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

Mirco Soffritti
msoffritti@nebrija.es
School of Arts and Science
Department of Business and Economics
Antonio de Nebrija University
Madrid, 28015, Spain

Abstract
We study the problem of maximization of a polymatroid subject to a cardinality constraint. When the set of elements to be considered is large, the greedy algorithm is the most natural heuristic for producing an approximate solution. With the greedy set in hand, we may desire to improve its value by swapping one or more of its elements with a corresponding number of elements that do not belong to it. Even though there is no guarantee that such a swap will succeed at improving the greedy solution, we define a (set-based) post-greedy measure of curvature of a polymatroid and utilize it to design a non-recursive test that, in polynomial time, provides sufficient conditions for the greedy collection to be ‘locally’ optimal. We verify that as the number of swapped elements increases, the likelihood for the greedy collection to remain locally optimal deteriorates. In other words, the probability of locally producing a feasible and better solution enhances.

Keywords: polymatroid, discrete maximization, greedy algorithm, cardinality constraint, local search, swaps, and post-greedy curvature.

1. Introduction
Submodularity is a property of set functions that can be found everywhere. For example, suppose my family and I are planning to spend four days in Venice, Italy. After having planned to visit to two of Venice’s main churches, I wonder if I should visit to another church. How would I feel about that? I would feel fine about that because I like religious architecture. However, I would not feel as good as when I was planning to visit the church of San Giorgio Maggiore, my second-choice church, just a few minutes ago. And I expect to enjoy it even less if I were to compare this experience with that of visiting the San Marco cathedral, my number one choice. This is ‘submodularity’ in practice, the discrete analogue of concavity that we encounter in continuous calculus (Lovász, 1983). Submodularity makes the optimization of a functions a difficult task to accomplish, in particular if we want to optimize subject to some binding constraints. For example, I may have the time to visit a maximum of three churches, or I may have to surrender to the fact that my stay cannot extend beyond the four days I planned to spend in ‘La Serenissima’, etc. In many areas of application of mathematics, a number of problems have been naturally structured as subset...
selection problems within a submodular framework. For example, 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 our work we are specifically interested in the problem of submodular maximization subject to a cardinality constraint, that is conditional to a 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 becomes impossible, and the greedy search reveals to be the most natural procedure. In the specific context where we move in our work, it is a well established result that the greedy approach may produce an approximation error to the value of the unknown optimum that is at most \((1 - e^{-1})\) (Nemhauser et al., 1978a,b). Also, it has been proved that the greedy heuristic represents the best deterministic procedure operating in polynomial time that we can possibly hope for for our problem (Feige, 1998).

Even though the greedy set comes with no validation of its ‘global’ optimality, we still wonder whether or not it represents an optimal solution. Inspired by the 2-OPT ALGORITHM (Croes, 1958), in our work we challenge its ‘local’ optimality in the context of the specific neighborhood structure that we obtain by swapping some elements of the greedy output with an identical number of elements not belonging to it. The simplest replacement is a swap of size one, that is a replacement that ejects one ‘insider’ element of the greedy set and replaces it with one of its ‘outsider’ elements. After defining a new (set based) ‘con- cavity’ index, which we call post-greedy curvature, we will use it to design a simple test that establishes whether sufficient conditions are met for the greedy set to be maximal within its own neighborhood. Not only our test runs in polynomial time, but knowing that the greedy output is locally optimal will save us the time needed by undergoing a brute-force evaluation of the competing collections that belong to the same neighborhood. Finally, we generalize our base-case test by discussing the idea of a swap of size greater than (or equal to) one, where the number of expelled elements from the greedy set still coincides with number of external elements that are introduced to replace them. The general version of the post-greedy curvature of a polymatroid will remain the foundation stone of the associated heuristic, which we call TWO-STEP POST-GREEDY ALGORITHM of size \(j\). The 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 an enhancing local solution to become available.

Our work is organized as follows. We will structure Chapter 2 around the concept of polymatroid function and we define the problem of its maximization subject to a cardinality constraint. In the same chapter we formalize a concrete example to be carried on throughout the paper. In Chapter 3 we will present the ADAPTIVE GREEDY ALGORITHM (AG) and we will 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 will structure 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 remaining chapters, will be designed to output new feasible solutions that may (or may not) enhance the polymatroid with respect to the AG heuristic. In particular, by keeping the cardinality requirement in check, we will explore the idea of swapping some elements of the greedy set with an identical number of ground elements that do not belong to it. In Chapter 5 we will begin simple, that is with a replacement of one element. We will make use of the post-greedy curvature to design a 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 will put forward a number of possible extensions and interesting lines of investigation.

Throughout the paper, we will use a white square ‘□’ to indicate where the implementation of an idea in our recurrent example is complete. A black square ‘■’ will be used to denote when a formal demonstration is achieved. The proof of theorems is provided within the text, while that of 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) $V = \{v_1, v_2, \ldots, v_n\}$ 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), an efficient computational black-box which, given a set $S \subseteq V$, returns the value $f(S)$ by using a polynomial time. We denote with $\#(S)$ such number of queries.

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 submodular if:

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

This definition states that adding an element to a larger set results in a non-increasing discrete derivative of $f$ than adding the same element 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 \( S \subseteq T \subseteq V \), \( f(S) \leq f(T) \).

In particular, 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, \( f \) and is called **modular** function. In practice, a modular function is just a special kind of a submodular application.

### 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:

1. \( \emptyset \in \mathcal{I} \),
2. if \( S \subseteq T \) and \( T \in \mathcal{I} \), then \( S \in \mathcal{I} \),
3. 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 called an **independent set** 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 \( S' = \text{argmax}_{S \in \mathcal{I}} |S| \).

We define **rank** of the matroid, \( \text{rank}(V, \mathcal{I}) \), the cardinality of a (any) basis of the matroid. If we further assume that \( \mathcal{I} = \{ S \in 2^V : |S| \leq k < n \} \), 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 will 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 \( f \). That is, 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:

\[
O = \arg\max_{S \in \mathcal{I}} f(S).
\]
2.2.1 Example

To make sense of the main ideas discussed 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 installed in the parking garage. 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 $f : 2^V \rightarrow \mathbb{R}$ 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 submodularity of $f$ can be verified by acknowledging the 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 establish. 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 $[f(C_5) = 6] < [f(C_3) = 27] < [f(V) = 81]$. 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 as an 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 $(i-1)^{th}$ step, then $G_i = G_{i-1} + g_i$, where

$$g_i \in \left\{ \arg\max_{v \in (V-G_{i-1})} f(v|G_{i-1}) \right\}$$

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 $\#(G_k) = \sum_{j=0}^{k} (n-j) = O(nk)$ 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.

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 $i$. 


Algorithm 1: Adaptive Greedy Algorithm (AG)

**Input:** $f : 2^V \rightarrow \mathbb{R}_+, \ 1 \leq k \leq n$, 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. $g^* = \text{argmax}_{u \in (V - G_k)} f(u|G_k)$ (choose randomly in case of ties);
   5. $G_k \leftarrow G_k + g^*$;
   6. $f(G_k) \leftarrow f(G_k + g^*)$;
4. return $G_k, f(G_k)$.

Despite not carrying any insurance of correctness for Problem (3) (otherwise the $P \neq NP$ conjecture would be violated), 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 $(1 - e^{-1})$ 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 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:

1. 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.
2. 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 $\rho(u^*_i) \leftarrow f(u^*_i|G_{i-1})$.
3. 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.
4. 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 produced by both 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 $C_8$. In particular, 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 arbitrarily 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 $G_3 = \{C_1, C_3, C_8\}$ is presented as a 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 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.

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                | ×                 | ×                 |

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 an unfortunate and arbitrary choice was made in favor of $C_8$. Finally, notice that

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

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 algorithm. In the first iteration, all cameras are considered and an arbitrary 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 24%. □
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 \( \emptyset \subseteq \)
$G_1 \subseteq \ldots G_{k-1} \subseteq G_k$, they introduce the concept of greedy curvature which, for the case of a polymatroid, can be expressed as:

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

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

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

(5)

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

(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, recent literature on submodular maximization produced a number of additional notions that have been 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 will call post-greedy curvature of the polymatroid $f$:

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

(7)

where $Z_j = \{z_1, \ldots, z_j\} \subseteq G_{k+j}$, and $j = 1, 2, \ldots, n-k$. This index, whose number of calls to the evaluation oracle is of the order of $O(k + j)$, 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 established by the following lemma.

**Lemma 2** For Problem (3) with a polymatroid $f : 2^V \rightarrow \mathbb{R}_+$, 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 \( \alpha \).

In Chapters 5 and 6 we will make an extensive application of the index \( \gamma_{k,j} \) when introducing sufficient conditions for the successful 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 \( \alpha \) and \( \alpha_G \) 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 \( G_4 = \{C_1, C_2, C_3, C_8\} \).

<table>
<thead>
<tr>
<th>( C_i )</th>
<th>( f(C_i) )</th>
<th>( 1 - \left( \frac{f(C_i \setminus G_k)}{f(C_i)} \right) ), ( C_i \in V )</th>
<th>( 1 - \left( \frac{f(C_i \setminus G_k)}{f(C_i)} \right) ), ( C_i \in G_3 )</th>
<th>( 1 - \left( \frac{f(C_i \setminus G_k)}{f(C_i)} \right) ), ( C_i \in G_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_1 )</td>
<td>25</td>
<td>1</td>
<td>0</td>
<td>0.400</td>
</tr>
<tr>
<td>( C_2 )</td>
<td>27</td>
<td>0.888</td>
<td>-</td>
<td>0.703</td>
</tr>
<tr>
<td>( C_3 )</td>
<td>27</td>
<td>0.925</td>
<td>0.333</td>
<td>0.333</td>
</tr>
<tr>
<td>( C_4 )</td>
<td>8</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>( C_5 )</td>
<td>6</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>( C_6 )</td>
<td>25</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>( C_7 )</td>
<td>27</td>
<td>0.888</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>( C_8 )</td>
<td>27</td>
<td>1</td>
<td>0.333</td>
<td>0.666</td>
</tr>
</tbody>
</table>

In the context of our example, \( \alpha = 1 \), \( \alpha_G = 0.333 \) and \( \gamma_{k,1} = 0.703 \). 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 make \( O(k+1) = |G_{k+1}| = 4 \) calls to the evaluation oracle. □

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 closely follow Resende and Ribeiro (2016) to introduce a number of new concepts and technical terms. Starting from the \( G_k \), a local search consists of any strategy that explores its (feasible or infeasible) neighbors to find an improving solution whose cardinality is still \( k \), and without iteration. Such a neighborhood search may either yield no improvement 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) we are dealing
with. In our paper we only 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. The $G_k$’s neighborhoods of ‘size’ $j \geq 1$ is formally defined as:

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

Technically speaking, $B_j^G$ 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 $S' \in B_j^G$ whenever $G_k \in B_j^{G_k}$. In addition, since $G_k$ is also a member of $B_j^G$, then any $S' \in B_j^G$ 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 want to undergo a search in the neighborhood

$$B_1^{G_k} = \{ S \subseteq V, |S| = k : S = (G_k + u) - Z, \text{ with } u, Z \subseteq V \},$$

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 $B_1^{G_k}$, that is:

  $$S^*_k = \arg\max_{S \in B_1^{G_k}} f(S).$$

- **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:

  $$\hat{S}_k = [(G_k + \hat{u}) - \hat{z}] \in B_1^{G_k} \text{ such that } \hat{u} = \arg\max_{u \in V - G_k} f(u|G_k) \text{ and }$$

  $$\hat{z} = \arg\min_{z \in (G_k + \hat{u})} f(z|(G_k + \hat{u}) - z).$$

  Now, since

  $$g_{k+1} = \arg\max_{u \in (V - G_k)} f(u|G_k),$$
then,
\[ \hat{S}_k = (G_{k+1} - \hat{z}) \text{ with } \hat{z} = \arg\min_{z \in G_{k+1}} f(z|G_{k+1} - z). \]

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

**Algorithm 2: One-step post-greedy algorithm of size one (1spg1)**

**Input:** \(f : 2^V \rightarrow \mathbb{R}_+\), \(G_k\), \(f(G_k)\), \(\mathcal{B}^1_{G_k}\), 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. \(\text{imp} \leftarrow \text{.FALSE.};\)
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. \(\text{imp} \leftarrow \text{.TRUE.};\)
12. **return** \(S^*_k, f^*_k, \text{imp}.\)

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 \(\text{imp}\) 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. Since a ‘better’ neighbor of \(G_k\) may or may not exist, in lines 9 to 11 the **if** loop takes care of comparing (the value of) 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 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 \(\text{imp}\).

The 2spg1 ALGORITHM initializes the solution and its image in line 1 and 2, respectively, while in line 3 a flag is set to .FALSE.. In lines 4 and 5, the heuristic scans all neighborhood of \(G_k\) and replaces the current solution with its neighbor associated with the greatest marginal improvement (which may or may not exist). To recreate feasibility, the **if** loop in
Algorithm 3: Two-step post-greedy algorithm of size 1 (2spg1)

**Input:** \( f : 2^V \rightarrow \mathbb{R}_+, \ G_k, \ f(G_k), \) and the value query oracle for \( f. \)

**Output:** \( \hat{S}_k \subseteq B^1_{G_k} \) 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. \( \text{imp} \leftarrow \text{.FALSE.}; \)
4. \( \hat{u} = \arg\max_{u \in (V - \hat{S}_k)} f(u|\hat{S}_k) \) (choose arbitrarily 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. \( \hat{z} = \arg\max_{z \in \hat{S}_k} f(\hat{S}_k - z) \) (choose arbitrarily in case of ties);
   9. \( \hat{S}_k \leftarrow \hat{S}_k - \hat{z}; \)
   10. \( \hat{f}_k \leftarrow f(\hat{S}_k); \)
   11. \( \text{imp} \leftarrow \text{.TRUE.}; \)
12. return \( \hat{S}_k, \hat{f}_k, \text{imp}. \)

Lines 7 to 11 compares the value of the updated solution with that of the greedy set. If the test delivers \( \hat{f}_k > f(G_k) \), 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 initial 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 \( O(n) \), 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 \( f : 2^V \rightarrow \mathbb{R}_+, \) 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 an 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:

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

which is a collection 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 value of the elements of $B^1_{G_3}$

<table>
<thead>
<tr>
<th></th>
<th>$S - C_1$</th>
<th>$S - C_3$</th>
<th>$S - C_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S = G_3 + C_2$</td>
<td>63</td>
<td>60</td>
<td>69</td>
</tr>
<tr>
<td>$S = G_3 + C_4$</td>
<td>47</td>
<td>60</td>
<td>54</td>
</tr>
<tr>
<td>$S = G_3 + C_5$</td>
<td>45</td>
<td>58</td>
<td>52</td>
</tr>
<tr>
<td>$S = G_3 + C_6$</td>
<td>49</td>
<td>56</td>
<td>56</td>
</tr>
<tr>
<td>$S = G_3 + C_7$</td>
<td>63</td>
<td>55</td>
<td>64</td>
</tr>
</tbody>
</table>

Table 4 reveals 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 $B^1_{G_3}$ 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 greater size than one). Finally, we underline that because $f(G_3) \leq f(S_3) \leq f(S^*_3)$ (see Lemma 3), running the 2SPG1 ALGORITHM would be a waste of time.

Our simple example clearly reveals that for a given instance of the Problem (3), there is no guarantee that a feasible collection $S \in B^1_{G_k}$ 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 reveals when such an improvement will be 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 are we certain that the greedy set is locally maximal (i.e. a different collection $S \in B^1_{G_k}$ does not exist to improve upon $G_k$)? The answer is provided by the following theorem:

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

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

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

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

15
**Proof** We prove the contrapositive statement. In order for \( f(G_k) < f(S) \), with \( S \in \mathcal{B}_{G_k}^1 \), an element \( z' \in G_{k+1} \) is required to exist to satisfy that
\[
\frac{f(G_{k+1}) - f(G_{k+1} - z')}{f(G_{k+1}) - f(G_k)} < \frac{f(G_{k+1}) - f(S_k)}{f(G_{k+1}) - f(S^*_k)}
\]
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. \( \blacksquare \)

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, owns 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 \( \mathcal{B}_{G_k}^1 \).

2. **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 \( \mathcal{B}_{G_k}^1 \) allows for a speedup of the order of \( 1 - \frac{(n+1)}{k(n-k-1)} \).
3. **Tradeoff.** As a ratio, the post-greedy index synthesizes the tradeoff characterizing our local search. 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.

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 share the same image of $g_1$.

5. **Cardinality of 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 $V$ possibly expands the number of new candidate elements that can be exposed to inequality (8).

5.2.2 **CCTV example**

According to our example, we see that:

$$\Gamma_{3,1} = \frac{f(g_1 | G_3)}{1 - \gamma_{3,1}} = \frac{f(G_4) - f(G_3)}{1 - \gamma_{3,1}} = \frac{78 - 70}{1 - 0.703} = 27.$$  

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 $B^1_{G_3}$. In other words, no swap of size is able to improve the value of the greedy collection and we are guaranteed that replacing one camera with another is a useless strategy. 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 $G_k$, why should we restrict our attention to swaps of size one? Can we do better by producing a swap of greater size? Could such a strategy succeed where a swap of a smaller size proved to fail? Does this produce (sufficient) conditions that could guarantee us the local optimality of a greedy set $f(G_k)$? Does this 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 will consider swaps of progressively larger sizes, that is up to $L \geq 1$, a number that is set and known *a priori*. This can be done by extending our local search around $G_k$ to larger neighborhoods $B^2_{G_k}, ..., B^L_{G_k}$. 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 $v \in (V - S_{j-1})$. Now, 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 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 transformed into the following solution.

Formally speaking, let \(G_k\) be the available solution at the step 0 of the iterative sequence. We label with \(\lambda_j\) and \(\epsilon_j\) 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 highest quality solution that was visited during the sequence.

We acknowledge that the ejection \(\lambda_j\) 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. Then:

\[
\hat{u}_j = g_{k+j} = \arg\max_{u \in (V - G_{k+j-1})} f(u|G_{k+j-1}) \implies f(\hat{u}_j|G_{k+j-1}) = f(G_{k+j})
\]

and then, to recreate feasibility:

\[
\hat{Z}_j = \arg\min_{Z_j \subseteq G_{k+j}} f(G_{k+j} - Z_j) \implies f(G_{k+j} - \hat{Z}_j).
\]

Ultimately, a swap of size \(j^*\) is implemented, where:

\[
j^* = \arg\max_{j \leq L} f(G_{k+j} - \hat{Z}_j).
\]

A graphical description of the procedure that we just described 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 \(j^*\)

![Diagram of the two-step post-greedy algorithm](image)

The 2SGtj 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 will inform us of
Algorithm 4: Two-step post-greedy algorithm of size $j^*$ ($2sgi_j^*$)

**Input:** $f : 2^V \rightarrow \mathbb{R}_+$, $G_k$, $f(G_k)$, 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. $imp \leftarrow \text{.FALSE.}$;
4. $count \leftarrow 0$;
5. **for** $j = 1, \ldots, L$ **do**
6. $\quad g^* = \arg\max_{u \in (V - S_L)} f(u|S_L)$ (choose arbitrarily in case of ties)
7. $\quad \text{if } f(S_L + g^*) > f(S_L) \text{ then}$
8. $\quad \quad S_L \leftarrow S_L + g^*$;
9. $\quad \quad Z_L = \arg\max_{Z \subseteq S_L, |Z| = j} f(S_L - Z)$ (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. $\quad \quad imp \leftarrow \text{.TRUE.}$;
13. $\quad \quad count \leftarrow j$;
14. **return** $S_L, f_L, imp, count$.

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 $2sgi_j^*$ 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 $2sgi_j^*$ 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 $f : 2^V \rightarrow \mathbb{R}_+$, 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 an equality.

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

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

(9)

for all $Z_j = \{z_1, \ldots, z_j\} \subseteq G_{k+j}$ and all $j \leq L$, then there is no collection $S_k^L \in \mathcal{B}_k^L$ 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:

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

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

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

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

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

for some ordered sequence of sets $Z'_1 \subseteq \ldots \subseteq Z'_j \subseteq G_{k+j}, \ j \leq L$. 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 $Z'_j \subseteq G_{k+j}$. \qed

We underline the fact that when $j = 1$, expression (9) admits expression (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 $G_k$. 

20
**Lemma 5** For Problem (3) and a polymatroid $f : 2^V \rightarrow \mathbb{R}_+$, 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 the probability of a new, feasible, and enhancing solution to become available via swap improves.

### 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 introduces a useful shortcut that enhances time tractability.

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

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

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

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 $^nP_j = \frac{n!}{(n-j)!}$ (worst-case).

#### 6.1.1 CCTV example

In our example, 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. 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

$$\Gamma_{3,2} = \frac{f(G_5) - f(G_3)}{(1 - \gamma_{3,1})(1 - \gamma_{3,2})} = \infty.$$
Table 5: The candidates for $\gamma_{3,2}$ (numbers above the main diagonal) [CCTV example]

<table>
<thead>
<tr>
<th>$(C_i, C_j)$</th>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_7$</th>
<th>$C_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>-</td>
<td>0.810</td>
<td>0.654</td>
<td>0.514</td>
<td>1</td>
</tr>
<tr>
<td>$C_2$</td>
<td>42</td>
<td>-</td>
<td>0.463</td>
<td>0.784</td>
<td>0.822</td>
</tr>
<tr>
<td>$C_3$</td>
<td>52</td>
<td>54</td>
<td>-</td>
<td>0.611</td>
<td>0.600</td>
</tr>
<tr>
<td>$C_7$</td>
<td>37</td>
<td>54</td>
<td>54</td>
<td>-</td>
<td>0.733</td>
</tr>
<tr>
<td>$C_8$</td>
<td>52</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>-</td>
</tr>
</tbody>
</table>

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 its neighborhood $B_{G_3}^G$. 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 those 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 our investigation is twofold. On one hand, we wanted to challenge the potential optimality of the greedy output for the maximization of a polymatroid subject to a cardinality constraint. We did so by building a test of its local optimality that runs within its swapping neighborhoods. On the other hand, we exploited the post-greedy curvature of the polymatroid to build a simple procedure that, in polynomial time, tells us if we are guaranteed that the greedy is locally maximal. In case the test is verified, no local inspection would be required to improve upon the greedy output, since 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, this procedure may lead to important speedups 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 our work, which is the constrained maximization of a polymatroid. We complete this discussion 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 potential 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 the 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. In addition to the before-mentionedLazy greedy heuristic, the Stochastic greedy Algorithm (SG) is perhaps the most important (Mirzasoleiman et al., 2015).
et al., 2018) alternative to the adaptive greedy procedure. In its $i$-th iteration, SG randomly samples $(-n/k \ln \epsilon)$ elements of $(V - G_{r_{i-1}})$, with $\epsilon \in (e^{-k}, 1)$, and where $G_{r_{i-1}}$ denotes the random greedy collection of cardinality $k < n$ 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 paper and every statement we have made about the greedy set $G_k$ could now be made with reference to $G_{r_k}$. In practice, every conclusion reached in this work will be easily adapted and remain valid.

- **Addressing ties.** The greedy procedure that yields the formation of the collection $G_k$ may cause ties that, in our paper, we addressed by making an arbitrary choice. For example, in the particular context of our CCTV exercise, the arbitrary choice that we made in the second iteration of the greedy procedure is the cause of a suboptimal greedy solution. 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 proposed for the AG algorithm in the context of a constrained maximization of a polymatroid. For this reason, an in-depth analysis of alternative tie-breaking models may represent a fruitful line of investigation for future works.

b) **Beyond-greedy extensions**

- **Variable neighborhoods.** Both the 1SPG1 and the 2SPG1 heuristics that we described in Chapter 5 are performed by operating swaps of size one on the greedy solution of Problem (3). If an improvement is found, then we replace $G_k$ with the maximum of $f(S)$ that was encountered in the neighborhood. If there is no possibility of improvement, the heuristic returns the greedy set. Since local search procedures 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 expose to a swap of size one. If the ‘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 $B^j_{S_{k+1}}$, with $j > 1$, until we find a feasible set that outperforms the greedy outcome. Then, we iteratively repeat the same steps by using the new set. This procedure describes the so-called VARIABLE NEIGHBORHOOD SEARCH (VNS) meta-heuristic, which 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). Given that the VNS process may represent a very efficient way to escape the greedy trap, we recognize it as a topic that deserves more exploration and formal treatment in future lines of work.

- **Iterated swaps and ejection chains.** The careful reader may find that one of our work’s main rigidities in the lack of iteration during the swapping process. Indeed,
once the greedy set is replaced by a ‘better’ set $S_L$, no more swappings will take place.
Yet, there is no reason for not trying to fish for additional improvements via a new
iteration starting from $S_L$ (rather than from $G_k$). A known procedure for doing so
is called ejection chains algorithm (EC), which was first introduced by Glover
(1991, 1992) and then 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 EC, as well as of all the procedures we discuss below, is that once the
iteration takes us away from the greedy set we are no longer able to exploit its formal
characteristics. In practice, a uniform treatment of the subject becomes progressively
difficult to maintain.

- **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 a number of ideas designed to improve a local search starting from
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 prohibi-
tion (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 iteratively move from one potential
solution to an improved solution, until some stopping criterion is met (Glover, 1989,
1990a,b). We believe that TS should be seen as a valid alternative to VNS and, for
this reason, be considered as a topic of relevance for future investigations.

- **GRASP.** As a method for producing new and feasible solutions starting from an
initial greedy collection, a greedy randomized adaptive search procedure
(GRASP) was introduced by Feo and Resende (1995) and systematized by Resende
and Ribeiro (2010, 2016). It represents an iterative process consisting 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 feature of a 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 called restricted candidate list
(RCL). In this sense, the construction phase replicates the key steps of the SG heuristic
that we discussed above. Then, the local search phase improves the constructed
solution in an iterative fashion by successively replacing the incumbent solution by
a better solution in its neighborhood. The process terminates when no improvement
is found in the neighborhood. The GRASP heuristic is adaptive in the sense that
the discrete derivative associated to each candidate in the RCL is updated at each
iteration of the construction phase. 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 Patricio de Sóusa. I also would like to thank professor Santiago Budria and professor Manuel Alejandro Betancourt Odio for their encouragement and valuable comments.
Appendix. Proof of 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 \( f : 2^V \rightarrow \mathbb{R}_+ \). 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 \( g_i = \arg\max_{z \in (V-G_{i-1})} f(z|G_{i-1}) \), and the second inequality exploits the definition of submodularity expressed by (1).

Lemma 2. For problem (3) with a polymatroid \( f : 2^V \rightarrow \mathbb{R}_+ \), 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 \( \alpha \).

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:

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

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

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

\[
\gamma_{k,j} - \gamma_{k,j-1} = \frac{f(G_{k,j-1}) - f(G_{k,j-1} - Z''_{j-1})}{f(Z''_{j-1})} - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_{j})}{f(Z'_{j})} \leq \frac{f(G_{k,j-1}) - f(G_{k,j-1} - Z'_{j-1})}{f(Z'_{j-1})} - \frac{f(G_{k,j}) - f(G_{k,j} - Z'_{j-1})}{f(Z'_{j-1})} \leq 0.
\]

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 \( j \).

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 \( \alpha \).

**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 equalities.

**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 \( B^1(G_k) \) 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 \( f : 2^V \rightarrow \mathbb{R}_+ \), 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 an 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 \( f : 2^V \rightarrow \mathbb{R}_+ \), 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 \( \Gamma_{k,j} = \sum_{i=1}^j f(g_{k+i}) \) 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 \( f(G_{k+j}) - f(G_k) = \sum_{i=1}^j f(g_{k+i}) \) and \( \gamma_{k,j} = 0 \) (see Lemma 2) for \( j \geq 1 \).

Corollary 6. Given a polymatroid \( f : 2^V \rightarrow \mathbb{R}_+ \) 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


