of one million movie ratings from 6040 unique users, from which users with more than 300 ratings are then selected. This results in a dataset of 955 users with 502k ratings for 3644 unique movies, divided into 18 genres. For each user, rated movies are split into a training and a test set with a 2 : 1 ratio. The test set forms the set Ω of recommendable movies. Data in the training set is used to create the user's interest and genre profiles. The user's interest profile is generated using matrix factorization via singular value decomposition [27] and provides, for each movie $\sigma\in\Omega$, the satisfaction probability $g(\sigma)$. The genre profile consists in the empirical distribution r_t on the movie genres $\mathcal{T}$ obtained from the training set rated by the user.

For Dum, the numbers N_t are computed as follows [3]: first, another empirical distribution r'_t is calculated by sampling 10 elements from the original distribution r_t on the genre set, and, then, it is established that $\tilde{N}_t=\lfloor r'_t\cdot K\rfloor$. If $D=K-\sum_t\tilde{N}_t=0$, then $N_t=\tilde{N}_t$ is set. Otherwise, D more elements $t_1,\dots,t_D$ are further sampled from the distribution r_t, and, finally, it is put that $N_t=\tilde{N}_t+|\{i=1,\dots,D\,|\,t_i=t\}|$.

In our approach, instead, we apply the original function F defined in Eq. (33) using, for each user, the original distribution r_t of the genre profile and maximizing directly over sequences of prescribed length K.

We perform experiments with different values of K. Every experiment is repeated 3 times, with different training and test set splits. In Table 2, we report the comparison between Dum and our generalized greedy algorithm that maximizes Eq. (32) with respect to ILD and nDCG. As shown in the Table 2, our approach performs better than Dum against both metrics. Our improved performance is achieved thanks to the flexibility of our algorithm, which incorporates the full set of genres in the optimization problem directly, as opposed to Dum that samples a subset of genres before the construction of the sequences.

10. Conclusions

In this paper, we show that, in several applicative domains, the problem of finding a sequence of objects that maximizes a reward can be expressed as the maximization of a recursive function that exhibits the structure captured by Eq. (1). After proving that existing greedy algorithms do not yield strong theoretical guarantees for such a function, we study its properties and generalize the notions of monotonicity and submodularity by adapting them to fully extendable sets of sequences. We then introduce an efficient generalized greedy approach that ensures finding solutions that are $O(1-\frac{1}{e})$ of the optimal. Our method is general and can be applied to any domain with an objective function that can be transformed in the form of Eq. (1). To support this thesis, we present evidence that our technique works across several applications and provide explicit numerical simulations for two domains, S&T and recommender systems. The experiments directly show the power of our new algorithm. Our work contributes to the discussion on submodularity by stepping away from the specific details of practical applications and presenting general properties of functions often encountered in them, which can be exploited to find better solutions more efficiently.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

[2] https://grouplens.org/datasets/movielens/1m/.

Acknowledgements

This work has been supported by EPSRC Grant EP/S016473/1, Leverhulme Trust Grant VP1-2019-037 and MIUR Grant "Dipartimenti di Eccellenza 2018-2022" (CUP: E11G18000350001). We thank the anonymous reviewers for their detailed and rigorous reviews. Research data used for this paper is available at https://doi.org/10.5281/zenodo.3695080.

References

[1] A. Krause, C. Guestrin, Near-optimal observation selection using submodular functions, in: Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 2, AAAI'07, AAAI Press, 2007, pp. 1650–1654.
[2] C. Piacentini, S. Bernardini, C. Beck, Autonomous target search with multiple coordinated UAVs, J. Artif. Intell. Res. 65 (2019) 519–568.
[3] A. Ashkan, B. Kveton, S. Berkovsky, Z. Wen, Optimal greedy diversity for recommendation, in: Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15, AAAI Press, 2015, pp. 1742–1748.
[4] S. Tschiatschek, A. Singla, A. Krause, Selecting sequences of items via submodular maximization, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17, AAAI Press, 2017, pp. 2667–2673.
[5] S. Alaei, A. Malekian, Maximizing sequence-submodular functions and its application to online advertising, in arXiv:1009.4153v1, 2010, pp. 1–18.
[6] J. McAuley, R. Pandey, J. Leskovec, Inferring networks of substitutable and complementary products, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'15, ACM, New York, NY, USA, 2015, pp. 785–794.
[7] M. Mitrovic, M. Feldman, A. Krause, A. Karbasi, Submodularity on hypergraphs: from sets to sequences, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, AISTATS'18, PMLR, 2018, pp. 1177–1184.
[8] S. Fujishige, Submodular Functions and Optimization, Annals of Discrete Mathematics, Elsevier, 2005.
[9] G.L. Nemhauser, L. Wolsey, An analysis of approximations for maximizing submodular set functions, Math. Program. 14 (1978) 265–294.
[10] D. Golovin, A. Krause, Adaptive submodularity: theory and applications in active learning and stochastic optimization, J. Artif. Intell. Res. 42 (1) (2011) 427–486.
[11] A. Krause, C. Guestrin, Near-optimal nonmyopic value of information in graphical models, in: Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence, UAI'05, AUAI Press, Arlington, Virginia, United States, 2005, pp. 324–331.
[12] S. Dughmi, T. Roughgarden, M. Sundararajan, Revenue submodularity, Theory Comput. 8 (2012) 95–119.
[13] B. Lehmann, D. Lehmann, N. Nisan, Combinatorial auctions with decreasing marginal utilities, Games Econ. Behav. 55 (2) (2006) 270–296.
[14] S.C.H. Hoi, R. Jin, J. Zhu, M.R. Lyu, Batch mode active learning and its application to medical image classification, in: Proceedings of the 23rd International Conference on Machine Learning, ICML'06, ACM, New York, NY, USA, 2006, pp. 417–424.
[15] Z. Zhang, E.K.P. Chong, A. Pezeshki, W. Moran, String submodular functions with curvature constraints, IEEE Trans. Autom. Control 61 (3) (2016) 601–616.
[16] A. Krause, A. Singh, C. Guestrin, Near-optimal sensor placements in gaussian processes: theory, efficient algorithms and empirical studies, J. Mach. Learn. Res. 9 (2008) 235–284.
[17] A. Krause, D. Golovin, Submodular Function Maximization, Cambridge University Press, 2014, pp. 71–99, Ch. 3.
[18] M. Streeter, D. Golovin, An online algorithm for maximizing submodular functions, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems, vol. 21, Curran Associates, Inc., 2009, pp. 1577–1584.
[19] C. Qian, C. Feng, K. Tang, Sequence selection by Pareto optimization, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI'18, International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 1485–1491.
[20] M. Mitrovic, E. Kazemi, M. Feldman, A. Krause, A. Karbasi, Adaptive sequence submodularity, in: 33rd Conference on Neural Information Processing Systems, NeurIPS'19, 2019, pp. 5353–5364.
[21] S. Bernardini, M. Fox, D. Long, C. Piacentini, Deterministic vs probabilistic methods for searching for an evasive target, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI'17, 2017, pp. 3709–3715.
[22] S. Bernardini, M. Fox, D. Long, C. Piacentini, Leveraging probabilistic reasoning in deterministic planning for large-scale autonomous search-and-tracking, in: Proceedings of the 26th International Conference on Automated Planning and Scheduling, ICAPS'16, 2016, pp. 47–55.
[23] S. Bernardini, M. Fox, D. Long, Combining temporal planning with probabilistic reasoning for autonomous surveillance missions, Auton. Robots 41 (1) (2017) 181–203.
[24] W. Stadje, Selecting jobs for scheduling on a machine subject to failure, Discrete Appl. Math. 63 (3) (1995) 257–265.
[25] M. Zhang, N. Hurley, Avoiding monotony: improving the diversity of recommendation lists, in: Proceedings of the 2008 ACM Conference on Recommender Systems, 2008, pp. 123–130.
[26] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of IR techniques, ACM Trans. Inf. Syst. 20 (4) (2002) 422–446.
[27] D. Scott, S. Dumais, G.W. Furnas, T.K. Landauer, R. Harshman, Indexing by latent semantic analysis, J. Am. Soc. Inf. Sci. Technol. 41 (6) (1990) 391–407.