Local search and swapping strategies. Challenging the greedy maximization of a polymatroid subject to a cardinality constraint

Mirco Soffritti MIRCO.SOFFRITTI@CAMPUS.EAE.ES

Editor:

Abstract

This paper studies the maximization of a polymatroid subject to a cardinality constraint. In particular, we consider the problem of improving the value of the greedy set by swapping one of its members with an element that does not belong to it. To achieve this goal, we first define a (set-based) *post-greedy* measure of curvature of a polymatroid. Then we use it to build a non-recursive test that, in polynomial time, provides sufficient conditions for the greedy collection to be 'locally' optimal. Finally, we generalize our test by considering larger swaps of elements and verify that, as the number of swapped elements increases, the likelihood for the greedy collection to remain locally maximal inevitably deteriorates.

Keywords: Polymatroid, greedy algorithm, cardinality constraint, local search, swaps, and post-greedy curvature.

1. Introduction

Submodularity is a property of set functions that is of application in many branches of Science. Examples of problems that have been structured as subset selection within a submodular framework are: electrical systems and sensor networks (Krause and Guestrin, 2005, 2007; Krause et al., 2008), document summarization (Shen and Li, 2010; Lin and Bilmes, 2011), algorithmic game theory and inventory management (Gerchak and Gupta, 1991; Hartman et al., 2000; Schulz and Uhan, 2013), image segmentation (Kohli et al., 2009; Jegelka and Bilmes, 2011), nonparametric learning (Reed and Ghahramani, 2013), viral marketing and social networks (Kempe et al., 2003; Mossel and Roch, 2007), etc.
In this paper we are specifically interested in the problem of submodular maximization subject to a cardinality constraint, which is a condition on the maximum number of elements that we are allowed to include in our solution. When the set of elements to be considered is large, an extensive inspection of all the feasible collections quickly becomes impractical and the greedy heuristic represents the best polynomial-time and deterministic procedure that we can hope for the solution to our problem (Feige, 1998). Even though the greedy set comes with no validation of its 'global' and 'local' optimality, we still wonder whether or not it represents an optimal solution. Inspired by the 2-opt algorithm designed by Croes (1958), in our work we will challenge the 'local' superiority of the greedy collection by swapping some of its elements with an identical number of elements not belonging to it. To achieve this, we first introduce a (set based) 'concavity' index, which we call *post-greedy curvature*. Then, we use this index to design a test that establishes a set of sufficient

conditions for the greedy set to be maximal within its local neighborhood. Not only our test runs in polynomial time, but knowing that the greedy output is locally optimal saves us the time that is required to undergo a brute-force evaluation of the competing collections that belong to the same neighborhood. In the last part of our paper we generalize our base-case by discussing the idea of swaps of size greater than one, i.e. replacements where the number of expelled elements from the greedy set still coincides with number of external elements that are introduced to replace them. The generalized version of the post-greedy curvature of a polymatroid remains the foundation stone of the associated heuristic, which we will call TWO-STEP POST-GREEDY ALGORITHM of size j. The associated general test verifies that as the size of the swap increases, the status of the greedy set as a local maximum deteriorates, while increases the likelihood of finding an enhancing local solution. To the best of our knowledge, no paper on polymatroidal maximization has been done to investigate swapping strategies that are designed to outperform a greedy output.

Our paper is organized as follows. We structure Chapter 2 around the concept of polymatroid maximization subject to a cardinality constraint. In Chapter 3 we present the ADAPTIVE GREEDY ALGORITHM (AG) and we briefly examine the limits and the advantages of its application to our maximization problem. After reviewing a number of measures of 'concavity' that are prominent in the existing literature, we organize Chapter 4 around the concept of *post-greedy curvature* of a polymatroid and we explore its properties. This index represents the building block for the swapping strategies that, in the second part of this paper, will be analyzed to produce new feasible solutions that may (or may not) enhance the output of the AG heuristic. In the same chapter, we will explore the idea of swapping some elements of the of the greedy set with an identical number of ground elements that do not belong to it. In Chapter 5 we make use of the post-greedy curvature to design a step-one formal test on the local optimality of the greedy output. In Chapter 6 we generalize the base-case test and discuss the idea of a swapping procedure of size greater (or equal) to one. The post-greedy curvature of a polymatroid will remain the foundation stone of the associated heuristic. Chapter 7 will collect the conclusions of our work and put forward a number of possible extensions and lines of investigation.

Throughout the paper, we use a white square '□' to indicate where the implementation of a concept in our recurrent example is complete. A black square '■' is used to indicate that a formal demonstration is achieved. The proof of theorems is provided within the text, while that of all lemmas and corollaries is deferred to the Appendix.

2. Submodularity and formal background

2.1 Polymatroid functions and notation

Consider a non-empty universe of elements (or ground set) $V = \{v_1, v_2, \dots, v_n\}$ where 2^n is the cardinality of the power set of V, which we denote with 2^V. Consistently with the existing literature, we assume that the function f can be accessed via a value oracle (also called incremental oracle), which is an efficient computational black-box that, given a set $S \subseteq V$, returns the value $f(S)$ in polynomial time.
For the sake of compactness, we now introduce some incremental notation. Given a set function $f : 2^V \to \mathbb{R}$, along with $S, T, U \subseteq V$, $U \subseteq S$, and $v \in V$, we define:

1. $f(S+v) := f(\{S \cup \{v\}\})$ and $f(S-v) := f(\{S \setminus \{v\}\})$,
2. $f(S+T) := f(\{S \cup T\})$ and $f(S-T) := f(\{S \setminus T\})$,
3. $f(T|S) := f(S+T) - f(S)$, and
4. $f(v|S) := f(\{v\}|S)$.

In particular, $f(v|S)$ is referred to as the *discrete derivative* (or marginal increment) of f at S with respect to v. A set function $f : 2^V \to \mathbb{R}$ is labelled as *submodular* if:

$$\text{for every } S \subseteq T \subseteq V,\ v \in V \setminus T: \quad f(v|S) \geq f(v|T). \tag{1}$$

This definition states that adding an element to a larger set results in a non-smaller discrete derivative of f than adding it to any of its subsets. A less intuitive but perhaps more useful definition of submodularity is given by:

$$\text{for every } S, T \subseteq V,\ f(S) + f(T) \geq f(S \cup T) + f(S \cap T), \tag{2}$$

which is equivalent to (1) (Buchbinder and Feldman, 2018). A submodular function can be further classified as:

- *normal*, if $f(\emptyset) = 0$,
- *non-negative*, if for every $S \subseteq V$, $f(S) \geq 0$,
- *monotone*, if for every $S \subseteq T \subseteq V$, $f(S) \leq f(T)$,

A submodular function that is normal, non-negative, and monotone, is referred to as *polymatroid* and it exhibits the following characterization (Nemhauser et al., 1978a):

$$\text{for every } S \subseteq T \subseteq V : f(T) \leq f(S) + \sum_{v \in (T-S)} f(v|S).$$

When both expressions (1) and (2) are satisfied as an equality, then f is called *modular* function. In practice, a modular application is just a special kind of a submodular function.

2.2 Matroids and constrained maximization

Assume that $\mathcal{I}$ is a family of subsets of V. If $\mathcal{I}$ satisfies the following three conditions:

i. $\emptyset \in \mathcal{I}$,

ii. if $S \subseteq T$ and $T \in \mathcal{I}$, then $S \in \mathcal{I}$,

iii. if $S, T \in \mathcal{I}$ and $|S| < |T|$, then there exists $e \in (T - S)$ such that $(T + e) \in \mathcal{I}$,

then the pair $(V, \mathcal{I})$ is called *matroid* and each $S \in \mathcal{I}$ is referred to as an *independent set* of $(V, \mathcal{I})$, and $\mathcal{I}$ as a *family of independent sets* of $(V, \mathcal{I})$ (Whitney, 1935). A set $S' \in \mathcal{I}$ is called a *basis* of $(V, \mathcal{I})$ if it is an independent set of maximal cardinality, that is $S' = \underset{S \in \mathcal{I}}{\operatorname{argmax}} |S|$. We define *rank* of the matroid, $rank(V, \mathcal{I})$, the cardinality of a (any) basis of the matroid.

If we further assume that $\mathcal{I} = \left\{S \in 2^V : |S| \leq k < n\right\}$, for some $k \in \mathbb{N}$, then $(V, \mathcal{I})$ is called *uniform matroid*. For a uniform matroid, any subset of V with k elements performs as a basis of $(V, \mathcal{I})$, being k the *rank* of the matroid (Oxley, 2006). Matroidal constraints are of great relevance in many applications presented in the recent literature (Chekuri and Kumar, 2004; Fleischer et al., 2006; Buchbinder et al., 2014).
In our work we focus on the problem of maximization of a polymatroid subject to a uniform matroid. In particular, our objective is to find $S \subseteq V$ of cardinality (at most) k that maximizes f. The problem's inputs are the function f and the cardinality level k, while its output is a set $S \subseteq V$ of size $|S| \leq k$ maximizing $f(S)$ (Calinescu et al., 2011; Liu et al., 2020). Formally:

$$\max_{S \subseteq V} f(S), \text{ s.t. } S \in \mathcal{I} \subseteq V \text{ where } \mathcal{I} = \{S \subseteq V : |S| \leq k < n\}. \tag{3}$$

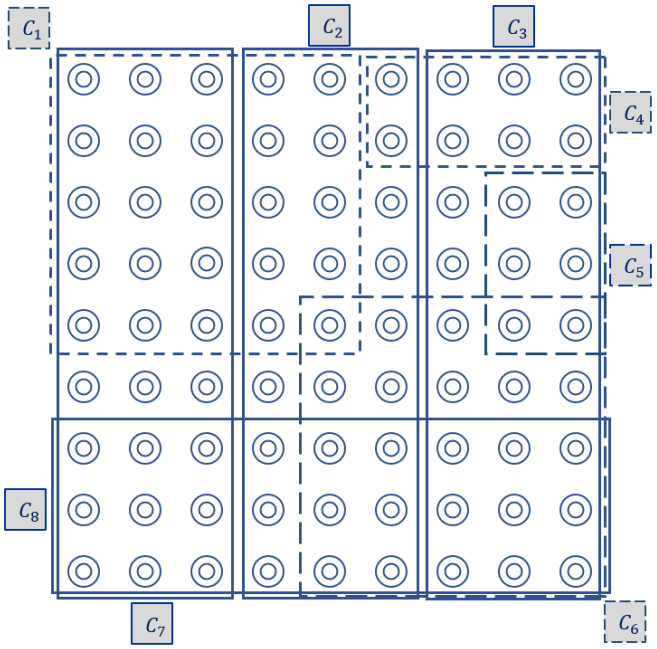
In addition, if we make the assumption that f is a polymatroid, then it holds that for every $v \in V$, $f(v|S) \geq 0$. When referring to a *globally optimal* solution to Problem (3), we will use the notation $O = \{o_1, \ldots, o_k\} \subseteq V$. Formally:

$$O = \underset{S \in \mathcal{I}}{\operatorname{argmax}} f(S).$$

2.2.1 Example

To make sense of the main ideas that we discuss in this paper, we will rely on a practical example for the submodular solution of a maximum-coverage problem (Roughgarden, 2020). Henceforth, we will refer to it as the 'CCTV example'.
The management of a company running a number of parking locations just signed a new rental contract with the owner of a lot that allocates up to 81 vehicles, disposed on nine rows and nine columns. As displayed by Figure 1, the parking garage already counts on a pre-installed CCTV security system composed of eight cameras, $C_1, \ldots, C_8$, where C_i denotes the i-th generic camera ($i = 1, 2, \ldots, 8$). In our example the ground set is $V = \{C_1, \ldots, C_8\}$, that is the collection of the cameras composing the CCTV system. In particular, two angular devices that are installed in the north-west and south-east corners of the garage (C_1 and C_6) can perform 90-degree surveillance on 25 cars each. Of the remaining six cameras, four oversee 27 vehicles each (C_2, C_3, C_7, and C_8), and two (C_4 and C_5) operate on a smaller range of eight and six vehicles. Notice that the individual ranges of surveillance partially overlap. Let's now consider the function $f : 2^V \to \mathbb{R}$ that associates to each $S \subseteq V$ the total number of vehicles that are monitored by the devices in S. For instance, if $S = \{C_1, C_2, C_3\}$, then $f(S) = |C_1 + C_2 + C_3| = 69$.
The 'submodular nature' of f can be verified by acknowledging the existence of a non-increasing discrete derivative with respect to each $C_i \in V$. For example, for $\emptyset \subseteq C_5 \subseteq C_3 \subseteq V$ and the set C_6, it holds that $[f(C_6|\emptyset) = 25] \geq [f(C_6|C_5) = 23] \geq [f(C_6|C_3) = 10] \geq [f(C_6|V) = 0]$.
The 'polymatroidal nature' of f is also easy to assess. First, as a measure of cardinality, f is trivially non-negative. Second, for the normality of f we observe that when all cameras are off, then $f(\emptyset) = 0$. Finally, monotonicity requires verification for each pair of subsets S, T such that $S \subseteq T \subseteq V$. For example, for $C_5 \subseteq C_3 \subseteq V$ it holds that $[f(C_5) = 6] < [f(C_3) = 27] < [f(V) = 81]$.

Figure 1: CCTV system for a parking garage



For simplicity, we assume that the operation cost of each camera is independent on its range of activity. Although the management's objective is to maximize the total number of vehicles that the CCVT system can monitor, it decides that no more than $k = 3$ cameras are allowed to be in operation. In practice, it needs to solve the following maximum-coverage problem:

$$\max_{S \subseteq V, |S| \leq 3} f(S),$$

which admits $O = \{C_2, C_3, C_7\} = V$ as the globally optimal solution, with $f(O) = 81$. □

3. Adaptive greedy heuristic

Even though the problem of maximizing a polymatroid subject to a uniform matroid has applications in many practical settings, for many instances of Problem (3) finding its brute-force solution is known to be a NP-hard task (unless P=NP) (Sviridenko, 2004; Feige, 1998; Krause and Guestrin, 2005). The *adaptive greedy algorithm* is a combinatorial optimization paradigm whose intent is to approach the global optimum by reducing the initial problem into a sequence of myopic and easier maximization problems and by making the locally optimal choice in each. The greedy heuristic for submodular optimization is *adaptive* in the sense that the marginal contribution of any given element $v \in V$ typically depends on the choices that were previously made by the algorithm itself (Goundan and Schulz, 2007). Unlike the dynamic programming paradigm, the greedy heuristic never reconsiders the choices that were made before each step.
Formally, if we denote with $G_{i-1} \subseteq V$ the subset selected by the algorithm in its $(i-1)^{\text{th}}$

step, then $G_i = G_{i-1} + g_i$, where

$$g_i \in \left\{ \underset{v \in (V - G_{i-1})}{\operatorname{argmax}} f(v|G_{i-1}) \right\}$$

for $i = 1, 2, \ldots, k$, and $G_0 =: \emptyset$. The AG algorithm, whose time complexity does not depend on the cardinality of the ground set V, makes $\#(G_k) = \sum_{j=0}^{k}(n-j) = \mathcal{O}(nk)$ queries to the incremental oracle and, for this reason, it runs in polynomial time. The following pseudo-code, shows how the greedy set G_k is built, one element at a time. The heuristic initializes the solution and its image in line 1 and 2, respectively. At each iteration of the **for** loop in lines 4 to 6, the algorithm moves to its best incremental set by first choosing a new element from the ground set (line 4) and then by adding it to the current solution (line 5). The value of the greedy set is finally updated in line 6. The iterative construction ends when the cardinality requirement is met. This is when the solution G_k and its value $f(G_k)$ are returned in line 7.

Algorithm 1: Adaptive greedy algorithm (AG)

Input: $f : 2^V \to \mathbb{R}_+,\ 1 \leq k \leq n,$ value query oracle for f.
Output: $G_k \subseteq V : |G_k| \leq k$, with greedy image.

1 $G_k \leftarrow \emptyset$;
2 $f(G_k) \leftarrow 0$;
3 **for** $i = 1, 2, \ldots, k$ **do**
4 $\quad g^* = \underset{u \in (V - G_k)}{\operatorname{argmax}} f(u|G_k)$ (choose randomly in case of ties);
5 $\quad G_k \leftarrow G_k + g^*$;
6 $\quad f(G_k) \leftarrow f(G_k + g^*)$;
7 **return** G_k, $f(G_k)$.

In the continuation of our work, we will need to reference to the following lemma:

Lemma 1 *Let Problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$. *Given the sequence* $G_0 \subseteq \ldots \subseteq G_i \subseteq \ldots \subseteq G_n$, *with* $G_0 := \emptyset$, *the greedy marginal increment* $f(g_{i+1}|G_i)$ *is non-increasing with* i.

The AG algorithm exhibits a known approximate correctness guarantee. In particular, when f is a uniform matroid, then the greedy heuristic achieves an approximation with a tight lower bound ratio of $\left(1 - e^{-1}\right)$ to the problem's maximum (Nemhauser et al., 1978a,b). Finally, Feige (1998) shows that for any $\lambda > 0$ it is NP-hard to achieve a $(1 - e^{-1} + \lambda)$-approximation for the max k-cover problem, which is a special case of the Problem (3), when f is a polymatroid.
In the literature on submodular optimization, the structural incapability of the AG algorithm to always produce the optimum set is often referred to as the 'submodular trap of greedy optimality'. The property of 'decreasing' discrete derivatives of f can be exploited to implement the 'lazy' version of the adaptive greedy algorithm, which is an accelerated

version of the standard adaptive AG heuristic we described above. In its first step, the lazy procedure behaves identically to its standard counterpart (Minoux, 1978). However, starting from the second step, instead of computing $f(u|G_{i-1})$ for each element of $(V - G_{i-1})$, the LAZY GREEDY proceeds according to the following sequence:

i) We keep a list of the upper bounds $\rho(u)$ on the discrete derivatives at G_{i-1} for each element of $(V - G_{i-1})$, and sort them in decreasing order.

ii) We evaluate $f(u|G_{i-1})$ only for element u_i^* with $\rho()$ on top of the list, and we update its upper bound. That is $\rho(u_i^*) \leftarrow f(u_i^*|G_{i-1})$.

iii) If, after the update, $\rho(u_i^*) \geq \rho(u)$ for all $u \in (V - G_{i-1})$, then we are guaranteed that u_i^* is the element with the largest discrete derivative and we set $G_i \leftarrow G_{i-1} + u_i^*$. If, instead, $\rho(u_i^*) < \rho(u)$ for some $u \in (V - G_{i-1})$, we return to step ii) and we apply its instructions to the second element of the list.

iv) And so on.

Even though the number of queries to the incremental oracle cannot be generalized to any possible instance of Problem (3), the lazy greedy algorithm may lead to orders of magnitude speedups. It has been shown that if the standard greedy outcome is unique, then the solution of the accelerated and the standard greedy algorithm are identical (Minoux, 1977).

3.0.1 CCTV example

Regarding our parking garage example, the AG algorithm begins by turning on the camera C_8. Since in this paper we are discussing how to escape the 'trap' of greedy optimality, and given that $f(C_2) = f(C_3) = f(C_7) = f(C_8) = 27$, we assume that C_8 is randomly chosen first. Subsequently, since $f(C_1|C_8) = 25$ and $f(C_2|C_8) = f(C_3|C_8) = f(C_7|C_8) = 18$, then C_1 is chosen as a second camera. Finally, to obtain the greatest increase in the number of monitored vehicles, camera C_3 is turned on. The greedy set $G_3 = \{C_1, C_3, C_8\}$ corresponds to the gray area in Figure 2. Table 1 presents the lists of discrete derivatives calculated by the value oracle in each of the three steps of the greedy selection process, with the convention that a $\boxed{\text{boxed}}$ number measures the change in f generated by a candidate camera that is not chosen, while a number in **bold** represents the discrete derivative with respect to a camera that is selected by the AG procedure.
Given that $[f(G_3) = 70] < [f(O) = 81]$, the greedy collection does not achieve global optimality. Also, since a full coverage of the ground set could be obtained by turning on cameras C_2, C_3, and C_7, this means that we had fallen in the 'trap' of greedy optimality. Indeed, in the first iteration of the **for** loop we had the option of choosing any camera covering 27 vehicles but a random (and unfortunate) choice of C_8. Finally, notice that

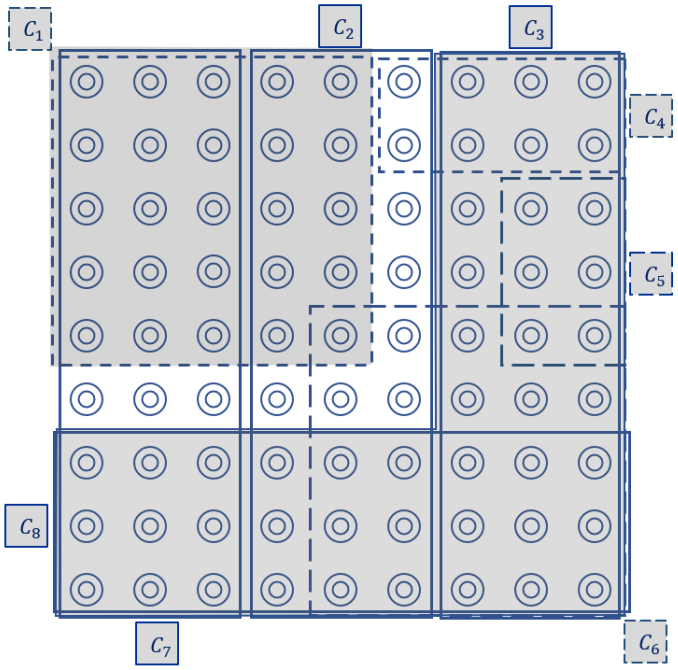
$$\left[\frac{f(G_3)}{f(O)} = \frac{70}{81} = 0.864\right] > \left[\left(1 - \frac{1}{e}\right) = 0.632\right],$$

showing that, in our example the AG algorithm complies with the general lower bound established by Nemhauser et al. (1978a,b).
In Table 2 we list the incremental evaluations made by the accelerated version of the greedy

Table 1: The discrete derivatives of the AG heuristic [CCTV example]

C_i	$f(C_i\|\emptyset)$	$f(C_i\|\{C_8\})$	$f(C_i\|\{C_1, C_8\})$
C_1	25	**25**	×
C_2	27	21	8
C_3	27	21	**18**
C_4	8	8	8
C_5	6	6	6
C_6	25	10	9
C_7	27	21	3
C_8	**27**	×	×

Figure 2: Greedy set G_3 (grey area) [CCTV example]



algorithm. In the first iteration, all cameras are considered and a random choice is made in favor of C_8. However, in the second iteration, the query oracle is initially asked to evaluate only the five cameras that are candidate to outperform C_4, and so on. As expected, the accelerated heuristic outputs the same collection as the AG ALGORITHM but reduces the number of queries to the value oracle by approximately one fourth. □

Table 2: The discrete derivatives of the lazy greedy heuristic [CCTV example]

C_i	$f(C_i\|\emptyset)$	$f(C_i\|\{C_8\})$	$f(C_i\|\{C_1, C_8\})$
C_1	25	**25**	×
C_2	$\boxed{27}$	21	8
C_3	$\boxed{27}$	21	**18**
C_4	8	-	-
C_5	6	-	-
C_6	25	10	-
C_7	$\boxed{27}$	21	3
C_8	**27**	×	×

4. Curvatures

4.1 Classic curvatures

In a well noted article on submodular maximization, Conforti and Cornuéjols (1984) define the *total curvature* of a normal and monotone submodular function as

$$\alpha := \max_{z \in V} \frac{f(z) - f(z|V-z)}{f(z)} = 1 - \min_{z \in V} \frac{f(z|V-z)}{f(z)},$$

which is an index designed to 'measure' the overall degree of 'concavity' of f over its discrete domain (Vondrák, 2010). Similarly, and after defining the greedy sequence $\emptyset \subseteq G_1 \subseteq \dots G_{k-1} \subseteq G_k$, they introduce the concept of *greedy curvature* which, for the case of a polymatroid, can be expressed as:

$$\alpha_G := 1 - \min_{z \in G_k} \frac{f(z|G_k - z)}{f(z)}, \tag{4}$$

capturing the overall degree of 'concavity' of f along the process of formation of the set G_k. Notice that the greedy curvature is computable with only k queries to the evaluation oracle.

Conforti and Cornuéjols (1984) also show that $0 \leq \alpha_G \leq \alpha \leq 1$, that $\alpha = 0$ if and only if f is modular, while α_G could be zero even when f is not modular. Finally, they show that for the maximization of a polymatroid subject to a cardinality constraint the following two bounds hold:

$$\frac{f(G_k)}{f(O)} \geq \frac{1}{\alpha}\left(1 - e^{-\alpha}\right) \quad \text{and} \quad \frac{f(G_k)}{f(O)} \geq 1 - \alpha_G \tag{5}$$

$$\frac{f(G_k)}{f(O)} \geq \frac{1}{\alpha}\left[1 - \left(1 - \frac{\alpha}{k}\right)^k\right] \geq \frac{1}{\alpha}\left(1 - e^{-\alpha}\right) \quad \text{and} \quad \frac{f(G_k)}{f(O)} \geq 1 - \alpha_G \frac{(k-1)}{k} \geq 1 - \alpha_G \tag{6}$$

which are expressed in terms of total curvature and greedy curvature, respectively.

Finally, notice that in case that the objective function f is modular, i.e. when $\alpha_G = \alpha = 0$, expressions (6) confirm that the greedy algorithm finds an optimal solution.

4.2 Post-greedy curvature

In addition to the concepts of *total curvature* and *greedy curvature*, the recent literature on submodular maximization produced a number of additional notions that have been extensively used to express the second-order property of the set function f. See, for example, the concept of *elemental curvature* introduced by Wang et al. (2014) and that of *partial curvature* defined by Liu et al. (2019).
In our work we design a new set-based measure, which we call *post-greedy curvature* of the polymatroid f:

$$\gamma_{k,j} := \max_{Z_j \subseteq G_{k+j}} \frac{f(Z_j) - [f(G_{k+j}) - f(G_{k+j} - Z_j)]}{f(Z_j)} = 1 - \min_{Z_j \subseteq G_{k+j}} \frac{f(G_{k+j}) - f(G_{k+j} - Z_j)}{f(Z_j)}, \tag{7}$$

where $Z_j =: \{z_1, \dots, z_j\} \subseteq G_{k+j}$, and $j = 1, 2, \dots, n-k$. This index, whose number of calls to the evaluation oracle is of the order of $\mathcal{O}(k+j)$, is designed to capture the 'concavity' of f in the sequential path that generates the greedy sets G_{k+j} starting from its predecessor G_k, via the incremental contribution provided by the collections Z_j.
The key features of the post-greedy curvature are presented in the following lemma.

Lemma 2 *For Problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, the post-greedy curvature owns the following properties:*

- ***i.*** *$\gamma_{k,j}$ takes its value in the interval* $[0; 1]$*;*
- ***ii.*** *if f is modular, then $\gamma_{k,j} = 0$ for all $j \geq 1$ and $k < n$;*
- ***iii.*** *$\gamma_{k,j}$ is non-increasing with j;*
- ***iv.*** *as k tends to $(n-1)$, $\gamma_{k,1}$ approaches α.*

In Chapter 5 and Chapter 6 we will make an extensive application of the index $\gamma_{k,j}$ when introducing sufficient conditions for the implementation of swapping moves that build upon the greedy output.

4.2.1 CCTV Example

Let's consider again the example regarding the parking garage. Table 3 collects the relevant magnitudes for the calculation of the classic curvatures α and α_G and of the post-greedy curvature $\gamma_{3,1}$. In particular, for the post-greedy curvature, the greedy set associated to a cardinality $k = 4$ is given by $G_4 = \{C_1, C_2, C_3, C_8\}$.
In the context of our example, $\alpha = 1$, $\alpha_G = 0.\overline{333}$ and $\gamma_{k,1} = 0.\overline{703}$. After taking the 'law' of non-increasing discrete derivatives into account, it holds that $0 \leq \alpha_G \leq \gamma_{k,1} \leq \alpha \leq 1$. As expected, the calculation of $\gamma_{k,1}$ requires us to to make $\mathcal{O}(k+1) = |G_{k+1}| = 4$ calls to the evaluation oracle. □

Table 3: The magnitudes for the calculation of α, α_G, and $\gamma_{3,1}$ [CCTV example]

C_i	$f(C_i)$	$1-\frac{f(C_i\|V-C_i)}{f(C_i)}$, $C_i \in V$	$1-\frac{f(C_i\|G_3-C_i)}{f(C_i)}$, $C_i \in G_3$	$1-\frac{f(C_i\|G_4-C_i)}{f(C_i)}$, $C_i \in G_4$
C_1	25	1	0	0.400
C_2	27	$0.\overline{888}$	–	$0.\overline{703}$
C_3	27	$0.\overline{925}$	$0.\overline{333}$	$0.\overline{333}$
C_4	8	1	–	–
C_5	6	1	–	–
C_6	25	1	–	–
C_7	27	$0.\overline{888}$	–	–
C_8	27	1	$0.\overline{333}$	$0.\overline{666}$

5. Improving strategies and simple swaps

5.1 Local search and neighborhoods

Given the greedy set, our main goal is to force an improvement in the polymatroid f by implementing a replacement strategy, that is by substituting some elements of G_k with a corresponding number of elements of $(V-G_k)$. To analyze such strategy we follow Resende and Ribeiro (2016) and introduce a number of new concepts and technical terms.
Starting from G_k, a *local search* consists of a strategy that explores its (feasible or infeasible) neighbors to find an improving solution whose cardinality is still k, but without iteration. Such a neighborhood search may either yield no improvement in the value of in f, or a 'better' local maximum, or a global maximum, depending on the procedure being used, the kind of neighborhood that is object of consideration, and the instance of Problem (3) that we are dealing with. In this paper we consider *swapping neighborhoods* of G_k, which can be defined as the collection of all the $S \subseteq V$ that can be reached from G_k by applying a swap. A *swap* is a move that adds to G_k one or more members of V and then, to satisfy the cardinality requirement, it expels from it an identical number of elements. A G_k's neighborhoods of 'size' $j \geq 1$ is formally defined as:

$$\mathcal{B}^j_{G_k} = \{S \subseteq V, |S| = k : S = ((G_k + U) - Z), \text{ with } U, Z \subseteq V, \text{ and } |U| = |Z| = j\},$$

Technically speaking, $\mathcal{B}^j_{G_k}$ is a mapping that associates G_k with the collection of the feasible solutions that can be reached from the greedy set by operating one (and only one) swap of j elements. Note that $S' \in \mathcal{B}^j_{G_k}$ whenever $G_k \in \mathcal{B}^j_{S'}$. In addition, since G_k is also a member of $\mathcal{B}^j_{G_k}$, then any $S' \in \mathcal{B}^j_{G_k}$ differs from G_k either by zero or by an even number of elements.

5.2 Neighborhood and swaps of size one

In the language of discrete optimization, a 2-opt algorithm is a local search technique that was first designed by Flood (1956) and Croes (1958) to improve the initial solution of any instance the Traveling Salesman Problem (TSP). In our work we adapt the underlying idea of a 2-OPT by making it suitable for our needs. In particular, starting from G_k, we

undergo a search in the neighborhood

$$\mathcal{B}^1_{G_k} = \{S \subseteq V, |S| = k : S = (G_k + u - z), \text{ with } u, z \in V\},$$

which is designed to produce a *swap of size one*. This is a move that adds to the greedy set a new element of the ground set and then implements a reversal operation that ejects another (or the same) element. Among the existing procedures of size one, we consider the following two:

- *One-step post-greedy heuristic of size one.* The 1SPG1 ALGORITHM is a procedure that replaces the greedy set with its *locally optimal* neighbor. This is done by browsing through the entire set $\mathcal{B}^1_{G_k}$, that is:

$$S^*_k = \underset{S \in \mathcal{B}^1_{G_k}}{\text{argmax}} f(S).$$

- *Two-step post-greedy heuristic of size one.* The 2SPG1 ALGORITHM is a two-step heuristic that constructs the set $\hat{S}_k$ by first determining the 'best' element $\hat{u} \in (V - G_k)$ to add to G_k, and then by calculating the 'best' element z to exclude from $(G_k + \hat{u})$. That is:

$$\hat{S}_k = [(G_k + \hat{u}) - \hat{z}] \in \mathcal{B}^1_{G_k} \text{ such that } \hat{u} = \underset{u \in V - G_k}{\text{argmax}} f(u|G_k) \text{ and}$$

$$\hat{z} = \underset{z \in (G_k + \hat{u})}{\text{argmin}} f\left(z|(G_k + \hat{u}) - z\right).$$

  Now, since

$$g_{k+1} = \underset{u \in (V - G_k)}{\text{argmax}} f(u|G_k),$$

  then,

$$\hat{S}_k = (G_{k+1} - \hat{z}) \text{ with } \hat{z} = \underset{z \in G_{k+1}}{\text{argmin}} f(z|G_{k+1} - z).$$

In both procedures, the associated *search space graph* $(\mathcal{N}^1, \mathcal{E}^1)$, has as a *node set* that corresponds to the collection of all feasible solutions $\mathcal{N}^1 =: \{S \subseteq V : |S| = k\}$, while its *edge set* $\mathcal{E}^1$ collects the ordered pairs (S, G_k) such that $S \in \mathcal{B}^1_{G_k}$.

In what follows we present the pseudo-code for these two heuristics.
We begin with the 1SPG1 ALGORITHM, which admits the greedy solution and its swapping neighborhood as inputs. The procedure initializes the solution and its value in lines 1 and 2, respectively. A flag (*imp*) indicating whether or not an improving collection was found is set to .FALSE. in line 3. At each iteration of the **forall** loop in lines 5 to 7, the algorithm scans all the elements of the swapping neighborhood of G_k and replaces the current solution with a neighbor that has a greater image. In lines 9 to 11 the **if** loop takes care of comparing the greedy solution with its 'best' neighbor. If the test delivers a success (that is if $f^* > f(G_k)$), then the current solution and its value are updated in line 9 and 10, and the flag is reset to .TRUE. in line 11, communicating that the one-step swap was able to

Algorithm 2: One-step post-greedy algorithm of size one (1SPG1)

Input: $f : 2^V \to \mathbb{R}_+$, G_k, $f(G_k)$, $\mathcal{B}^1_{G_k}$, and the value query oracle for f.
Output: $S^*_k \subseteq \mathcal{B}^1_{G_k}$, and f^*_k (the greatest one-step improvement w.r.t. G_k).

1 $S^*_k \leftarrow G_k$;
2 $f^*_k \leftarrow f(G_k)$;
3 $imp \leftarrow$ .FALSE.;
4 **forall** $S' \in \mathcal{B}^1_{G_k}$ **do**
5 **if** $f(S') > f^*_k$ **then**
6 $S^* \leftarrow S'$;
7 $f^* \leftarrow f(S')$;
8 **if** $f^* > f(G_k)$ **then**
9 $S^*_k \leftarrow S^*$;
10 $f^*_k \leftarrow f^*$;
11 $imp \leftarrow$ .TRUE.;
12 **return** S^*_k, f^*_k, *imp*.

Algorithm 3: Two-step post-greedy algorithm of size 1 (2SPG1)

Input: $f : 2^V \to \mathbb{R}_+$, G_k, $f(G_k)$, and the value query oracle for f.
Output: $\hat{S}_k \subseteq \mathcal{B}^1_{G_k}$ and $\hat{f}_k$ (the greatest two-step improvement w.r.t. G_k).

1 $\hat{S}_k \leftarrow G_k$;
2 $\hat{f}_k \leftarrow f(G_k)$;
3 $imp \leftarrow$ .FALSE.;
4 $\hat{u} = \underset{u \in (V - \hat{S}_k)}{\operatorname{argmax}} f(u|\hat{S}_k)$ (choose arbitrarily/randomly in case of ties);
5 $\hat{S}_k \leftarrow \hat{S}_k + \hat{u}$;
6 $\hat{f}_k \leftarrow f(\hat{S}_k)$;
7 **if** $\hat{f}_k > f(G_k)$ **then**
8 $\hat{z} = \underset{z \in \hat{S}_k}{\operatorname{argmax}} f(\hat{S}_k - z)$ (choose arbitrarily/randomly in case of ties);
9 $\hat{S}_k \leftarrow \hat{S}_k - \hat{z}$;
10 $\hat{f}_k \leftarrow f(\hat{S}_k)$;
11 $imp \leftarrow$ .TRUE.;
12 **return** $\hat{S}_k$, $\hat{f}_k$, *imp*.

produce a better solution than the greedy output. Otherwise, the flag retains its .FALSE. value. In line 12 the 1SPG algorithm returns the best-improving collection S^*_k, its value f^*_k, and the flag *imp*.

The 2SPG1 algorithm initializes the solution and its image in line 1 and 2, respectively, while in line 3 a flag is s set to .FALSE.. In lines 4 and 5, the heuristic scans all

neighborhood of G_k and replaces the current solution with the neighbor (which may or may not exist) that exhibits the greatest marginal improvement. To recreate feasibility, the **if** loop in lines 7 to 11 compares the value of the updated solution with that of the greedy set. If the test performs $\hat{f}_k > f(G_k)$, then the procedure scans all the elements of $\hat{S}_k$ until it encounters z whose ejection has the minimal negative impact on f. The current solution and its value are updated in line 9 and 10, and the flag is set to .TRUE. in line 11, indicating that a two-step swap was able to deliver a better solution than the greedy set. Otherwise, the flag remains equal to .FALSE.. Finally, in line 12 the 2SPG1 ALGORITHM delivers the two-step improving set, its value, and the flag.

Both 1SPG1 and 2SPG1 call for computations of f in addition to those already made by the greedy procedure. Since the number of extra calls is of the order of $\mathcal{O}(n)$, both 1SPG1 and 2SPG1 run in polynomial time.

Two questions are of relevance at this point of our discussion. How do S_k^* and $\hat{S}_k$ rank in terms of the objective function f? Are there sufficient conditions that guarantee (or exclude) that any these two sets will succeed at outperforming the greedy collection? In other words, can we be sure that the greedy set is optimum within its local neighborhood? The answer to these questions is introduced by the following lemma and it is articulated in the subsequent theorem.

Lemma 3 *With reference to Problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, it holds that*

$$f(G_k) \leq f(\hat{S}_k) \leq f(S_k^*) \leq f(O).$$

If f *is a modular polymatroid, then the above inequalities hold as equality.*

5.2.1 CCTV Example

In the context of our CCTV example, the neighborhood of 'size' one of the greedy set $G_3 = \{C_1, C_3, C_8\}$ is given by:

$$\mathcal{B}^1_{G_3} = \{S \subseteq V, |S| = 3 : S = ((G_3 + C_i) - C_j), \text{ with } C_i, C_j \in V\},$$

which is a set of $5 \times 4 = 20$ collections of three cameras each. In practice, excluding the non-enhancing collections generated by $(G_k + C_i) - C_i$, the 1SPG1 ALGORITHM requires to make $20 - 5 = 15$ queries to the value oracle.

Table 4: The elements of $\mathcal{B}^1_{G_3}$ [CCTV example]

	$S - C_1$	$S - C_3$	$S - C_8$
$S = G_3 + C_2$	63	60	69
$S = G_3 + C_4$	47	60	54
$S = G_3 + C_5$	45	58	52
$S = G_3 + C_6$	49	56	56
$S = G_3 + C_7$	63	55	64

Table 4 indicates that no swap of size one is able to improve upon the greedy set. This implies that that G_3 is the locally maximal element of the neighborhood $\mathcal{B}^1_{G_3}$ and that any hope we may have to improve upon $f(G_3)$ rests on a more forceful replacement strategy (i.e. on a swap of size greater than one). Finally, we underline that because $f(G_3) \leq f(\hat{S}_3) \leq f(S^*_3)$ (see Lemma 3), running the 2SPG1 ALGORITHM would also be unfruitful.
Our simple example clearly reveals that for a given instance of the Problem (3), there is no guarantee that a feasible collection $S \in \mathcal{B}^1_{G_k}$ can be produced to perform $f(S) > f(G_k)$. Nevertheless, in the continuation of our work we will show that in the particular case of a polymatroid function, the 'law' of non-increasing discrete derivatives can be exploited to produce a test that can establish when such an improvement is impossible to take place. □

Given that S^*_k 'dominates' every member of the local neighborhood of G_k, we reformulate the main question of our work in the following and more precise way: under what conditions can we be certain that the greedy set is locally maximal (i.e. that no collection $S \in \mathcal{B}^1_{G_k}$ can improve upon G_k)? The answer is provided by the following theorem:

Theorem 4 ***(Test on the local optimality of the greedy set)***

Let $f : 2^V \to \mathbb{R}_+$ be a polymatroid and G_k be the greedy set for Problem (3)*. If*

$$f(z) \geq \frac{f(g_{k+1}|G_k)}{1-\gamma_{k,1}} = \frac{f(G_{k+1}) - f(G_k)}{1-\gamma_{k,1}} \tag{8}$$

for all $z \in G_{k+1}$, then there is no $S \in \mathcal{B}^1_{G_k}$ that performs $f(G_k) < f(S)$.

Proof We prove the contrapositive statement. In order for $f(G_k) < f(S)$, with $S \in \mathcal{B}^1_{G_k}$, an element $z' \in G_{k+1}$ is required to exists to satisfy that

$$\underbrace{f(G_{k+1}) - f(G_{k+1} - z')}_{\text{marginal loss from swap}} < \underbrace{f(G_{k+1}) - f(G_k)}_{\text{marginal gain from swap}}$$

or, which is the same, that

$$1 - \frac{f(g_{k+1}|G_k)}{f(z')} < 1 - \frac{f(z'|G_{k+1} - z')}{f(z')}.$$

Now, given that for all $z \in G_{k+1}$ it holds through that

$$1 - \frac{f(z|G_{k+1} - z)}{f(z)} \leq \gamma_{k,1},$$

in order for $f(G_k) < f(S^+_k)$ to hold it is necessary that

$$f(z') < \frac{f(g_{k+1}|G_k)}{1-\gamma_{k,1}}.$$

The last result, in conjunction with Lemma 3, implies that $f(G_k) < f(S^+_k) < f(S^*_k)$, and this completes our proof. ■

To simplify our notation, in the continuation of our work we will set:

$$\Gamma_{k,1} =: \frac{f(g_{k+1}|G_k)}{1-\gamma_{k,1}},$$

a ratio that we refer to as *post-greedy index of size one* for the local search around G_k.

The index $\Gamma_{k,1}$, along with the necessary conditions offered by Theorem 4, exhibits a number of interesting features:

1. *Test of local optimality.* Knowing that all elements of V comply with the condition expressed by Theorem 4 implies that no collection in the neighborhood of G_k exists to outperform the greedy set. In practice, the condition (8) should be interpreted as a test of the local optimality of G_k within $\mathcal{B}^1_{G_k}$.

2. *Tradeoff.* As a ratio, the post-greedy index synthesizes the tradeoff characterizing our local search. One one hand, its numerator reflects a greedy improvement while, on the other hand, its denominator minimizes the subsequent loss from ejecting the element of G_{k+1} where f performs its highest, set/based, degree of concavity.

3. *Speedups.* The time complexity needed to verify the existence of an element of the ground set that complies with (8) is $\mathcal{O}(1)$. This is because, the data structure required to carry on the first step of the ADAPTIVE GREEDY algorithm already produced a sorted array of $f(u)$, for all $u \in V$, that can be used for this purpose. In addition, the total number of queries to the value oracle required by the calculation of $\Gamma_{k,1}$ is $(n-k)+(k+1) = n+1$ of which, $(n-k)$ are necessary to identify G_{k+1}, while $(k+1)$ are needed to evaluate $\gamma_{k,1}$. This implies that running the test (8) instead of evaluating all the elements of $\mathcal{B}^1_{G_k}$ allows for a speedup of the order of $\left[1-\frac{(n+1)}{k(n-k-1)}\right]$.

4. *Post-greedy curvature.* As $\gamma_{k,1}$ decreases, the likelihood of finding an element of the ground set that complies with Theorem 4 becomes tighter. In particular, if f is modular then $\gamma_{k,1} = 0$, and expression (8) simplifies to $f(u) < f(g_{k+1}) < f(g_1)$ which, due to Lemma 1, is trivially satisfied by any $u \in V$ that does not have the same value of g_1.

5. *Cardinality of the ground set.* The post-greedy index is independent on the cardinality of V. This implies that admitting new elements to the ground collection V would not affect the lower bound for $f(z)$. Yet, we might still be able to improve the outcome of our local search because enlarging V expands the number of new candidates that will be exposed to the inequality (8).

5.2.2 CCTV example

According to our example, we see that:

$$\Gamma_{3,1} = \frac{f(g_4|G_3)}{1-\gamma_{3,1}} = \frac{f(G_4)-f(G_3)}{1-\gamma_{3,1}} = \frac{78-70}{1-0.\overline{703}} = 27.$$

Now, given that $G_4 = \{C_1, C_2, C_3, C_8\}$, then condition (8) guarantees that G_3 is locally optimal within its swapping neighborhood $\mathcal{B}^1_{G_3}$. In other words, there is no swap of size one that can improve the value of the greedy collection. The calculation of $\Gamma_{k,1}$ requires us to make only $8 + 1 = 9$ queries to the value oracle, while the extensive evaluation of all the entries of Table 4 requires 12 calls. In practice, using (8) saves us $\frac{12-9}{12} = \frac{1}{4}$ of computation time. □

6. Improving strategies and more elaborate swaps

If our objective is to outperform the greedy collection G_k, there is no reason to restrict our attention to swaps of size one? Could we do better by producing a swap of greater size? in other words, could such a strategy succeed where a swap of a smaller size proved to fail? Does this produce (sufficient) conditions that ensures the local optimality of a greedy set $f(G_k)$? Does this gain of information outweigh the increased time complexity of the associated algorithm?

To escape the initial solution of an optimization problem by bringing the attention to unexplored feasible sets, we now consider swaps of progressively larger sizes, that is of size up to any $L \geq 1$, a number that is known *a priori*. This can be done by extending our local search around G_k to larger neighborhoods $\mathcal{B}^2_{G_k}$, ..., $\mathcal{B}^L_{G_k}$. In particular, at the level j of the swapping sequence, and in presence of a current solution S_{j-1}, a new solution S_j will be produced by first incorporating an element $v \in (V - S_{j-1})$. Since the obtained collection $(S_{j-1} + v)$ certainly violates the cardinality constraint, it cannot constitute a feasible solution of the original Problem (3). Therefore, we then convert it into a feasible collection by using a reversal move, consisting of expelling from $(S_{j-1} + v)$ as many elements as is needed to recreate feasibility. Ultimately, this generates an iterative sequence of feasible solutions, each of which is then transformed into its succeeding solution. We call this procedure Two-step post-greedy algorithm of size j^* (2sgij^*).

Formally speaking, let G_k be the available solution at the step 0 of the iterative sequence. We label with λ_j and ϵ_j the *injection* and the *ejection*, respectively, at the level j of the swapping sequence. Our methodology consists of building the sequence $\lambda_1, \epsilon_1, \ldots \lambda_j, \epsilon_j, \ldots, \lambda_L, \epsilon_L$, such that the transition that departs from G_k is obtained by performing $\lambda_1, \lambda_2, \ldots, \lambda_{j^*}, \epsilon_{j^*}$, where $j^* \leq L$ represents the level associated with the greatest quality solution that was visited during the sequence.

We acknowledge that the ejection λ_j requires us to add a new 'best' element g_{k+j} to the available set G_{k+j-1}, and then to eliminate as many elements as needed to achieve the feasible set with the greatest image. Thus:

$$\hat{u}_j = g_{k+j} = \underset{u \in V - G_{k+j-1}}{\operatorname{argmax}} f(u|G_{k+j-1}) \longmapsto f(\hat{u}_j|G_{k+j-1}) = f(G_{k+j})$$

and then, to recreate feasibility:

$$\hat{Z}_j = \underset{Z_j \subseteq G_{k+j}}{\operatorname{argmin}} f\left(G_{k+j} - Z_j\right) \longmapsto f(G_{k+j} - \hat{Z}_j).$$

Therefore, a swap of size j^* is implemented, where:

$$j^* = \underset{j \leq L}{\operatorname{argmax}} f(G_{k+j} - \hat{Z}_j).$$

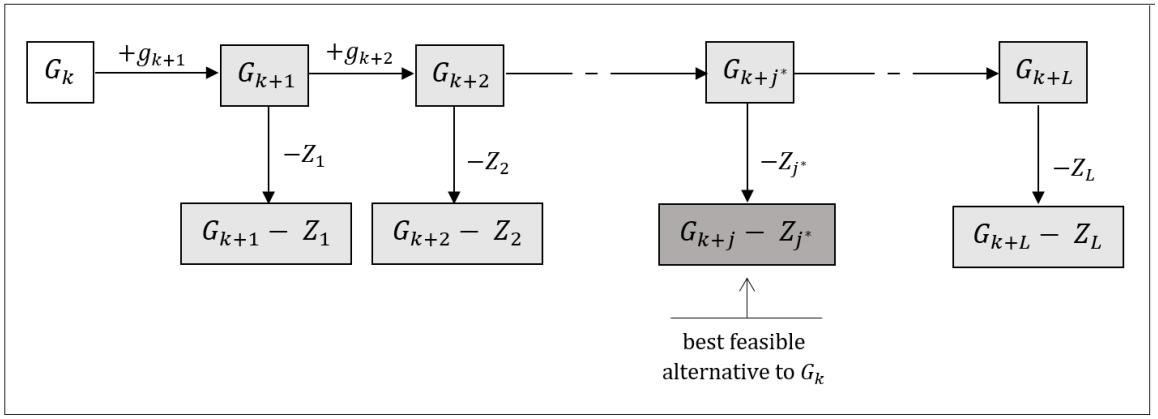
Algorithm 4: Two-step post-greedy algorithm of size j^* (2sgij^*)

Input: $f : 2^V \to \mathbb{R}_+$, G_k, $f(G_k)$, and the value query oracle for f.
Output: $S_L \subseteq N(G_k)$ and f_L (the greatest L-step greedy improvement w.r.t. G_k).

1 $S_L \leftarrow G_k$;
2 $f_L \leftarrow f(G_k)$;
3 $imp \leftarrow$ .FALSE.;
4 $count \leftarrow 0$;
5 **for** $j = 1, \dots, L$ **do**
6 $\quad g^* = \underset{u \in (V - S_L)}{\operatorname{argmax}} f(u|S_L)$ (choose arbitrarily in case of ties)
7 $\quad$ **if** $f(S_L + g^*) > f(S_L)$ **then**
8 $\quad\quad S_L \leftarrow S_L + g^*$;
9 $\quad\quad Z_L = \underset{Z \subseteq S_L, |Z| = j}{\operatorname{argmax}} f(S_L - Z)$ (choose arbitrarily in case of ties);
10 $\quad\quad S_L \leftarrow S_L - Z_L$;
11 $\quad\quad f_L \leftarrow f(S_L)$;
12 $\quad\quad imp \leftarrow$ .TRUE.;
13 $\quad\quad count \leftarrow j$;
14 **return** S_L, f_L, *imp*, *count*.

A graphical description of this procedure is sketched in Figure 3. In what follows we also present the pseudo-code for the corresponding heuristic.

Figure 3: The scheme of the two-step post-greedy algorithm of size j^*



The 2sgi$j$ procedure initializes the solution and its image in line 1 and 2, respectively, while in line 3 a flag is set to .FALSE. and in line 4 we introduce a counter that measures the number of elements that will be swapped between the greedy set and the final set S_L. The **for** loop takes place in lines 6 through 13, where the heuristic first scans the neighborhood of the current temporary solution S_L to identify its neighbor with the greatest marginal

improvement (line 5). Then, an **if** sub-loop (lines 8 through 13) takes over to recreate feasibility by ejecting from the temporary solution as many elements as needed to achieve feasibility with the minimum marginal impact on f. The current solution and its value are updated in lines 10 and 11, respectively. In case of successful search the flag is set to .TRUE. in line 12, indicating that a swap of size j was able to improve the image of the initial greedy set. The counter is then assigned value j in line 13. Finally, in line 14 the 2SGIj^* ALGORITHM delivers the two-step improving set, its value, the flag and the value taken by the counter.

Before digging into the discussion of what (sufficient) conditions would ensure an unsuccessful delivery by 2SGIj^* ALGORITHM, we are now in condition to provide an updated version of Lemma 3 and of the correlated Theorem 4.

Lemma 3' *Given Problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, it holds that*

$$f(G_k) \leq f(\hat{S}_k) \leq f(S_k^*) \leq f(S_L) \leq f(O).$$

If f *is a modular polymatroid, then the above inequalities hold as equality.*

The key question of our work remains the same: can we produce sufficient conditions that guarantee that no collection of $(G_{k+j} - \hat{Z}_j)$ can be found to improve upon the greedy set G_k? The answer is provided by the following theorem:

Theorem 4' (*General test on the local optimality of the greedy set* G_k)

Let $f : 2^V \to \mathbb{R}_+$ *be a polymatroid and* G_k *be the greedy set for Problem* (3)*. If*

$$f(Z_j) \geq \frac{f(g_{k+j}|G_k)}{(1-\gamma_{k,1})\,(1-\gamma_{k,2})\ldots(1-\gamma_{k,j})} = \frac{f(G_{k+j}) - f(G_k)}{\prod_{i=1}^{j}(1-\gamma_{k,i})} \tag{9}$$

for all $Z_j = \{z_1, \ldots, z_j\} \subseteq G_{k+j}$ *and all* $j \leq L$*, then there is no collection* $S_k^L \in \mathcal{B}_{G_k}^L$ *for which* $f(G_k) < f(S_k^L)$*.*

Proof Again, we prove by contraposition. In order to achieve $f(G_{k+j} - Z_j) > f(G_k)$, a collection $Z'_j = \{z'_1, \ldots, z'_j\} \subseteq G_{k+j}$ is required to exist so that:

$$\underbrace{f(G_{k+j}) - f(G_{k+j} - Z'_j)}_{\text{marginal loss from swap}} < \underbrace{f(G_{k+j}) - f(G_k)}_{\text{marginal gain from swap}}.$$

We use a telescopic transformation and take advantage of the normality and monotonicity of f to rewrite the initial inequality as:

$$\frac{1}{f(Z'_j)}\left[\prod_{i=1}^{j}\frac{f(G_{k+i}) - f(G_{k+i} - Z'_i)}{f(G_{k+i-1}) - f(G_{k+i-1} - Z'_{i-1})}\right] < \frac{f(G_{k+j}) - f(G_k)}{f(Z'_j)},$$

where $f(Z'_0) := 1$. In light of the submodularity of f, the last inequality can be expressed as

$$1 - \frac{f(G_{k+j}) - f(G_k)}{f(Z'_j)} < 1 - \prod_{i=1}^{j} \frac{f(G_{k+j}) - f(G_{k+j} - Z'_j)}{f(Z'_j)},$$

for some ordered sequence of sets $Z'_1 \subseteq \dots Z'_j \subseteq G_{k+j},\ j \leq L$. Finally, and after considering the maximal nature of $\gamma_{k,j}$, the last expression yields:

$$1 - \frac{f(G_{k+j}) - f(G_k)}{f(Z'_j)} < 1 - \prod_{i=1}^{j} (1 - \gamma_{k,i})\,.$$

Thus, for $f(G_{k+j} - Z'_j) > f(G_k)$ it is necessary that

$$f(Z'_j) < \frac{f(G_{k+j}) - f(G_k)}{\prod_{i=1}^{j} (1 - \gamma_{k,i})},$$

for some $Z'_j \subseteq G_{k+j}$. ■

We underline the fact that when $j = 1$, expression (9) admits (8) as a special case. To simplify our notation, we set:

$$\Gamma_{k,j} =: \frac{f(g_{k+j}|G_k)}{\prod_{i=1}^{j} (1 - \gamma_{k,i})} = \frac{f(G_{k+j}) - f(G_k)}{\prod_{i=1}^{j} (1 - \gamma_{k,i})},$$

a ratio that we define *post-greedy index of size* j for the local search around G_k.

Lemma 5 *For Problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, the post-greedy index of size* j *exhibits the following properties:*

i. $\Gamma_{k,j}$ *is positive;*

ii. $\Gamma_{k,j}$ *increases with* j*;*

iii. *if* f *is modular, then* $\Gamma_{k,j} = \sum_{i=1}^{j} f(g_{k+j})$ *for all* $j \geq 1$ *and* $k < n$*.*

We complete our discussion by looking again at the expression (9). As expected, our general test verifies that as the size j of a swap increases, the capability of the greedy set to preserve its local status as a maximum element deteriorates, while improves the probability of a new, feasible, and enhancing solution to become available via swap.

6.1 Further reducing the time complexity

Since the implementation of the theorem may impact the time complexity of our local search, the next corollary exploits the property of non-increasing returns of our polymatroid to introduce a useful shortcut that enhances time tractability.

Corollary 6 *Given a polymatroid $f : 2^V \to \mathbb{R}_+$ and the greedy set G_k, in order for $f(G_k) < f(G_{k+j} - \hat{Z}_j)$, it suffices that:*

$$\sum_{i=1}^{j} f(z_i) < \frac{f(G_{k+j}) - f(G_k)}{\prod_{i=1}^{j} (1 - \gamma_{k,i})}, \tag{10}$$

for some $(z_1, \ldots, z_j) \subseteq G_{k+j}$ and $j \leq L$.

Notice that the data structure required to implement the ADAPTIVE GREEDY algorithm already produced a sorted array of $f(u)$, for all $u \in V$. Therefore, this shortcut may further reduce the number of queries to the computational oracle by the number of permutations of n objects taken j at a time, that is ${}^nP_k = \frac{n!}{(n-j)!}$ (worst-case).

6.1.1 CCTV example

In the previous paragraphs we have seen that a swap of size one is unable to improve upon the greedy set. By making use of the necessary conditions expressed by (9), we now test whether the same should be expected of a replacement of size two.

Table 5: The candidates for $\gamma_{3,2}$ (numbers above the main diagonal) [CCTV example]

	$f(C_i, C_j)$	C_1	C_2	C_3	C_7	C_8
$f(C_i, C_j)$	C_1	–	0.810	0.654	0.514	**1**
	C_2	42	–	0.463	0.784	0.822
	C_3	52	54	–	0.611	0.600
	C_7	37	54	54	–	0.733
	C_8	52	45	45	45	–

After considering the 10 different combinations of three cameras that can be obtained from G_5, we produce the data presented in Table 5. Below the main diagonal we present the image of each $\{C_i, C_j\}$ while, above the main diagonal, we list the candidate values for $\gamma_{3,2}$ for each pair (C_i, C_j). Because $\gamma_{3,2} = 1$,it holds that

$$\Gamma_{3,2} = \frac{f(G_5) - f(G_3)}{(1 - \gamma_{3,1})(1 - \gamma_{3,2})} = \infty.$$

In practice, since $f(C_i, C_j) < \infty$ for each pair $(C_i, C_j) \subseteq G_5$, we are no longer guaranteed of the local optimality of G_3 within $\mathcal{B}^2_{G_3}$. For this reason, we are now open to the possibility that a swap of two cameras might succeed at improving the performance of G_3. A close inspection of the elements of G_3 and of $(V - G_3)$ reveals that swapping cameras (C_1, C_8) with (C_2, C_7) not only improves upon G_3 but also covers all the 81 vehicles of the parking garage. □

7. Conclusions

The main objective of this paper is to challenge the local optimality of the greedy output for the maximization of a polymatroid subject to a cardinality constraint. We did so by

testing its local optimality in the contest of its swapping neighborhoods.
We exploited the post-greedy curvature of the polymatroid to build a simple procedure that, in polynomial time, tells us whether we are guaranteed that the greedy collection is maximal within its swapping neighborhood. In case the test is verified, no element of its swapping neighborhood will be able to outperform its value. Since the greedy algorithm might be costly when applied to large-size instances, running this procedure may lead to significant savings of computation time. To the best of our knowledge, no polynomial-time procedure has been designed in literature to establish the presence (or the lack) of local optimality of the greedy algorithm in the particular context of the constrained maximization of a polymatroid.
We complete our paper by providing a brief list of future lines of investigation that surround the ideas we presented in our work. We ideally distinguish between 'greedy extensions' and 'beyond-greedy extensions'. In the first group we include some modifications to the standard AD procedure with the objective of enhancing its efficiency. In the second group we comprise a number of procedures that are designed to either outperform the greedy heuristic or to help us escape its local trap.

a) Greedy extensions

- *Randomization.* In this paper we use the greedy output as the initial collection for a subsequent swapping phase. Since the AD algorithm might be too time consuming in particular instances of Problem (3), a number of faster algorithms have been proposed as an alternative (Mirzasoleiman et al. (2015); Hashemi et al. (2018)), the most popular of which is the stochastic greedy procedure (SG). In its i-th iteration the SG randomly samples $(-n/k \ln \epsilon)$ elements of $\left(V - G^r_{i-1}\right)$, with $\epsilon \in \left(e^{-k}, 1\right)$, and where G^r_{i-1} denotes the random greedy collection from the previous step. Then, the algorithm chooses the element that provides the greatest marginal gain out of those that were sampled. This heuristics makes $(-n \ln \epsilon)$ calls to the evaluation oracle and achieves a $(1 - 1/e - \epsilon)$-approximation guarantee (in expected terms). The adoption of a SG heuristic would only have a marginal impact on the formalities of our work and every conclusion reached in our work will be easily adapted and remain valid.

- *Addressing ties.* The greedy procedure that generates the collection G_k often generates ties that, in our paper, we addressed by assuming a random choice. In any heuristic, the adopted tie-breaking mechanism is known to have a certain influence in its output. In the particular case of the problem that we investigate in our paper, such an impact extends to the likelihood of falling in a greedy trap. To the best of our knowledge, no particular tie-breaking mechanism has ever been been proposed for the AG algorithm in the context of a constrained maximization of a polymatroid. An in-depth analysis of more elaborate tie-breaking schemes may represent a fruitful line of future investigation.

b) Beyond-greedy extensions

- *Iterated swaps and ejection chains.* The careful reader may find that one of our work's main rigidities is the lack of iteration in the swapping process. Indeed, once

the greedy set is replaced by a 'better' set S_L, no more swapping takes place. Yet, there is no reason for not searching for additional improvements starting from S_L via a new iteration. A known procedure for doing so is called ejection chain algorithm (EC), which was presented by Glover (1991, 1992) and further developed by Rego and Glover (2010), mainly in the context of the TSP algorithm. The literature on this topic is relatively new and, at the moment, mainly characterized by problem-specific results. One of the main pitfalls of ejection chains is that, as the iteration brings us away from the greedy set, we are no longer able to exploit its formal characteristics. Therefore, as new iterations take place, a unified and consistent formal treatment of the subject becomes progressively difficult to maintain.

- *Variable neighborhoods.* Both the 1spg1 and the 2spg1 local search heuristics are known to easily become stuck in areas of the ground set where scores plateau. We could move away from G_k, by arbitrarily building a new feasible solution S_{k+1} to be exposed to swaps of size one by implementing the so-called Variable Neighborhood Search (VNS) meta-heuristic. This procedure systematically exploits the idea of producing controlled and pre-defined neighborhood changes, both in escaping from and in ascending to a local maximum (Hansen and Mladenović, 2003). According to VNS, if a 'best' neighboring element dominates the greedy set, then we use it to replace it. Vice versa, i.e. in case G_k remains locally maximal, we undergo a local search within progressively 'larger' neighborhood structures $\mathcal{B}^j_{S_{k+1}}$, with $j > 1$, until we find a feasible set that outperforms the greedy outcome. Then, we iteratively repeat the same steps with the new set. Given that the VNS process represents an efficient way to escape the greedy trap, it is a procedure that deserves further exploration.

- *Tabu search.* Facing the risk of being trapped in a greedy suboptimal solution, we may want to consider regions of the search space that are left unexplored by standard local search procedures. tabu search (TS) is not an algorithm but a complex procedure that builds around the step-bay-step creation of the greedy set G_k, by relaxing some of its standard rules (Glover and Laguna, 1997). First, at each step of its iteration process, the heuristic is open to admit non-improving feasible collections when an improving move is not available. In addition, a prohibition (hence 'taboo') is introduced to block the returning to any solution that was previously explored. By making an intensive use of memory structures, this routine uses a neighborhood dynamic search procedure to move from a potential to an improved solution, until some stopping criterion is met (Glover, 1989, 1990a,b). The TS procedure may represent a valid alternative to VNS and should be considered of great relevance for future lines pf work.

- *GRASP.* The greedy randomized adaptive search procedure (GRASP) was introduced by Feo and Resende (1995) and systematized by Resende and Ribeiro (2010, 2016). GRASP as an iterative procedure for producing new and feasible solutions starting from an initial greedy collection which consists of two phases: a *construction phase* and a *local search phase*. In each iteration, the best overall solution is temporarily stored and becomes ready to be challenged by a new search. The stochastic nature of GRASP appears in the construction phase, where a new feasible

solution is iteratively constructed via a greedy approach by randomly choosing one element at a time and from a list of best candidates. Then, the local search phase improves the constructed solution in an iterative fashion by successively replacing the incumbent solution with a locally improving collection. We also recommend and in-depth investigation of GRASP in the context of polymatroid maximization subject to a matroid constraint.

Acknowledgments

I would like to acknowledge patience and support for this work from my family and friends, and especially from my husband Patric J. Walton. I also show to Patric my gratitude for his valuable editorial and style comments.

Appendix. Proof of all the lemmas and corollaries

We here provide a proof of the lemmas and corollaries that we presented in our work.

Lemma 1. *Let Problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$. *Given the sequence* $G_0 \subseteq \ldots \subseteq G_i \subseteq \ldots \subseteq G_n$, *with* $G_0 := \emptyset$, *the greedy marginal increment* $f(g_{i+1}|G_i)$ *is non-increasing with* i.

Proof To show that $f(g_{i+1}|G_i)$ is non-increasing we build the following sequence for $i = 0, 1, \ldots, n-1$:

$$f(g_i|G_{i-1}) \geq f(g_{i+1}|G_{i-1}) \geq f(g_{i+1}|G_i),$$

where the first inequality relies on the fact that $g_i = \underset{z \in (V - G_{i-1})}{\operatorname{argmax}} f(z|G_{i-1})$, and the second inequality exploits the definition of submodularity expressed by (1). ■

Lemma 2. *For problem* (3) *with a polymatroid* $f : 2^V \to \mathbb{R}_+$, *the post-greedy curvature owns the following properties:*

i. $\gamma_{k,j}$ *takes its value in the interval* $[0, 1]$;

ii. *if* f *is modular, then* $\gamma_{k,j} = 0$ *for all* $j \geq 1$ *and* $k < n$;

iii. $\gamma_{k,j}$ *is non-increasing with* j;

iv. *as* k *tends to* $(n-1)$, $\gamma_{k,1}$ *approaches* α.

Proof Let Z'_j and Z''_{j-1} denote, respectively, the vectors that solve the implicit maximization problem in the definition of $\gamma_{k,j}$ and $\gamma_{k,j-1}$ (see expression (7)).

i. To show that $0 \leq \gamma_{k,j}$, we take advantage of the definition of submodularity that is expressed by (2) and write:

$$\gamma_{k,j} = 1 - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_j)}{f(Z'_j)} \geq 1 - \frac{f(G_{k+j}) - f(G_{k+j}) + f(Z'_j)}{f(Z'_j)} = 0.$$

Instead, to prove that $\gamma_{k,j} \leq 1$, we invoke the monotonicity of f to acknowledge that

$$\frac{f(G_{k,j}) - f(G_{k,j} - Z'_j)}{f(Z'_j)} \geq 0.$$

ii. If f is modular, then

$$\gamma_{k,j} = 1 - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_j)}{f(Z'_j)} = 1 - \frac{f(Z'_j)}{f(Z'_j)} = 0.$$

iii. To show that the post-greedy curvature is non-increasing with j, we manipulate the difference $\gamma_{k,j} - \gamma_{k,j-1}$ along the following lines:

$$\gamma_{k,j} - \gamma_{k,j-1} = \frac{f(G_{k,j-1}) - f(G_{k,j-1} - Z''_{j-1})}{f(Z''_{j-1})} - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_j)}{f(Z'_j)} \leq$$

$$\leq \frac{f(G_{k,j-1}) - f(G_{k,j-1} - Z'_{j-1})}{f(Z'_{j-1})} - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_{j})}{f(Z'_{j})} \leq$$

$$\leq \frac{f(G_{k,j-1}) - f(G_{k,j-1} - Z'_{j-1})}{f(Z'_{j-1})} - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_{j-1})}{f(Z'_{j-1})} \leq 0.$$

where the first inequality is obtained by replacing Z'_{j-1} with Z''_{j-1} and then realizing that the subtrahend no longer takes its minimum value, while the inequality with zero is consistent with the definition of submodularity expressed by (1). In particular, the last step establishes that $\gamma_{k,j} \leq \gamma_{k,j-1}$ and proves that the post-greedy curvature does not increase with j.

iv. First, we realize that if $k = (n-1)$, then $G_{k+1} = G_n = V$. Then, we calculate the following limit:

$$\lim_{k \to (n-1)} \gamma_{k,1} = 1 - \min_{z \in V} \frac{f(V) - f(V-z)}{f(z)},$$

which is the definition of the total curvature α.

■

Lemma 3. *With reference to problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, it holds that*

$$f(G_k) \leq f(\hat{S}_k) \leq f(S^*_k) \leq f(O).$$

If f *is a modular polymatroid, then the above inequalities hold as equality.*

Proof We proceed by comparing pairs of sets:

1. $f(G_k) \leq f(\hat{S}_k)$. Since $\hat{S}_k = (G_k + \hat{u}) - \hat{z}$ admits $\hat{u} = \hat{z}$ as a special case, the image of $\hat{S}_k$ cannot be smaller than $f(G_k)$.

2. $f(\hat{S}_k) \leq f(S^*_k)$. The locally optimum member of $\mathcal{B}^1(G_k)$ outperforms any other feasible subset of the same neighborhood, $\hat{S}_k$ included.

3. $f(S^*_k) \leq f(O)$. The image of the global optimum collection, that is O, is non-smaller than the image of any other feasible set, including S^*_k.

As displayed by (6), when f is modular then $f(G_k) = f(\hat{S}_k) = f(S^*_k) = f(O)$. ■

Lemma 3' *Given problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, it holds that*

$$f(G_k) \leq f(\hat{S}_k) \leq f(S^*_k) \leq f(S_L) \leq f(O).$$

If f *is a modular polymatroid, then the above inequalities hold as equality.*

Proof It suffices to show that $f(S_k^*) \leq f(S_L)$. Since $S_L = (G_k + U_{j^*} - Z_{j^*})$ admits $U_{j^*} = g_{k+1}$ and $Z_{j^*} = \hat{z}$ as a special case for $j^* = 1$, then the image of $\hat{S}_k$ cannot be greater than $f(S_L)$. ■

Lemma 5. *For problem* (3) *and a polymatroid* $f : 2^V \to \mathbb{R}_+$*, the post-greedy index of size* j *exhibits the following properties:*

i. $\Gamma_{k,j}$ *is positive;*

ii. $\Gamma_{k,j}$ *increases with* j*;*

iii. *if* f *is modular, then* $\Gamma_{k,j} = \sum_{i=1}^{j} f(g_{k+j})$ *for all* $j \geq 1$ *and* $k < n$*.*

Proof

i. The fact that $\Gamma_{k,1} \geq 0$ is an immediate consequence of the monotonicity of f.

ii. To study how $\Gamma_{k,j}$ responds to a change in j, we recognize that a raise in j both increases the numerator of

$$\frac{f(G_{k+j}) - f(G_k)}{\prod_{i=1}^{j}(1 - \gamma_{k,i})},$$

and decreases its denominator.

iii. Finally, notice that if f is modular, then both $f(G_{k+j}) - f(G_k) = \sum_{i=1}^{j} f(g_{k+j})$ and $\gamma_{k,j} = 0$ (see Lemma 2) for $j \geq 1$.

■

Corollary 6. *Given a polymatroid* $f : 2^V \to \mathbb{R}_+$ *and the greedy set* G_k*, in order for* $f(G_k) < f(G_{k+j} - \hat{Z}_j)$*, it suffices that:*

$$\sum_{i=1}^{j} f(z_i) < \frac{f(G_{k+j}) - f(G_k)}{\prod_{i=1}^{j}(1 - \gamma_{k,i})}, \tag{11}$$

for some $(z_1, \ldots, z_j) \subseteq G_{k+j}$ *and* $j \leq L$*.*

Proof The result is immediate after acknowledging that for a polymatroid f and any collection $Z_j \in V$, the following inequality is always satisfied:

$$f(Z_j) < \sum_{z \in Z_j} f(z).$$

■

References

Niv Buchbinder and Moran Feldman. *Submodular Functions Maximization Problems - A Survey*, volume 1 of *Handbook of Approximation Algorithms and Metaheuristics*, chapter 42. Ed. Teofilo F. Gonzalez, second edition, 2018.

Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *Proceedings of the 2014 Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1433–1452, 2014.

Gruia Calinescu, Chandra Chekur, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *International Conference on Integer Programming and Combinatorial Optimization*, volume 4513, IPCO 2007. Lecture Notes in Computer Science, june 2011. Fischetti M., Williamson D.P. (eds).

Chandra Chekuri and Amit Kumar. Maximum coverage problem with group budget constraints and applications. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 72–83. Springer, Berlin, Heidelberg, 2004.

Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado–edmonds theorem. *Discrete Applied Mathematics*, 7(3):251–274, 1984.

G. A. Croes. A method for solving traveling salesman problems. *Operation Research*, 6(6): 253–266, 1958.

Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the Association for Computing Machinery (ACM)*, 45(4):634–652, 1998.

Thomas A. Feo and Mauricio G. C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, (6):109–133, 1995.

Lisa Fleischer, Michel X. Goemans, Vahab S. Mirrokni, and Maxim Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 36, pages 611–620. Springer, Berlin, Heidelberg, 2006.

Merrill M. Flood. The traveling-salesman problem. *Operation Research*, 4:61–75, 1956.

Yigal Gerchak and Diwakar Gupta. On apportioning costs to customers in centralized continuous review inventory systems. *Journal of Operations Management*, 10(4):546–551, 1991.

Fred W. Glover. Tabu search – part 1. *ORSA Journal on Computing.*, 1(2):190–206, 1989.

Fred W. Glover. Tabu search – part 2. *ORSA Journal on Computing.*, 4(32):4–32, 1990a.

Fred W. Glover. Tabu search: A tutorial. *The Practice of Mathematical Programming*, 20 (4):74–94, 1990b.

Fred W. Glover. *Multilevel tabu search and embedded search neighborhoods for the traveling salesman problem.* Leeds School of Business. University of Colorado, CO, 1991.

Fred W. Glover. New ejection chain and alternating path methods for traveling salesman problems. *Computer Science and Operations Research*, 65:449–509, 1992.

Fred W. Glover and Manuel Laguna. *Tabu Search.* Springer Science - Business Media, LLC, 1997.

Pranava R. Goundan and Andreas S. Schulz. Revisiting the greedy approach to submodular set function maximization. *Optimization Online*, 2007. URL `http://www.optimization-online.org/DB_HTML/2007/08/1740.html`.

Pierre Hansen and Nenad Mladenović. *A Tutorial on Variable Neighborhood Search.* Hec Montreal and Gerad, 2003.

Bruce C. Hartman, Moshe Dror, and Moshe Shaked. Cores of inventory centralization games. *Games and Economic Behavior*, 31(1):26–49, 2000.

Abolfazl Hashemi, Mahsa Ghasemi, Haris Vikalo, and Ufuk Topcu. A randomized greedy algorithm for nearoptimal sensor scheduling in large-scale sensor networks. In *2018 Annual American Control Conference*, page 1027–1032, 2018.

Stefanie Jegelka and Jeff Bilmes. Submodularity beyond submodular energies: Coupling edges in graph cuts. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 1645–1656, 2011.

David Kempe, Jon Kleinberg, and Evan Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 137–146, 2003.

Pushmeet Kohli, M. Pawan Kumar, and Philip H. S. Torr. p^3 and beyond: move making algorithms for solving higher order functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1645–1656, 2009.

Andreas Krause and Carlos E. Guestrin. A near-optimal nonmyopic value of information in graphical models. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, volume UAI, page 5, 2005.

Andreas Krause and Carlos E. Guestrin. Near-optimal observation selection using submodular functions. In *Proceedings of the 22nd national conference on Artificial intelligence*, volume AAAI Press, page 1650–1654, 2007.

Andreas Krause, Ajit Singh, and Carlos E. Guestrin. Nearoptimal sensor placements in gaussian processes: theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, January(9):235–284, 2008.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 510–520. Association for Computational Linguistics, 2011.

Yajing Liu, Edwin K. P. Chong, Ali Pezeshki, and Zhenliang Zhang. Improved bounds for the greedy strategy in optimization problems with curvatures. *Journal of Combinatorial Optimization*, 37(4):1126–1149, 2019.

Yajing Liu, Edwin K. P. Chong, Ali Pezeshki, and Zhenliang Zhang. Submodular optimization problems and greedy strategies: A survey. *Discrete Event Dynamic Systems*, (30): 381–412, 2020.

Michel Minoux. Algorithmes gloutons et algorithmes glotons accélérés pour la résolution des grands problèmes combinatoires. *Bulletin de la Direction des Etudes et Recherches, EDF (France)*, Série C(1):41–58, 1977.

Michel Minoux. *Accelerated greedy algorithms for maximizing submodular set functions*, volume 7 of *Optimization Techniques*, chapter Lecture Notes in Control and Information Sciences. Stoer J. (eds), Springer, Berlin, Heidelberg, 1978.

Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrak, and Andreas Krause. Lazier than lazy greedy. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, page 1812–1818. AAAI Press., 2015.

Elchanan Mossel and Sébastien Roch. On the submodularity of influence in social networks. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of computing*, page 128–134, 2007.

George Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions - i. *Mathematical Programming*, 14(1): 265–294, 1978a.

George Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions - ii. *Polyhedral Combinatorics*, 14:73–87, 1978b.

James G. Oxley. *Matroid Theory*. Oxford Garduate Texts In Mathematrics. Oxford University Press, 2006.

Colorado Reed and Zoubin Ghahramani. Scaling the indian buffet process via submodular maximization. In *Proceedings of the 30 th International Conference on Machine Learning*, volume 28, pages 1013–1021, 2013.

César Rego and Fred W. Glover. Ejection chain and filter-and-fan methods in combinatorial optimization. *Ann Oper Res*, (175):77–105, 2010.

Mauricio G. C. Resende and Celso C. Ribeiro. Grasp: Greedy randomized adaptive search procedures. In *Search Methodologies. Introductory Tutorials in Optimization and Decision Support Systems*, pages 287–312. E. K. Burke and G. Kendall (Eds.), 2010.

Mauricio G. C. Resende and Celso C. Ribeiro. *Optimization by GRASP, Greedy Randomized Adaptive Search Procedures*. Springer, 2016.

Tim Roughgarden. *Algorithms Illuminated*, volume Part 4: Algorithms for NP-hard problems. SoundLikeYourself Publishing, LLC, 2020.

Andreas S. Schulz and Nelson A. Uhan. Approximating the least core value and least core of cooperative games with supermodular costs. *Discrete Optimization*, 10(2):163–180, 2013.

Chao Shen and Tao Li. Multi-document summarization via the minimum dominating set. In *Proceedings of the 23rd International Conference on Computational Linguistics*, page 984–992. Coling 2010 Organizing Committee, 2010.

Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32:41–43, 2004.

Jan Vondrák. Submodularity and curvature: the optimal algorithm. *RIMS Kkyuroku Bessatsu*, B(23):253–266, 2010.

Zengfu Wang, Bill Moran, Xuezhi Wang, and Quan Pan. Approximation for maximizing monotone non-decreasing set functions with a greedy method. *Journal of Combinatorial Optimization*, 31:29–43, 2014.

Hassler Whitney. On the abstract properties of linear dependence. *American Journal of Mathematics*, 57(3):509–533, 1935.