Acta Informatica (2026) 63:31
https://doi.org/10.1007/s00236-026-00546-x

RESEARCH

Incremental–decremental maximization

Yann Disser[1] · Max Klimm[2] · Annette Lutz[1] · Lea Strubberg[2]

Received: 30 December 2025 / Accepted: 3 August 2026 / Published online: 14 August 2026
© The Author(s) 2026

Abstract
We introduce a framework for incremental–decremental maximization that captures the gradual transformation or renewal of infrastructures. In our model, an initial solution is transformed one element at a time and the utility of an intermediate solution is given by the sum of the utilities of the transformed and untransformed parts. We propose a simple randomized algorithm and a more sophisticated deterministic algorithm, both of which find an order in which to transform the elements while maintaining a large utility during all stages of transformation, relative to an optimal solution for the current stage. More specifically, our algorithms yield competitive solutions for utility functions of bounded curvature and/or generic submodularity ratio, and, in particular, for submodular functions and functions satisfying the gross substitutes property. Our results show that incremental–decremental maximization is substantially more difficult than incremental maximization.

1 Introduction

Incremental maximization provides a mathematical model to study situations where a new infrastructure is constructed over time. The infrastructure consists of a set of elements E. The benefit for having built a subset of elements is defined by a set function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ that maps each partial solution $S \subseteq E$ to its benefit $f(S)$. Depending on the application, the elements may model, e.g., pipes in a pipeline system, wires in an electric network, or machines in a workshop. The benefit $f(S)$ models, e.g., average annual returns from operating the partial infrastructure S. Given such a situation, it is a natural question in which order the

A preliminary version of this article appeared in Proceedings of the 23rd Workshop on Approximation and Online Algorithms (WAOA), 2025, pp. 97–110.

✉ Max Klimm
klimm@math.tu-berlin.de

Yann Disser
disser@mathematik.tu-darmstadt.de

Annette Lutz
lutz@mathematik.tu-darmstadt.de

Lea Strubberg
strubberg@math.tu-berlin.de

[1] TU Darmstadt, Darmstadt, Germany

[2] TU Berlin, Berlin, Germany

elements should be constructed. This question is addressed by the theory of incremental maximization.

Formally, an incremental solution is an order $\pi = (e_1, \dots, e_n)$ of the elements of E where $n = |E|$. The quality of an incremental solution is measured by a competitive analysis. To this end, for $k \in \{0, \dots, n\}$, let

$$\mathrm{OPT}_k := \max\big\{ f(S) : S \subseteq E \text{ with } |S| = k \big\}$$

be the value of an optimal solution with k elements. For an incremental solution $\pi = (e_1, \dots, e_n)$ and $k \in \{1, \dots, n\}$, let $S_k := \{e_1, \dots, e_k\}$ denote the prefix of the first k elements of π. Then, the competitive ratio of an incremental solution π is defined as $\rho := \inf\big\{r \in \mathbb{R}_{\geq 0} : r\, f(S_k) \geq \mathrm{OPT}_k \text{ for all } k \in \{1, \dots, n\}\big\}$, and one is interested in finding incremental solutions with a low competitive ratio. In this framework, upper and lower bounds on the best possible competitive ratio have been derived for various classes of set functions f [1–4].

While the incremental maximization framework is a suitable model for situations where a new infrastructure needs to be constructed from scratch, it fails to capture scenarios where an existing infrastructure is upgraded to a new technological standard. As an example, suppose that E is a set of sinks in a gas supply network, and let $g(S)$ be the maximum flow value of natural gas from a fixed source to a subset $S \subseteq E$ of the sinks in the existing network. The sinks are to be converted to operate with hydrogen instead; let $h(S)$ be the maximum flow value of hydrogen from a fixed source in the existing hydrogen network to a subset $S \subseteq E$ of converted sinks. In this context it is natural to ask in which order the sinks should be converted to operate with hydrogen in order to ensure that the amount of energy supplied in total across both networks is high at every point in time. To phrase and answer these types of questions formally, we introduce a novel *incremental–decremental* optimization framework that generalizes incremental maximization.

In analogy to the incremental maximization framework, we call an ordering $\pi = (e_1, \dots, e_n)$ of the elements an incremental–decremental solution. Such an order increments the set of elements using the new technology while decrementing the set of elements using the old technology, hence the name. For two arbitrary set functions $h, g \colon 2^E \to \mathbb{R}_{\geq 0}$, let $f \colon 2^E \to \mathbb{R}_{\geq 0}$ be defined as $f(S) := h(S) + g(E \setminus S)$. Note that in the application above $f(S)$ is equal to the total amount of energy supplied using both networks once the subset $S \subseteq E$ of sinks has been converted to operate with hydrogen. More generally, this additive model captures situations in which the contributions of the old and the transformed elements to the objective value are independent. As before, let $\mathrm{OPT}_k := \max\big\{ f(S) : S \subseteq E \text{ with } |S| = k \big\}$. The *competitive ratio* of the incremental–decremental solution π is then defined as

$$\rho := \inf\big\{ r \in \mathbb{R}_{\geq 0} : r f(S_k) \geq \mathrm{OPT}_k \text{ for all } k \in \{1, \dots, n\} \big\}.$$

The competitive ratio of an algorithm is the supremum over the competitive ratios of its solutions. Likewise the competitive ratio of a problem is the infimum over the competitive ratios of an algorithm for the problem. We say that an algorithm or a solution is *ρ-competitive* if its competitive ratio is at most ρ.

Since the notions of an incremental–decremental solution for h and g, and that of a purely incremental solution for f defined by $f(S) = h(S) + g(E \setminus S)$ coincide, one may wonder why it is insightful to introduce incremental–decremental solutions in the first place. The reason for this stems from the fact that the incremental–decremental framework allows to impose natural requirements on the functions h and g that may not be phrased as conveniently for the combined function f. Specifically, we focus on the case where g and h are monotone and submodular. For instance, this is the case in the above example of the conversion from natural

gas to hydrogen, if we suppose that both the gas and hydrogen networks are capacitated flow networks and that $h(S)$ and $g(S)$ denote the maximum amount of flow that can be routed from a fixed source to the set of sinks S (see [5]). While the combined objective $f(S) = h(S) + g(E\backslash S)$ is also submodular in this case, it is generally non-monotone. As a consequence, known results for incremental maximization do not apply, and, as we will see, the best possible competitive ratio in the incremental–decremental setting is strictly worse than that in the incremental setting.

On the other hand, while our framework allows for non-monotone objectives $f(S) = h(S) + g(E\backslash S)$, requiring g and h to be monotone and submodular means that our results only apply to a structured subclass of non-monotone submodular functions. In fact, incremental maximization with non-monotone, submodular objective does not allow for bounded (deterministic) competitive ratios, as we show in Proposition 18 in Section A.

1.1 Our results

We give polynomial-time algorithms computing an incremental–decremental solution whose performance guarantees are parametrized by the generic submodularity ratio $\gamma \in (0, 1]$ and the curvature $c \in (0, 1]$ of the functions g and h; these properties of set functions are formally defined in Definitions 1 and 2.

As a warm-up, we examine a simple randomized algorithm that returns, uniformly at random, either the greedy order of the elements in E with respect to h or the reverse greedy order with respect to g. For this randomized algorithm, an upper bound of $2c\frac{e^{c\gamma}}{e^{c\gamma}-1}$ on the competitive ratio follows rather straightforwardly from known results for incremental maximization.

As our main contribution, we then introduce a deterministic algorithm that can be viewed as a way to derandomize the randomized approach. Instead of following the greedy order either only for g or only for h, the algorithm builds both greedy orders partially at the same time depending on which promises a larger incremental gain in the current step. For this deterministic algorithm that we term *double-greedy*, we prove the following competitive ratio which improves on the randomized competitive ratio for a large range of values of the curvature $c \in (0, 1]$ and the generic submodularity ratio $\gamma \in (0, 1]$. Specifically, it provides a better competitive ratio as long as $\gamma \geq 0.54$, which includes, in particular, all submodular functions.

Theorem 1 *For monotone functions $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ with curvature $c \in (0, 1]$ and generic submodularity ratio $\gamma \in (0, 1]$, the double-greedy algorithm is $\frac{1}{\gamma}\left(1 + c\frac{e^c}{e^c-1}\right)$-competitive.*

Using that for a submodular function, we have $\gamma = 1$, $c \in (0, 1]$, and that the function $c \mapsto c\frac{e^c}{e^c-1}$ is non-decreasing in c, we obtain the following direct corollary of Theorem 1.

Corollary 2 *For monotone submodular functions $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$, the double-greedy algorithm is $\left(1 + \frac{e}{e-1}\right)$-competitive.*

We further consider gross substitutes functions, a subclass of submodular functions. For this class of functions, we show the following result.

Theorem 3 *For monotone and gross substitutes functions $g, h : 2^E \to \mathbb{R}_{\geq 0}$ the double-greedy algorithm is 2-competitive.*

We complement our results by the following lower bounds for deterministic algorithms.

Table 1 Lower and upper bounds on the competitive ratio of incremental–decremental maximization for different classes of functions h and g with curvature c and generic submodularity ratio γ; the result with shaded background is for randomized algorithms. Known upper bounds on the competitive ratios for incremental maximization (corresponding to the special case $g \equiv 0$) are for comparison

	Incremental–decremental				Incremental	
	Lower bound		Upper bound		Upper bound	
General	$1 + \frac{c}{2-c}$	(Thm 4)	$2c\frac{e^{c\gamma}}{e^{c\gamma}-1}$	(Cor 6)	$c\frac{e^{c\gamma}}{e^{c\gamma}-1}$	[6]
[5pt]	$\frac{2}{3} + \frac{1}{3\gamma}$	(Thm 4)	$\frac{1}{\gamma}(1 + c\frac{e^c}{e^c-1})$	(Thm 1)		
Submodular	2	(Thm 4)	$1+\frac{e}{e-1} \approx 2.58$	(Cor 2)	$\frac{e}{e-1}$	[7]
Gross substitutes	5/4	(Thm 4)	2	(Thm 3)	1	[8]

Theorem 4 *Let $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ be monotone functions. Then no deterministic algorithm has a better competitive ratio than*

- $1 + \frac{c}{2-c}$ *if g and h are submodular with curvature $c \in [0, 1]$;*
- $\frac{2}{3} + \frac{1}{3\gamma}$ *if g and h have generic submodularity ratio $\gamma \in (0, 1]$;*
- $5/4$ *if g and h are gross substitutes.*

For an overview of known results as well as our contributions, we refer to Table 1.

1.2 Related work

Our work on incremental–decremental maximization directly relates to the literature on incremental maximization which corresponds to the special case that $g \equiv 0$. The natural greedy algorithm for cardinality-constrained maximization inherently produces an incremental solution. Its analysis for monotone submodular functions by Nemhauser et al. [7] translates to a competitive ratio of $\frac{e}{e-1}$. This approximation guarantee is best possible among all polynomial-time algorithms, unless $\mathsf{P} = \mathsf{NP}$, as shown by Feige [9]. It is interesting to note that in a setting where both h and g are monotone submodular, our results imply that the best possible competitive ratio is in the interval $[2, 1 + \frac{e}{e-1}]$, thus separating the competitive ratios achievable in the incremental and the incremental–decremental settings. For monotone gross substitutes functions, a result of Paes Leme [8] implicitly yields an incremental solution that is 1-competitive. In contrast, our results imply that in a setting where both g and h have the gross substitutes property, the best possible competitive ratio for incremental–decremental maximization is in the interval $[5/4, 2]$, showing that incremental–decremental maximization is strictly more difficult than incremental maximization also for this class of functions. For submodular functions with curvature c, Conforti and Cornuéjols [10] provided a more fine-grained analysis that yields a competitive ratio of $c\frac{e^c}{e^c-1}$ for the greedy algorithm. For functions with submodularity ratio $\tilde{\gamma}$, Das and Kempe [11] showed that the greedy algorithm yields a $\frac{e^{\tilde{\gamma}}}{e^{\tilde{\gamma}}-1}$-approximation. Bian et al. [6] combined the analysis of Conforti and Cornuéjols with the analysis of Das and Kempe and showed that for functions of curvature c and submodularity ratio $\tilde{\gamma}$, the greedy algorithm yields a $c\frac{e^{c\tilde{\gamma}}}{e^{c\tilde{\gamma}}-1}$-approximation. Disser and Weckbecker [3] considered the more general setting of $\tilde{\gamma}$-α-augmentable functions and showed that the greedy algorithm is a $\frac{\alpha}{\tilde{\gamma}} \cdot \frac{e^{\alpha}}{e^{\alpha}-1}$-approximation. This class of functions contains all functions with submodularity ratio $\tilde{\gamma}$ as well as all α-augmentable functions. While the results above are stated in terms of the submodularity ratio $\tilde{\gamma}$, they also hold for the generic submodularity ratio γ since $\tilde{\gamma} \geq \gamma$.

Incremental maximization beyond the greedy algorithm has been considered by Bernstein et al. [1]. For accountable functions, they gave a $(\varphi+1)$-competitive solution where $\varphi \approx 1.62$ is the golden ratio. Here, a function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ is called accountable if for all $S \subseteq E$ there is $s \in S$ such that $f(S\backslash\{s\}) \geq f(S) - f(S)/|S|$. They further showed that no deterministic algorithm can provide a competitive ratio better than 2.18 in this setting. Disser et al. [2] gave an improved lower bound of 2.24 and gave a randomized upper bound of 1.77.

Goemans and Unda [12] considered a different setting of incremental maximization where, instead of the worst-case ratio over all time steps, the ratio of the sum over all time steps is to be maximized, and obtained bounds for special functions containing rank functions of independence systems.

The double-greedy algorithm that we consider is reminiscent of double-greedy algorithms for non-monotone submodular maximization. Buchbinder et al. [13] used this approach and obtained a deterministic 3-approximation and a randomized 2-approximation for this problem. Feige et al. [14] showed earlier that no better approximation than 2 is possible with polynomially many oracle calls, implying that the randomized 2-approximation of Buchbinder et al. cannot be improved. Non-monotone submodular function maximization has also been considered under knapsack, matroid, and other constraints [15–18].

2 A simple randomized algorithm

As a warm-up, we first give a simple randomized algorithm for the incremental–decremental problem. For randomized algorithms, the returned incremental–decremental solution π and, hence, all prefixes S_k are random variables; such an algorithm is called ρ-competitive if $\rho\,\mathbb{E}[f(S_k)] \geq \mathrm{OPT}_k$ for all k.

Proposition 5 *Let $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$. If there is a ρ-competitive algorithm both for the incremental problem for g and for the incremental problem for h, then there is a 2ρ-competitive randomized algorithm for the incremental–decremental maximization problem for $f\colon 2^E \to \mathbb{R}_{\geq 0}$, $S \mapsto h(S) + g(E\backslash S)$.*

Proof The randomized algorithm takes with probability $1/2$ the solution of the incremental algorithm for h and with probability $1/2$ the reverse order of the incremental solution for g. Let π^h and π^g be ρ-competitive solutions to the incremental problem with respect to the set functions h and g, respectively. Let $k \in \{1, \dots, n\}$. Denote with $S_k^h \subseteq E$ the subset with the first k elements of π^h and with $S_k^g \subseteq E$ the subset containing the last k elements of π^g. Let OPT_k^h, OPT_k^g and OPT_k^f be the maximum values of a subset with k elements for objective functions h, g and f, respectively.

The expected value of the first k elements of the order returned by the randomized algorithm for the incremental–decremental problem is then given by

$$\frac{1}{2}\big(f(S_k^h) + f(S_k^g)\big) \geq \frac{1}{2}\big(h(S_k^h) + g(E\backslash S_k^g)\big) \geq \frac{1}{2\rho}\big(\mathrm{OPT}_k^h + \mathrm{OPT}_{n-k}^g\big) \geq \frac{1}{2\rho}\mathrm{OPT}_k^f,$$

where in the second inequality we used ρ-competitiveness of π^h and π^g. □

To obtain concrete bounds on the competitiveness of this algorithm, we use the upper bounds on the competitive ratio of incremental maximization for functions with bounded curvature $c \in (0, 1]$ and bounded generic submodularity ratio $\gamma \in (0, 1]$ listed in Table 1. The generic submodularity ratio is a way to relax submodularity and allows to bound the competitive ratio for functions that are in some sense close to submodular. In the following

definitions, we set $\min \emptyset := 1$. For the marginal increase in $f\colon 2^E \to \mathbb{R}_{\geq 0}$ of an element $e \in E$ or of a set $T \subseteq E$ with respect to $S \subseteq E$, we introduce the notations $f(e \mid S) := f(\{e\} \cup S) - f(S)$ and $f(T \mid S) := f(T \cup S) - f(S)$.

Definition 1 *(Gong et al. [19])* A monotone set function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ has *generic submodularity ratio*

$$\gamma := \min\left\{ \frac{f(e \mid A)}{f(e \mid A \cup B)} : A, B \in 2^E, e \in E \text{ with } f(e \mid A \cup B) > 0 \right\}.$$

Intuitively, the generic submodularity ratio interpolates between monotone functions with $\gamma = 0$ and submodular functions with $\gamma = 1$. To see this, note that for a submodular function, we have $f(e \mid A) \geq f(e \mid A \cup B)$ for all $A, B \in 2^E$ and $e \in E$, so that the minimum is attained for $B = \emptyset$ and the submodularity ratio is $\gamma = 1$. Das and Kempe [11] introduced the stronger notion of the *submodularity ratio*

$$\tilde{\gamma} := \min\left\{ \frac{\sum_{b \in B} f(b \mid A)}{f(B \mid A)} : A, B \in 2^E \text{ with } f(B \mid A) > 0 \right\}$$

and showed that the greedy algorithm is $\frac{e^{\tilde{\gamma}}}{e^{\tilde{\gamma}}-1}$-competitive for $g \equiv 0$. Observe that the generic submodularity ratio γ is always smaller or equal to $\tilde{\gamma}$, since for all subsets $B = \{b_1, \ldots, b_k\}$, $A \subseteq E$ with $f(B \mid A) > 0$,

$$\frac{\sum_{b \in B} f(b \mid A)}{f(B \mid A)} = \frac{\sum_{i=1}^k f(b_i \mid A)}{\sum_{i=1}^k f(b_i \mid A \cup \{b_1, \ldots, b_{i-1}\})} \geq \frac{\sum_{i=1}^k f(b_i \mid A)}{\gamma^{-1} \sum_{i=1}^k f(b_i \mid A)} = \gamma.$$

Thus, their bound implies also $\frac{e^{\gamma}}{e^{\gamma}-1}$-competitiveness in terms of the generic submodularity ratio in the incremental setting.

The curvature $c \in [0, 1]$ measures how close a function is to being supermodular. A function is supermodular if it has curvature $c = 0$.

Definition 2 *(Bian et al. [6])* A monotone set function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ has *curvature*

$$c := 1 - \min\left\{ \frac{f(e \mid A \cup B)}{f(e \mid A)} : A, B \in 2^E, e \in E \backslash B \text{ with } f(e \mid A) > 0 \right\}.$$

Bian et al. [6] showed that the greedy algorithm is $c\frac{e^{c\tilde{\gamma}}}{e^{c\tilde{\gamma}}-1}$-competitive for $g \equiv 0$. For submodular functions the notion of curvature was already introduced by Conforti and Cornuéjols [10]. In this case, the curvature simplifies to

$$1 - \min_{e \in E} \frac{f(e \mid E \backslash \{e\})}{f(\{e\})} \tag{1}$$

because of $f(e \mid A \cup B) \geq f(e \mid E \backslash \{e\})$ and $f(e \mid A) \leq f(\{e\})$. We observe that submodular functions with curvature zero are modular since they are both submodular and supermodular.

The upper bound on the competitive ratio of the incremental problem together with Proposition 5 translates to a first upper bound on the competitive ratio of incremental–decremental maximization.

Corollary 6 *Let $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ be monotone with generic submodularity ratio $\gamma \in (0, 1]$ and curvature $c \in (0, 1]$. The algorithm that returns uniformly at random the greedy order of E with respect to h or the reverse greedy order with respect to g is $2c\frac{e^{c\gamma}}{e^{c\gamma}-1}$-competitive.*

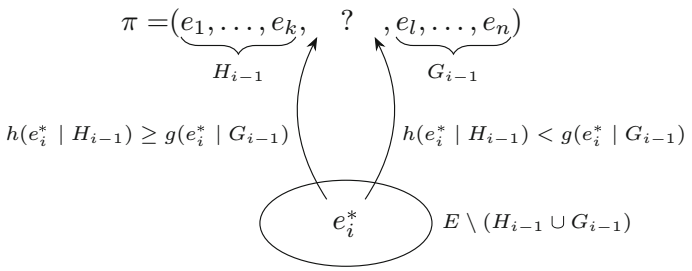


Fig. 1 In step i of the double-greedy algorithm (with $\prec = <$), the element e_i^* maximizing $\max\{g(e \mid G_{i-1}),\ h(e \mid H_{i-1})\}$ among the remaining elements $e \in E \setminus (G_{i-1} \cup H_{i-1})$ is added to the left or right depending on which marginal increase, $g(e_i^* \mid G_{i-1})$ or $h(e_i^* \mid H_{i-1})$, is higher

3 A deterministic algorithm

The double-greedy algorithm that we introduce in this section can be seen as a way to derandomize the randomized solution. Instead of only considering the elements in the greedy order defined by h or the reverse greedy order defined by g, it instead builds partial greedy orders for both functions from both ends simultaneously. Initially, both the considered prefix and the considered suffix of the solution are empty. In each step, an element is selected that maximizes the maximum of the increment relative to the current prefix for h and the increment relative to the current suffix for g. It is appended to the current prefix if the maximum is attained for h, and prepended to the current suffix if the maximum is attained for g. This procedure is visualized in Fig. 1.

In the following, we denote by $F_k \coloneqq G_k \cup H_k$ the set of all elements that are added to the solution by the double-greedy algorithm up to step k. Also, we denote the complement of a set S in E by $S^{\mathrm{c}} \coloneqq E \setminus S$ and write $S_k^{\mathrm{c}} \coloneqq (S_k)^{\mathrm{c}}$ for indexed sets. Further, we generally let $n \coloneqq |E|$ and let $f \colon 2^E \to \mathbb{R}_{\geq 0}$ denote the objective function of the incremental–decremental maximization problem, i.e., $f(S) \coloneqq h(S) + g(S^{\mathrm{c}})$. The double-greedy algorithm is symmetric with respect to the roles of g and h with the slight complication that the handling of ties between g and h is inherently asymmetric. In order to still exploit symmetry, we parameterize the algorithm by a comparison operator $\prec \in \{\leq, <\}$ that governs whether it favors adding elements to G_i or H_i in case of a tie and consider both variants simultaneously. The following proposition makes precise how we can exploit symmetry in our proofs. In particular, it suffices to show competitiveness of the solution up to cardinality $|H_n|$.

Proposition 7 *Let $\mathcal{C} \subseteq \{h \colon 2^E \to \mathbb{R}_{\geq 0}$ monotone$\}$. Assume that for all functions $g, h \in \mathcal{C}$, for all $\prec \in \{\leq, <\}$, for all $k \leq n$ and for all subsets $S \subseteq E$ with $|S| = |H_k|$ the double-greedy algorithm ensures that*

$$\rho f(H_k) \geq f(S), \tag{2}$$

for some $\rho \geq 1$. Then, the double-greedy algorithm is ρ-competitive for all functions $g, h \in \mathcal{C}$.

Proof Let $\pi = (e_1, \ldots, e_n)$ denote the ordering computed by the double-greedy algorithm. Then, the double-greedy algorithm is ρ-competitive if, for all $\ell \leq n$ and all sets $S \subseteq E$ with $|S| = \ell$, we have $\rho f(\{e_1, \ldots, e_\ell\}) \geq f(S)$. Equivalently,

$$\rho f(H_k) \geq f(S) \tag{3}$$

Algorithm: double-greedy algorithm

input : $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ monotone, $\prec \in \{\leq, <\}$
output : ordering $\pi = (e_1, \dots, e_n)$ of the elements in E
$G_0 \leftarrow \emptyset$; $H_0 \leftarrow \emptyset$
for $i \leftarrow 1, \dots, n$:
  $e_i^* \leftarrow \arg\max_{e \in E \setminus (G_{i-1} \cup H_{i-1})} \{\max\{g(e \mid G_{i-1}), h(e \mid H_{i-1})\}\}$
  if $h(e_i^* \mid H_{i-1}) \prec g(e_i^* \mid G_{i-1})$:
    $G_i \leftarrow G_{i-1} \cup \{e_i^*\}$
    $H_i \leftarrow H_{i-1}$
    $e_{n-|G_i|+1} \leftarrow e_i^*$
  else
    $H_i \leftarrow H_{i-1} \cup \{e_i^*\}$
    $G_i \leftarrow G_{i-1}$
    $e_{|H_i|} \leftarrow e_i^*$
return $\pi \leftarrow (e_1, \dots, e_n)$

for all sets $S \subseteq E$ with $|S| = |H_k|$ and

$$\rho f(G_k^{\mathrm{c}}) \geq f(S) \tag{4}$$

for all sets $S \subseteq E$ with $|S| = |G_k^{\mathrm{c}}|$. By assumption, (3) holds and it remains to prove (4).

The double-greedy algorithm is symmetric in the following sense. If we swap the functions g and h in the input and change the choice of $\prec \in \{\leq, <\}$, the double-greedy algorithm produces exactly the reversed ordering of elements, provided that ties are handled consistently.

Denote the modified input to the double-greedy algorithm by $g' \coloneqq h$, $h' \coloneqq g$ and $\prec' \in \{\leq, <\}$ with $\prec' \neq \prec$. Furthermore, we write $f'(S) \coloneqq h'(S) + g'(S^{\mathrm{c}})$. Let H_i' and G_i' be the sets H_i and G_i computed by the double-greedy algorithm for the modified input g', h' and $\prec'$. Similarly, let H_i and G_i be the sets computed by the algorithm on the input g, h and $\prec$. We show by induction on i that $H_i' = G_i$ and $G_i' = H_i$. This trivially holds for $i = 0$.

Consider iteration $i+1$. By induction, $g'(e \mid G_i') = h(e \mid H_i)$ and $h'(e \mid H_i') = g(e \mid G_i)$. Thus, for both inputs the same choice for e_{i+1}^* is possible in the double-greedy algorithm. The if-condition in the double-greedy algorithm with input g', h' and $\prec'$ is $h'(e_{i+1}^* \mid H_i') \prec' g'(e_{i+1}^* \mid G_i')$, which is equivalent to $g(e_{i+1}^* \mid G_i) \prec' h(e_{i+1}^* \mid H_i)$. Hence, the condition is true if and only if the if-condition in the double-greedy algorithm with input g, h and $\prec$ given by $h(e_{i+1}^* \mid H_i) \prec g(e_{i+1}^* \mid G_i)$ is false. Thus, e_{i+1}^* is added to H_i' if and only if it is added to G_i. Similarly for G_i' and H_i, which implies $H_{i+1}' = G_{i+1}$ and $G_{i+1}' = H_{i+1}$. Therefore, the double-greedy algorithm on g', h' and $\prec'$ computes the reversed ordering.

We can now apply (2) for the double-greedy algorithm on g', h' and $\prec'$ to obtain

$$\begin{aligned} \rho f(G_k^{\mathrm{c}}) &= \rho(h(G_k^{\mathrm{c}}) + g(G_k)) = \rho(g'((H_k')^{\mathrm{c}}) + h'(H_k')) \\ &= \rho f'(H_k') \overset{(2)}{\geq} f'(S^{\mathrm{c}}) = f(S), \end{aligned}$$

which finishes the proof. □

From now on, we assume that h and g are *normalized*, i.e., $h(\emptyset) = g(\emptyset) = 0$. This assumption is without loss of generality, since it suffices to show ρ-competitiveness for the shifted functions $g'(S) \coloneqq g(S) - g(\emptyset)$ and $h'(S) \coloneqq h(S) - h(\emptyset)$, because of

$$f(S) \coloneqq h(S) + g(S^{\mathrm{c}}) = h'(S) + h(\emptyset) + g'(S^{\mathrm{c}}) + g(\emptyset) \leq \rho\big(h'(H_k) + g'(H_k^{\mathrm{c}})\big) + h(\emptyset) + g(\emptyset)$$

$$\leq \rho\big(h'(H_k) + h(\emptyset) + g'(H_k^c) + g(\emptyset)\big) = \rho f(H_k), \quad (5)$$

for all $k \leq n$ and $|S| = |H_k|$, which proves competitiveness by Proposition 7. Both shifted functions g' and h' have the same submodularity ratio and curvature as g and h and monotonicity is preserved, so we can apply our analysis to them instead.

We conclude this subsection with a lemma which we will use to prove our upper bounds.

Lemma 8 *Let $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ be monotone. After iteration k of the double-greedy algorithm we have $f(H_k) \geq \sum_{i=1}^{k} \max\{h(e_i^* \mid H_{i-1}), g(e_i^* \mid G_{i-1})\}$.*

Proof By monotonicity of g, using $G_k \subseteq H_k^c$, we have

$$\begin{aligned} f(H_k) &= h(H_k) + g(H_k^c) \geq h(H_k) + g(G_k) \\ &= \sum_{i:e_i^* \in H_k} h(e_i^* \mid H_{i-1}) + \sum_{i:e_i^* \in G_k} g(e_i^* \mid G_{i-1}). \end{aligned}$$

By the definition of the algorithm we obtain

$$\begin{aligned} f(H_k) &\geq \sum_{i:e_i^* \in H_k} h(e_i^* \mid H_{i-1}) + \sum_{i:e_i^* \in G_k} g(e_i^* \mid G_{i-1}) \\ &\geq \sum_{i=1}^{k} \max\{h(e_i^* \mid H_{i-1}), g(e_i^* \mid G_{i-1})\}, \end{aligned}$$

which finishes the proof. □

3.1 Bounded generic submodularity ratio and curvature

In this section, we consider the case where g and h have bounded curvature c and bounded generic submodularity ratio γ which we have defined in Sect. 2. Both parameters, the generic submodularity ratio and the curvature characterize how much the marginal increase of an element is allowed to change on different subsets. A function has generic submodularity ratio γ if the marginal values of each element increase at most by a factor of γ, i.e., we have that

$$f(e \mid S) \geq \gamma f(e \mid T) \qquad \text{for } S \subseteq T \subseteq E, e \in E, \quad (6)$$

where, in Definition 1, we took $A = S$ and $B = T \backslash S$.

Likewise, the curvature measures how much the marginal increase decreases at most, i.e., Definition 2 implies

$$(1 - c) f(e \mid S) \leq f(e \mid T) \qquad \text{for } S \subseteq T \subseteq E, e \in E \backslash T. \quad (7)$$

We proceed to analyze the double-greedy algorithm for functions with fixed generic submodularity ratio and curvature. Recall, that $F_m = H_m \cup G_m$ is the set of elements that are added to the solution up to step m. Let $\varphi_i := \max\{g(e_i^* \mid G_{i-1}), h(e_i^* \mid H_{i-1})\}$ be the increment governing the choice of e_i^* in each step of the algorithm. We obtain a lower bound on φ_i by considering the gap to the value of a larger solution $S \subseteq E$. In particular, we will use this bound with S being an optimum solution for f of size $|S|$.

Lemma 9 *Let $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ be monotone with generic submodularity ratio $\gamma \in (0, 1]$ and curvature $c \in (0, 1]$. For $S \subseteq E$ and $0 \leq m < |S|$ the double-greedy algorithm ensures*

that

$$\varphi_{m+1} \geq \frac{c}{|S\setminus F_m|}\left(\gamma h(S) - \sum_{i=1}^{m}\varphi_i\right) + \frac{1-c}{|S\setminus F_m|}\left(\gamma h(S) - \sum_{i:e_i^*\in F_m\cap S}\varphi_i\right).$$

Proof Let $\{s_1,\dots,s_k\} \coloneqq S\setminus F_m$. Then, we have using $H_m \subseteq \{s_1,\dots,s_{i-1}\}\cup F_m$ that

$$h(S\setminus F_m \mid F_m) = \sum_{i=1}^{k} h(s_i \mid \{s_1,\dots,s_{i-1}\}\cup F_m) \overset{(6)}{\leq} \sum_{s\in S\setminus F_m} \gamma^{-1} h(s \mid H_m). \tag{8}$$

Now, we use $\varphi_{m+1} \geq h(e \mid H_m)$ for $e \notin F_m$ by the greedy choice of e_{m+1}^* to bound

$$\sum_{s\in S\setminus F_m} \gamma^{-1} h(s \mid H_m) \leq |S\setminus F_m|\,\gamma^{-1}\varphi_{m+1}. \tag{9}$$

Together, we can compute

$$\begin{aligned}
h(S) &= h(F_m\cup S) - h(F_m \mid S)\\
&= h(F_m) + h(S\setminus F_m \mid F_m) - h(F_m \mid S)\\
&\overset{(8)}{\leq} h(F_m) + \sum_{s\in S\setminus F_m}\gamma^{-1}h(s \mid H_m) - h(F_m \mid S)\\
&\overset{(9)}{\leq} h(F_m) + |S\setminus F_m|\cdot\gamma^{-1}\varphi_{m+1} - h(F_m \mid S)\\
&= \sum_{i=1}^{m} h(e_i^* \mid F_{i-1}) + |S\setminus F_m|\cdot\gamma^{-1}\varphi_{m+1} - \sum_{i=1}^{m} h(e_i^* \mid S\cup F_{i-1})\\
&\overset{(7)}{\leq} \sum_{i=1}^{m} h(e_i^* \mid F_{i-1}) + |S\setminus F_m|\cdot\gamma^{-1}\varphi_{m+1} - \sum_{i:e_i^*\in F_m\setminus S}(1-c)h(e_i^* \mid F_{i-1})\\
&= (1-c)\sum_{i:e_i^*\in F_m\cap S} h(e_i^* \mid F_{i-1}) + c\sum_{i=1}^{m} h(e_i^* \mid F_{i-1}) + |S\setminus F_m|\,\gamma^{-1}\varphi_{m+1}\\
&\overset{(6)}{\leq} (1-c)\sum_{i:e_i^*\in F_m\cap S}\gamma^{-1}h(e_i^* \mid H_{i-1}) + c\sum_{i=1}^{m}\gamma^{-1}h(e_i^* \mid H_{i-1}) + |S\setminus F_m|\,\gamma^{-1}\varphi_{m+1}\\
&\overset{c\in[0,1]}{\leq} (1-c)\sum_{i:e_i^*\in F_m\cap S}\gamma^{-1}\varphi_i + c\sum_{i=1}^{m}\gamma^{-1}\varphi_i + |S\setminus F_m|\,\gamma^{-1}\varphi_{m+1}.
\end{aligned} \tag{10}$$

Isolation of φ_{m+1} in (10) yields the desired bound. □

We combine the bounds derived in Lemma 9 on the increments φ_{m+1} in each step of the algorithm to bound the value of every solution by the sum of those increments. An optimum solution of each size is particularly bounded as follows.

Lemma 10 *Let $g,h\colon 2^E \to \mathbb{R}_{\geq 0}$ be monotone with generic submodularity ratio $\gamma \in (0,1]$ and curvature $c \in (0,1]$. For $S \subseteq E$ and $k \coloneqq |S|$, the double-greedy algorithm ensures that $\frac{1}{c}\left(1-\frac{1}{\mathrm{e}^c}\right)\gamma h(S) \leq \sum_{i=1}^{k}\varphi_i$.*

Proof First assume $S = F_k$. In this case, since we have $\frac{1}{c}\left(1 - \mathrm{e}^{-c}\right)\gamma \leq 1$ for $c, \gamma \in (0, 1]$, it suffices to compute

$$\gamma h(F_k) \leq \gamma \sum_{i=1}^{k} h(e_i^* \mid F_{i-1}) \overset{(6)}{\leq} \sum_{i=1}^{k} h(e_i^* \mid H_{i-1}) \leq \sum_{i=1}^{k} \varphi_i.$$

In the following, we assume $S \neq F_k$. For $0 \leq m \leq k$ let $k_m \coloneqq |S \backslash F_m|$ and let χ_i indicate whether e_i^* is part of the solution S, i.e. $\chi_i = 1$ if $e_i^* \in S$ and $\chi_i = 0$ otherwise. By definition, we have

$$k_{m+1} = k_m - \chi_{m+1}. \tag{11}$$

We denote the difference between the sum of the increments in each step of the algorithm to the value $\gamma h(S)$, which we seek to bound, by $d_m \coloneqq \gamma h(S) - \sum_{i=1}^{m} \varphi_i$. Likewise, for the sum of only those increments belonging to elements contained in S, we denote the difference by $d_m^* \coloneqq \gamma h(S) - \sum_{i=1}^{m} \chi_i \varphi_i$. If $d_m \leq 0$, the sum of the increments already exceeds the value $\gamma h(S)$ and the statement of the lemma follows directly since we have $\frac{1}{c}\left(1 - \mathrm{e}^{-c}\right)\gamma \leq 1$ for $c, \gamma \in (0, 1]$. Thus, we assume in the following that $d_m > 0$, which implies $d_m^* > 0$ as in the difference d_m^* only the increments of elements in S are subtracted from $\gamma h(S)$.

We show by induction on $0 \leq m \leq k$ that

$$\begin{aligned}\left[\left(1 - \frac{c}{k}\right)^k - 1 + c\right]\gamma h(S) \\ \geq \left(1 - \frac{c}{k}\right)^{k-m} c\, d_m - \left(1 - \left(1 - \frac{c}{k}\right)^{k-m}\right)\frac{(1-c)k}{k_m} d_m^*.\end{aligned} \tag{12}$$

For $m = k$ the last term vanishes and the inequality bounds the difference d_k in terms of $\gamma h(S)$ which can be rearranged to finish the proof, as we show in the end.

For $m = 0$, (12) holds with equality since by $d_0 = d_0^* = \gamma h(S)$ and $k_0 = |S \backslash F_0| = k$ we have

$$\begin{aligned}&\left(1 - \frac{c}{k}\right)^k c\, d_0 - \left(1 - \left(1 - \frac{c}{k}\right)^k\right)\frac{(1-c)k}{k_0} d_0^* \\ &\quad = \left[\left(1 - \frac{c}{k}\right)^k c - \left(1 - \left(1 - \frac{c}{k}\right)^k\right)(1-c)\right]\gamma h(S) \\ &\quad = \left[\left(1 - \frac{c}{k}\right)^k - 1 + c\right]\gamma h(S).\end{aligned}$$

In the following, we assume the statement holds for $m \geq 0$ to prove it for $m + 1$. We first introduce the increment φ_{m+1} into the right hand side of (12) for $m + 1$ to then leverage our bound on the increment from Lemma 9. With this bound we can conclude the induction by a long computation showing that the right hand side of (12) is decreasing in m.

Inserting $d_{m+1} = d_m - \varphi_{m+1}$ and $d_{m+1}^* = d_m^* - \chi_{m+1}\varphi_{m+1}$ and isolating the factor before φ_{m+1} yields

$$\begin{aligned}&\left(1 - \frac{c}{k}\right)^{k-m-1} c\, d_{m+1} - \left(1 - \left(1 - \frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}} d_{m+1}^* \\ &\quad = \left(1 - \frac{c}{k}\right)^{k-m-1} c\, d_m - \left(1 - \left(1 - \frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}} d_m^* \\ &\quad - \left[\left(1 - \frac{c}{k}\right)^{k-m-1} c - \left(1 - \left(1 - \frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\chi_{m+1}\right]\varphi_{m+1}.\end{aligned} \tag{13}$$

To apply Lemma 9, we show non-negativity of the factor before $-\varphi_{m+1}$. For this, we use Bernoulli's inequality which states that

$$(1-x)^n \geq 1-nx \qquad \text{for } x \geq -1,\, n \in \mathbb{N}. \tag{14}$$

With $c \in [0,1]$ and $\frac{c}{k} \leq 1$ we obtain non-negativity of the factor before $-\varphi_{m+1}$ given by

$$\begin{aligned}
&\left(1-\frac{c}{k}\right)^{k-m-1} c - \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\chi_{m+1} \\
&\overset{\chi_{m+1}\leq 1}{\geq} \left(1-\frac{c}{k}\right)^{k-m-1}\left(c+\frac{(1-c)k}{k_{m+1}}\right) - \frac{(1-c)k}{k_{m+1}} \\
&\overset{(14)}{\geq} \left(1-\frac{(k-m-1)c}{k}\right)\left(c+\frac{(1-c)k}{k_{m+1}}\right) - \frac{(1-c)k}{k_{m+1}} \\
&= c - \frac{(k-m-1)c^2}{k} - \frac{(k-m-1)c(1-c)}{k_{m+1}} \\
&\overset{k_{m+1}\leq k}{\geq} c - \frac{k-m-1}{k_{m+1}}c \\
&\geq 0,
\end{aligned} \tag{15}$$

where we use $k_{m+1} \geq k-m-1$ in the last step. Reformulating Lemma 9 in terms of the definitions of k_m, d_m and d_m^*, we have

$$\varphi_{m+1} \geq \frac{c}{k_m} d_m + \frac{1-c}{k_m} d_m^*. \tag{16}$$

Applying this to the right hand side of (12) for $m+1$ yields

$$\begin{aligned}
&\left(1-\frac{c}{k}\right)^{k-m-1} c\, d_{m+1} - \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}} d_{m+1}^* \\
&\overset{(13)}{=} \left(1-\frac{c}{k}\right)^{k-m-1} c\, d_m - \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}} d_m^* \\
&\quad - \left[\left(1-\frac{c}{k}\right)^{k-m-1} c - \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\chi_{m+1}\right]\varphi_{m+1} \\
&\overset{(15),(16)}{\leq} \left(1-\frac{c}{k}\right)^{k-m-1} c\, d_m - \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}} d_m^* \\
&\quad - \left[\left(1-\frac{c}{k}\right)^{k-m-1} c\right]\left(\frac{c}{k_m} d_m + \frac{1-c}{k_m} d_m^*\right) \\
&\quad + \left[\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\chi_{m+1}\right]\left(\frac{c}{k_m} d_m + \frac{1-c}{k_m} d_m^*\right) \\
&= \left(1-\frac{c}{k}\right)^{k-m-1}\left(1-\frac{c}{k_m}\right) c\, d_m \\
&\quad - \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\left(1-\chi_{m+1}\frac{1-c}{k_m}\right) d_m^* \\
&\quad - \left(1-\frac{c}{k}\right)^{k-m-1}\frac{c(1-c)}{k_m} d_m^* \\
&\quad + \left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\chi_{m+1}\frac{c}{k_m} d_m
\end{aligned}$$

$$
\begin{aligned}
&\overset{k\geq k_m}{\leq} \left(1-\frac{c}{k}\right)^{k-m} c\, d_m \\
&\quad -\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\left(\frac{k_m-\chi_{m+1}+c\chi_{m+1}}{k_m}\right)d_m^* \\
&\quad -\left(1-\frac{c}{k}\right)^{k-m-1}\frac{c(1-c)}{k_m}d_m^* \\
&\quad +\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{(1-c)k}{k_{m+1}}\chi_{m+1}\frac{c}{k_m}d_m
\end{aligned}
$$

$$
\begin{aligned}
&\overset{(11)}{=} \left(1-\frac{c}{k}\right)^{k-m} c\, d_m \\
&\quad -\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{k}{k_{m+1}}\left(\frac{k_{m+1}+c\chi_{m+1}}{k_m}\right)(1-c)d_m^* \\
&\quad -\left(1-\frac{c}{k}\right)^{k-m-1}\frac{c}{k_m}(1-c)d_m^* \\
&\quad +\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{k}{k_{m+1}}\frac{c\chi_{m+1}}{k_m}(1-c)d_m \\
&= \left(1-\frac{c}{k}\right)^{k-m} c\, d_m \\
&\quad -\left[\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{k}{k_m}+\left(1-\frac{c}{k}\right)^{k-m-1}\frac{c}{k_m}\right](1-c)d_m^* \\
&\quad +\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{k}{k_{m+1}}\frac{c\chi_{m+1}}{k_m}(1-c)(d_m-d_m^*) \\
&\overset{d_m\leq d_m^*}{\leq} \left(1-\frac{c}{k}\right)^{k-m} c\, d_m \\
&\quad -\left[\left(1-\left(1-\frac{c}{k}\right)^{k-m-1}\right)\frac{k}{k_m}+\left(1-\frac{c}{k}\right)^{k-m-1}\frac{c}{k_m}\right](1-c)d_m^* \\
&= \left(1-\frac{c}{k}\right)^{k-m} c\, d_m-\left[1-\left(1-\frac{c}{k}\right)^{k-m-1}\left(1-\frac{c}{k}\right)\right]\frac{(1-c)k}{k_m}d_m^* \\
&= \left(1-\frac{c}{k}\right)^{k-m} c\, d_m-\left(1-\left(1-\frac{c}{k}\right)^{k-m}\right)\frac{(1-c)k}{k_m}d_m^* \\
&\overset{(12)}{\leq} \left[\left(1-\frac{c}{k}\right)^{k}-1+c\right]\gamma h(S),
\end{aligned}
$$

where we used the induction hypothesis for m in the last inequality.

For $m=k$, (12) yields

$$
\left[\left(1-\frac{c}{k}\right)^{k}-1+c\right]\gamma h(S)\geq c\left(\gamma h(S)-\sum_{i=1}^{k}\varphi_i\right). \tag{17}
$$

Rearranging we derive, using $\left(1+\frac{(-c)}{k}\right)^k\leq \mathrm{e}^{-c}$ for $k\geq 1$, $|c|\leq k$, that

$$
c\sum_{i=1}^{k}\varphi_i \overset{(17)}{\geq} \left(1-\left(1-\frac{c}{k}\right)^{k}\right)\gamma h(S)\geq\left(1-\frac{1}{\mathrm{e}^c}\right)\gamma h(S). \qquad \square
$$

With this at hand, we are ready to prove our main result.

Theorem 1 *For monotone functions $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ with curvature $c \in (0, 1]$ and generic submodularity ratio $\gamma \in (0, 1]$, the double-greedy algorithm is $\frac{1}{\gamma}\left(1 + c\frac{e^c}{e^c-1}\right)$-competitive.*

Proof Let $k \leq n$ and $S \subseteq E$ with $\ell := |H_k| = |S|$. By Proposition 7, it suffices to show that

$$\gamma^{-1}\left(1 + c\frac{e^c}{e^c - 1}\right) f(H_k) \geq f(S). \tag{18}$$

Since $\varphi_i \geq 0$ by monotonicity of g and h, and since $k \geq \ell$, we obtain

$$f(H_k) \overset{\text{Lem. 8}}{\geq} \sum_{i=1}^{k} \varphi_i \geq \sum_{i=1}^{\ell} \varphi_i \overset{\text{Lem. 10}}{\geq} \frac{1}{c}\left(1 - \frac{1}{e^c}\right)\gamma h(S). \tag{19}$$

By monotonicity of g, we have

$$g(S^c) \leq g(E) = g(H_k \cup H_k^c) = \sum_{i:e_i^* \in H_k} g(e_i^* \mid H_{i-1} \cup H_k^c) + g(H_k^c). \tag{20}$$

Further, using $G_{i-1} \subseteq H_{i-1} \cup H_k^c$ and the definition of the algorithm, we derive

$$\begin{aligned} g(S^c) &\overset{(20)}{\leq} \sum_{i:e_i^* \in H_k} g(e_i^* \mid H_{i-1} \cup H_k^c) + g(H_k^c) \\ &\overset{(6)}{\leq} \gamma^{-1} \sum_{i:e_i^* \in H_k} g(e_i^* \mid G_{i-1}) + g(H_k^c) \\ &\overset{\text{alg.}}{\leq} \gamma^{-1} \sum_{i:e_i^* \in H_k} h(e_i^* \mid H_{i-1}) + g(H_k^c) \\ &= \gamma^{-1} h(H_k) + g(H_k^c) \\ &\overset{\gamma \leq 1}{\leq} \gamma^{-1} f(H_k). \end{aligned} \tag{21}$$

Together with (19), we obtain the desired approximation (18)

$$\begin{aligned} f(S) = h(S) + g(S^c) &\overset{(19),(21)}{\leq} \gamma^{-1} c\left(\frac{e^c}{e^c - 1}\right) f(H_k) + \gamma^{-1} f(H_k) \\ &= \gamma^{-1}\left(1 + c\frac{e^c}{e^c - 1}\right) f(H_k). \end{aligned}$$

□

We now show the lower bound on the competitive ratio for the case that g and h are submodular with curvature c. This proves the first part of Theorem 4.

Proposition 11 *For $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ monotone submodular functions with curvature $c \in [0, 1]$ no deterministic algorithm has a better competitive ratio than $1 + \frac{c}{2-c}$.*

Proof To show the lower bound of $1 + \frac{c}{2-c}$, consider the following instance. Let $n := |E|$, $c \in [0, 1]$ and fix an element $e^* \in E$. We define

$$h(S) := g(S) := \begin{cases} n - 1 + (1 - c)|S \setminus \{e^*\}| & \text{if } e^* \in S, \\ |S| & \text{otherwise.} \end{cases}$$

The marginal increase of the element e^* is $h(e^* \mid S) = n-1-c|S|$ if $e^* \notin S$ and $h(e^* \mid S) = 0$ otherwise, implying $h(e^* \mid S) \geq h(e^* \mid T)$ for $S \subseteq T \subseteq E$. The marginal increase of every other element $e \neq e^*$ is $h(e \mid S) = 1$ if $e^* \notin S$ and $h(e \mid S) = 1 - c$ otherwise, implying $h(e \mid S) \geq h(e \mid T)$ for all $S \subseteq T \subseteq E$. Together, we have that h and g are submodular. Thus, their curvature is given by (1) as

$$1 - \min_{e \in E} \frac{h(e \mid E \backslash \{e\})}{h(\{e\})} = 1 - \min \left\{ \frac{(1-c)(n-1)}{n-1}, \frac{1-c}{1} \right\} = c.$$

We now show that no solution π to this instance is better than $\frac{2(n-1)}{(2-c)(n-1)+c}$-competitive. For $n \to \infty$ we then obtain the desired lower bound of $\frac{2}{2-c} = 1 + \frac{c}{2-c}$. We observe that the optimum value of $f(S) = h(S) + g(S^c)$ with $|S| = 1$ is given by

$$f(\{e^*\}) = h(\{e^*\}) + g(E \backslash \{e^*\}) = n - 1 + n - 1 = 2(n-1),$$

since, for $e \neq e^*$, we have

$$\begin{aligned} f(\{e\}) = h(\{e\}) + g(E \backslash \{e\}) &= 1 + n - 1 + (1-c)(n-2) \\ &= 2n - cn - 2 + 2c = (2-c)(n-1) + c. \end{aligned}$$

Similarly, for $|S| = n - 1$ the optimum value is

$$f(E \backslash \{e^*\}) = h(E \backslash \{e^*\}) + g(\{e^*\}) = n - 1 + n - 1 = 2(n-1),$$

since, for $e \neq e^*$, we have

$$f(E \backslash \{e\}) = h(E \backslash \{e\}) + g(\{e\}) = n - 1 + (1-c)(n-2) + 1 = (2-c)(n-1) + c.$$

Fixing π, we distinguish two cases. If the first element in π is $e \neq e^*$, we consider the resulting solution of size 1 which has value $f(\{e\}) = (2-c)(n-1) + c$. Therefore, in this case π has competitive ratio at least $\frac{2(n-1)}{(2-c)(n-1)+c}$. If the last element in π is $e \neq e^*$, we consider the solution of size $n-1$ which has value $f(E \backslash \{e\}) = (2-c)(n-1) + c$. Thus, also in this case the competitive ratio is at least $\frac{2(n-1)}{(2-c)(n-1)+c}$. □

We also give a lower bound in terms of the generic submodularity ratio; this proves the second part of Theorem 4.

Proposition 12 *For monotone functions $g, h \colon 2^E \to \mathbb{R}_{\geq 0}$ with generic submodularity ratio $\gamma \in (0, 1]$ no deterministic algorithm has a better competitive ratio than $\frac{2}{3} + \frac{1}{3\gamma}$.*

Proof To show the lower bound of $\frac{2}{3} + \frac{1}{3}\gamma^{-1}$, consider the following instance. Let $E = \{a, b, c\}$, $g, h \colon 2^E \to \mathbb{R}_{\geq 0}$ and $g \coloneqq h$ with

$$\begin{aligned} &h(\emptyset) = 0, \quad h(\{a\}) = h(\{b\}) = h(\{c\}) = 1, \quad h(\{a, b\}) = h(\{b, c\}) = 2, \\ &h(\{a, c\}) = 1 + \gamma^{-1} \quad \text{and} \quad h(\{a, b, c\}) = 2 + \gamma^{-1}. \end{aligned}$$

By definition of h, we have $h(b \mid S) = 1$ for all $S \subseteq \{a, c\}$. For $e, e' \in \{a, c\}$, $e \neq e'$ and $S \subseteq E \backslash \{e\}$ we have $h(e \mid S) = \gamma^{-1}$ if $S = \{e'\}$ or $S = \{b, e'\}$ and $h(e \mid S) = 1$ else. Thus, for $e \in \{a, c\}$ we have

$$\min_{A, B \subseteq E : h(e \mid A \cup B) > 0} \frac{h(e \mid A)}{h(e \mid A \cup B)} = \min \left\{ \frac{1}{1}, \frac{\gamma^{-1}}{\gamma^{-1}}, \frac{1}{\gamma^{-1}} \right\} = \gamma$$

and

$$\min_{A, B \subseteq E : h(b \mid A \cup B) > 0} \frac{h(b \mid A)}{h(b \mid A \cup B)} = \min \left\{ \frac{1}{1} \right\} = 1$$

which yields that the generic submodularity ratio of g and h is γ.

We show that no solution π is better than $\frac{2}{3}+\frac{1}{3}\gamma^{-1}$-competitive. The optimum value of size 1 is $f(\{b\}) = h(\{b\}) + g(\{a,c\}) = 1+1+\gamma^{-1} = 2+\gamma^{-1}$. The optimum value of size 2 is $f(\{a,c\}) = h(\{a,c\}) + g(\{b\}) = 1+\gamma^{-1}+1 = 2+\gamma^{-1}$. If the first element in π is $e \neq b$, with $e' \in \{a,c\}, e' \neq e$ we compute the value of the solution of size 1, which is $f(\{e\}) = h(\{e\}) + g(\{b,e'\}) = 1+2 = 3$. Likewise, if the last element in π is $e \neq b$, with $e' \in \{a,c\}, e' \neq e$, we compute the value of the solution of size 2, which is $f(\{b,e'\}) = h(\{b,e'\}) + g(\{e\}) = 2+1 = 3$. This yields in both cases that the competitive ratio is at least $\frac{2+\gamma^{-1}}{3} = \frac{2}{3}+\frac{1}{3}\gamma^{-1}$. □

4 Gross substitutes functions

The gross substitutes property originates from the economics literature [20]. In this context, given a value function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ and a modular price function $p\colon 2^E \to \mathbb{R}$, one is interested in sets S maximizing the utility $f(S) - p(S)$. A function f is gross substitutes if, for every price function and for every subset S maximizing the utility before increasing some of those prices, all elements of S with non-increased price are still part of a utility-maximizing set after the prices are increased.

Definition 3 *(Kelso and Crawford [20])* A function $f\colon 2^E \to \mathbb{R}$ is *gross substitutes* if for all modular price functions $p, p'\colon 2^E \to \mathbb{R}$ with $p \leq p'$ and $S \in \arg\max_{S\subseteq E}\{f(S) - p(S)\}$ there is $S' \in \arg\max_{S\subseteq E}\{f(S) - p'(S)\}$ with $S \cap \{e \mid p(\{e\}) = p'(\{e\})\} \subseteq S'$.

Paes Leme [8] surveys several equivalent ways to characterize gross substitutes functions. We will work with the following characterization.

Lemma 13 (Paes Leme [8]) *A monotone set function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ is* gross substitutes *if and only if it is submodular and for all $A, B \subseteq E$ with $A \cap B = \emptyset$, $|B| \geq 2$ and all $b \in B$ there is $b' \in B\backslash\{b\}$ with*

$$f(b \mid A) + f(B\backslash\{b\} \mid A) \leq f(b' \mid A) + f(B\backslash\{b'\} \mid A).$$

The greedy algorithm is optimal for the incremental maximization problem with gross substitutes objective function. Together with Proposition 5 we obtain the following result for the randomized algorithm discussed in Sect. 2 that returns uniformly at random the greedy order with respect to h or the reverse greedy order with respect to g.

Corollary 14 *For $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ monotone and gross substitutes the randomized algorithm is 2-competitive.*

However, we obtain the same bound for gross substitutes functions with the deterministic double-greedy algorithm. To show this, we need the following observation, which even holds for submodular functions. Recall that $f(S) = h(S) + g(S^c)$. Again, we assume in the following that g and h are normalized. This is without loss of generality by (5), since the shifted functions g' and h' remain monotone and gross substitutes or submodular.

Lemma 15 *For $g, h\colon 2^E \to \mathbb{R}_{\geq 0}$ submodular and $\prec \in \{\leq, <\}$, the double-greedy algorithm ensures that*

$$g(E) = f(H_0) \leq \cdots \leq f(H_n) = f(G_n^c) \geq \cdots \geq f(G_0^c) = h(E).$$

Proof By definition of f, we have $f(H_0) = f(\emptyset) = h(\emptyset) + g(E) = g(E)$ and, likewise, we have $f(G_0^{\mathrm{c}}) = h(E)$.

To show $f(H_{i-1}) \leq f(H_i)$ for $1 \leq i \leq n$, it suffices to consider an iteration of the algorithm in which we obtain H_i by adding e_i^* to H_{i-1}, since in every other iteration $H_{i-1} = H_i$ holds, implying trivially $f(H_{i-1}) = f(H_i)$. Submodularity of g and $G_{i-1} \subseteq H_i^{\mathrm{c}}$ imply $g(e_i^* \mid H_i^{\mathrm{c}}) \leq g(e_i^* \mid G_{i-1})$. Since e_i^* is added to H_{i-1}, we have $g(e_i^* \mid G_{i-1}) \prec' h(e_i^* \mid H_{i-1})$ for $\prec' \in \{\leq, <\}$ with $\prec' \neq \prec$. Together, we obtain

$$\begin{aligned} f(e_i^* \mid H_{i-1}) &= f(H_i) - f(H_{i-1}) \\ &= h(H_i) - h(H_{i-1}) + g(H_i^{\mathrm{c}}) - g(H_{i-1}^{\mathrm{c}}) \\ &= h(e_i^* \mid H_{i-1}) - g(e_i^* \mid H_i^{\mathrm{c}}) \\ &\geq g(e_i^* \mid G_{i-1}) - g(e_i^* \mid H_i^{\mathrm{c}}) \\ &\geq 0. \end{aligned}$$

The inequalities $f(G_{i-1}^{\mathrm{c}}) \leq f(G_i^{\mathrm{c}})$ for $1 \leq i \leq n$ follow by symmetry, as the algorithm for $g' = h$, $h' = g$ and $\prec'$ computes the reversed ordering. □

With this, we are able to prove our second main result.

Theorem 3 *For monotone and gross substitutes functions $g, h : 2^E \to \mathbb{R}_{\geq 0}$ the double-greedy algorithm is 2-competitive.*

Proof Fix $k \leq n$ and let $S \subseteq E$ with $|S| \leq |H_k|$ elements. By Proposition 7 it suffices to show that $2f(H_k) \geq f(S) = h(S) + g(S^{\mathrm{c}})$. By Lemma 15 and monotonicity of g we have that $f(H_k) \geq g(E) \geq g(S^{\mathrm{c}})$. Therefore, it is left to show that $f(H_k) \geq h(S)$. By monotonicity of h, we have for all $i \leq n$ that

$$\begin{aligned} h(S) &\leq h(S \cup H_i) \\ &= h(S \mid H_i) + h(H_i) \\ &\leq h(S \mid H_i) + \sum_{j : e_j^* \in H_i} h(e_j^* \mid H_{j-1}) + \sum_{j : e_j^* \in G_i} h(e_j^* \mid H_{j-1}) \\ &= h(S \mid H_i) + \sum_{j=1}^{i} h(e_j^* \mid H_{j-1}). \end{aligned} \tag{22}$$

Let $i(S) \in \mathbb{N}$ smallest such that $H_{i(S)} \not\subseteq S$ or $G_{i(S)} \cap S \neq \emptyset$ and $n+1$ if $S = H_n$, i.e., iteration $i(S)$ is the first iteration in which the choice of the algorithm is no longer consistent with first including the elements of S. In the following, we show that we can modify S iteratively, either removing $e_{i(S)}^*$ from S if $e_{i(S)}^* \in G_{i(S)}$ or exchanging $e_{i(S)}^*$ with an element in S if $e_{i(S)}^* \in H_{i(S)}$ without increasing the bound on $h(S)$ derived in (22) for $i = i(S) - 1$, until $S = H_\ell$ for some $\ell \leq k$. □

Claim 1 Either $S = H_{i(S)-1}$ or there is S' with $i(S) < i(S')$, $|S'| \leq |H_k|$, and

$$h(S \mid H_{i(S)-1}) + \sum_{j=1}^{i(S)-1} h(e_j^* \mid H_{j-1}) \leq h(S' \mid H_{i(S')-1}) + \sum_{j=1}^{i(S')-1} h(e_j^* \mid H_{j-1}). \tag{23}$$

We can use this claim to prove the theorem as follows. Since we have $i(S) < i(S') \leq n$, after applying the claim iteratively with the resulting S' as next S finitely many times, we

obtain $S' = H_\ell$ for $\ell := i(S') - 1$. Since $|H_\ell| = |S'| \leq |H_k|$, we have $\ell \leq k$. Chaining the inequalities (23) for S and S' iteratively, we obtain that

$$h(S) \overset{(22)}{\leq} h(S \mid H_{i(S)-1}) + \sum_{j=1}^{i(S)-1} h(e_j^* \mid H_{j-1}) \overset{(23)}{\leq} h(H_\ell \mid H_\ell) + \sum_{j=1}^{\ell} h(e_j^* \mid H_{j-1}). \quad (24)$$

Application of Lemma 8 concludes the proof of the theorem

$$h(S) \overset{(24)}{\leq} \sum_{j=1}^{\ell} h(e_j^* \mid H_{j-1}) \leq \sum_{j=1}^{k} \max\{h(e_j^* \mid H_{j-1}), g(e_j^* \mid G_{j-1})\} \overset{\text{Lem. 8}}{\leq} f(H_k).$$

It remains to prove the claim. First, we observe that, for $i \leq i'$ and $S \subseteq E$, we have

$$\begin{aligned} h(S \mid H_i) &= h(S \cup H_i) - (h(H_{i'}) - h(H_{i'} \mid H_i)) \\ &\leq h(S \cup H_{i'}) - h(H_{i'}) + \sum_{i < j \leq i' : e_j^* \in H_{i'}} h(e_j^* \mid H_{j-1}) \\ &\leq h(S \mid H_{i'}) + \sum_{j=i+1}^{i'} h(e_j^* \mid H_{j-1}). \end{aligned} \quad (25)$$

Thus, we obtain for $i \leq i'$ and $S \subseteq E$ that

$$h(S \mid H_i) + \sum_{j=1}^{i} h(e_j^* \mid H_{j-1}) \overset{(25)}{\leq} h(S \mid H_{i'}) + \sum_{j=1}^{i'} h(e_j^* \mid H_{j-1}). \quad (26)$$

Now, let $i := i(S)$ and $S \neq H_{i-1}$ which implies $H_{i-1} \subsetneq S \subseteq G_{i-1}^c$. We consider two cases. First, assume that $e_i^* \in H_i$. Then, by $G_i = G_{i-1}$ and definition of $i(S)$, e_i^* is the only element in $H_i \backslash S$. Together, $H_{i-1} \subsetneq S$ and $e_i^* \notin H_{i-1}$ imply that $B := (S \backslash H_{i-1}) \cup \{e_i^*\}$ contains at least 2 elements. Thus, we can apply Lemma 13 with $b := e_i^*$ and $A := H_{i-1}$ yielding the existence of $b' \in S \backslash H_{i-1}$ with

$$\begin{aligned} h(e_i^* \mid H_{i-1}) + h(S \backslash H_{i-1} \mid H_{i-1}) \\ \leq h(b' \mid H_{i-1}) + h(S \backslash H_{i-1} \cup \{e_i^*\} \backslash \{b'\} \mid H_{i-1}). \end{aligned} \quad (27)$$

Since $S \subseteq G_{i-1}^c$ and $e_i^* \notin G_{i-1} \cup H_{i-1}$, we have $B \subseteq E \backslash (H_{i-1} \cup G_{i-1})$. By the choice of the algorithm in iteration i, e_i^* maximizes $\max\{h(e_i^* \mid H_{i-1}), g(e_i^* \mid G_{i-1})\}$ in $B \subseteq E \backslash (H_{i-1} \cup G_{i-1})$. Further e_i^* is added to H_{i-1}, which implies $h(e_i^* \mid H_{i-1}) \geq g(e_i^* \mid G_{i-1})$, yielding

$$h(e_i^* \mid H_{i-1}) \geq h(b' \mid H_{i-1}). \quad (28)$$

Together, we obtain

$$h(S \mid H_{i-1}) = h(S \backslash H_{i-1} \mid H_{i-1}) \overset{(27),(28)}{\leq} h(S \cup \{e_i^*\} \backslash \{b'\} \mid H_{i-1}). \quad (29)$$

Setting $S' := S \cup \{e_i^*\} \backslash \{b'\}$, we have $H_i \subseteq S'$ by $H_{i-1} \subseteq S$ and $b' \notin H_i$. Because of $S \subseteq G_{i-1}^c$ and $e_i^* \notin G_{i-1}$, we have $S' \subseteq G_{i-1}^c = G_i^c$. Together, $H_i \subseteq S' \subseteq G_i^c$ implies $i(S') > i = i(S)$. In addition, we check $|S'| = |S| \leq |H_k|$. The proof of the claim follows in this case by inferring

$$h(S \mid H_{i-1}) + \sum_{j=1}^{i-1} h(e_j^* \mid H_{j-1}) \overset{(29)}{\leq} h(S' \mid H_{i-1}) + \sum_{j=1}^{i-1} h(e_j^* \mid H_{j-1})$$

$$\overset{(26)}{\leq} h(S' \mid H_{i(S')-1}) + \sum_{j=1}^{i(S')-1} h(e_j^* \mid H_{j-1}).$$

Now consider the case $e_i^* \in G_i$. Then, $H_i = H_{i-1} \subseteq S$ and the definition of $i(S)$ imply e_i^* is the only element in $G_i \cap S$. Consider $S' := S \backslash \{e_i^*\} \subseteq G_i^c$. Then, $H_i = H_{i-1} \subseteq S' \subseteq G_i^c$ implies $i(S') > i = i(S)$. In addition, we check $|S'| < |S| \leq |H_k|$. We have by $H_{i-1} \subseteq S'$ and submodularity of h that

$$h(e_i^* \mid S') \leq h(e_i^* \mid H_{i-1}). \tag{30}$$

Using $H_{i-1} \subseteq S'$ in the first step we can conclude the proof of the claim by inferring

$$\begin{aligned} h(S \mid H_{i-1}) + \sum_{j=1}^{i-1} h(e_j^* \mid H_{j-1}) &= h(e_i^* \mid S') + h(S' \mid H_{i-1}) + \sum_{j=1}^{i-1} h(e_j^* \mid H_{j-1}) \\ &\overset{(30)}{\leq} h(S' \mid H_{i-1}) + \sum_{j=1}^{i} h(e_j^* \mid H_{j-1}) \\ &\overset{(26)}{\leq} h(S' \mid H_{i(S')-1}) + \sum_{j=1}^{i(S')-1} h(e_j^* \mid H_{j-1}). \end{aligned}$$

□

Remark 1 The analysis of the double-greedy algorithm for gross substitutes functions is tight. Even for modular functions, the algorithm is not better than 2-competitive. Consider the instance $E = \{a, b\}$, $g, h \colon 2^E \to \mathbb{R}_{\geq 0}$ modular, given by $g(\{a\}) = h(\{a\}) = h(\{b\}) = 1$ and $g(\{b\}) = 0$, and $\prec \; = \; <$. Then, the double-greedy algorithm can choose $e_1^* = a$ computing $\pi = (a, b)$. The solution of size 1 has value $f(\{a\}) = h(\{a\}) + g(\{b\}) = 1$. But the optimum value for a solution of size 1 is $f(\{b\}) = h(\{b\}) + g(\{a\}) = 2$. Thus, the algorithm is no better than 2-competitive.

We finally state that no deterministic algorithm can be better than 5/4-competitive which proves the third part of Theorem 4.

Proposition 16 *For $g, h \colon 2^E \to \mathbb{R}_{\geq 0}$ monotone and gross substitutes functions no deterministic algorithm has a better competitive ratio than* 5/4.

Proof Consider the following set functions h and g on $E = \{a, b, c\}$. Let

$$h(\emptyset) = 0, \quad h(\{a\}) = 1, \quad h(\{b\}) = h(\{c\}) = h(\{b, c\}) = 2 \quad \text{and}$$
$$h(\{a, b\}) = h(\{a, c\}) = h(\{a, b, c\}) = 3.$$

Similarly, let

$$g(\emptyset) = 0, \quad g(\{b\}) = 1, \quad g(\{a\}) = g(\{c\}) = g(\{a, c\}) = 2 \quad \text{and}$$
$$g(\{a, b\}) = g(\{b, c\}) = g(\{a, b, c\}) = 3.$$

We show that both functions are gross substitutes. Since h and g differ only by swapping a and b, it suffices to consider h. To show submodularity of h, we check whether the marginal values for each $e \in E$ decrease, i.e., $h(e \mid S) \geq h(e \mid T)$ for $S \subseteq T \subseteq E$. For $e = a$, this follows by $h(a \mid S) = 1$ for $S \subseteq \{b, c\}$, and $h(a \mid S) = 0$ otherwise. For $e = b$, this follows by $h(b \mid S) = 2$ if $S = \emptyset$, $h(b \mid S) = 1$ if $S = \{a\}$, and $h(b \mid S) = 0$ else for $S \subseteq E$. By h being symmetric in b and c, we conclude submodularity of h.

Using that $h(\{a\}) + h(\{b, c\}) = 3 \leq 5 = h(\{b\}) + h(\{a, c\}) = h(\{c\}) + h(\{a, b\})$, we derive that for $B = \{a, b, c\}$, $A = \emptyset$ and each $e \in B$ there is $f \in B \backslash \{e\}$ with

$$h(e \mid A) + h(B \backslash \{e\} \mid A) \leq h(f \mid A) + h(B \backslash \{f\} \mid A).$$

In addition, observe that the condition in Lemma 13 for $|B| = 2$ is always fulfilled with equality. We can therefore conclude that h is gross substitutes.

Observe that the optimum value for $f(S) = h(S) + g(S^c)$ with $|S| = 1$ is $f(\{c\}) = 2+3 = 5$. With $|S| = 2$ the optimum value is $f(\{a, b\}) = 3 + 2 = 5$. We show that the competitive ratio ρ of all orderings π is at least 1.25. If π starts with a or b, the solution of size 1 has value $f(\{a\}) = h(\{a\}) + g(\{b, c\}) = 1+3 = 4$ or $f(\{b\}) = h(\{b\}) + g(\{a, c\}) = 2+2 = 4$, implying $\rho \geq \frac{5}{4}$ in both cases. If π ends with a or b, the solution of size 2 has value $f(\{b, c\}) = h(\{b, c\}) + g(\{a\}) = 2+2 = 4$ or $f(\{a, c\}) = h(\{a, c\}) + g(\{b\}) = 3+1 = 4$. Also, in those cases we have $\rho \geq \frac{5}{4}$. Since π cannot both start and end with c, we obtain $\rho \geq \frac{5}{4}$. □

5 Discussion

In this work, we introduced the incremental–decremental maximization framework and established several competitive ratios for randomized and deterministic algorithms. While our results provide new insights, some open questions remain.

Concerning lower bounds for the double-greedy algorithm, for monotone and submodular functions h and g, Theorem 1 establishes $(1 + \frac{e}{e-1})$-competitiveness. The algorithm cannot perform better than that, as shown in Proposition 20 in the appendix. We do not know whether the general upper bound of $\frac{1}{\gamma}(1 + c\frac{e^c}{e^c-1})$ for the setting where h and g have curvature c and generic submodularity ratio γ is tight and leave this question as an open problem.

For the special case that $g(E) = h(E)$ and both functions are monotone and submodular, the double-greedy algorithm guarantees a 2-competitive solution, which is optimal.

Proposition 17 *For monotone and submodular set functions $g, h \colon 2^E \to \mathbb{R}_{\geq 0}$ with $g(E) = h(E)$, the double-greedy algorithm is exactly 2-competitive.*

Proof Due to the monotonicity and the assumption that $g(E) = h(E)$, we have $f(S) \leq 2h(E)$ for every subset $S \subseteq E$, implying that $\mathrm{OPT}_k \leq 2h(E)$ for every k. Because of Lemma 15, the value of the double-greedy solution $S_k = \{e_1, \ldots, e_k\}$ of size k is at least $h(E)$ if $g(E) = h(E)$. All in all, we conclude that OPT_k is at most twice as much as the value $f(S_k)$ of the double-greedy solution of size k. According to Proposition 11, there is no algorithm which attains a better competitive ratio in this setting. □

It would be interesting to see similar results which exploit dependencies between h and g.

Another open question is whether Theorem 1 can be strengthened in settings where g and h have different curvatures $c_g \neq c_h$ and different generic submodularity ratios $\gamma_g \neq \gamma_h$. By defining $c := \max\{c_g, c_h\}$ and $\gamma := \min\{\gamma_g, \gamma_h\}$ one can adapt the proofs leading to Theorem 1 obtaining $\frac{1}{\gamma}(1 + c\frac{e^c}{e^c-1})$-competitiveness of the double-greedy algorithm. Note that the function space $\mathcal{C}$ in Proposition 7 is then given by all monotone set functions on the ground set E with curvature at most c and generic submodularity ratio at least γ.

While the simple randomized algorithm stated in Corollary 6 provides in some cases a better competitive analysis, it is outperformed by the double-greedy algorithm for submodular functions. Buchbinder et al. [13] introduced a randomized version of the double-greedy

algorithm improving the approximation factor for unconstrained submodular maximization from 3 to 2. Their improved analysis, however, does not directly lead to a better competitive ratio for a randomized version of the double-greedy for incremental–decremental maximization as the worst case ratio is not necessarily attained at the position where k equals the cardinality of a maximum set of f. We leave the design of randomized algorithms with better competitive ratio as an interesting problem for future research.

Finally, another open problem is to narrow or even close the gaps identified in Table 1 for deterministic algorithms, in particular, the gap between 2 and $1 + \frac{e}{e-1}$ for submodular h and g, as well as the gap between 1.25 and 2 in the gross substitutes setting.

Appendix A Unboundedness of the incremental maximization problem

We show that for submodular non-monotone functions, no deterministic incremental algorithm can achieve a bounded competitive ratio.

Proposition 18 *The competitive ratio of the deterministic incremental maximization problem with non-monotone submodular objective function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ is unbounded.*

Proof Let $n := |E|$. Fix $e^* \in E$ and consider the set function $f\colon 2^E \to \mathbb{R}_{\geq 0}$ for $0 < \varepsilon < 1$ given by

$$f(S) := \begin{cases} 1 & \text{if } e^* \in S, \\ |S| \cdot \varepsilon & \text{else.} \end{cases}$$

This function is submodular, since $f(e^* \mid S) = 1 - |S| \cdot \varepsilon$ for $e^* \notin S \subseteq E$ and for $e \in E$ with $e \neq e^*$ the increase is $f(e \mid S) = \varepsilon$ if $e^* \notin S$ and $f(e \mid S) = 0$ if $e^* \in S$. Thus, we have decreasing marginal gains in both cases, i.e., $f(e \mid A) \geq f(e \mid A \cup B)$ for $A, B \subseteq E$ and $e \in E \setminus B$.

Consider an algorithm that adds element e^* first. Thus, the solution of the algorithm is 1 for every k while the optimum value is $k\varepsilon$ for $k \geq \frac{1}{\varepsilon}$. On the other hand, an algorithm that does not add e^* first has value ε for $k = 1$ but the optimum solution is 1. For $\varepsilon := \frac{1}{\sqrt{n}}$ considering steps $k = 1$ and $k = n - 1$ one can see that all algorithms have competitive ratio of $\Omega(\sqrt{n})$ which is unbounded for $n \to \infty$. □

Appendix B Tight bound for the double-greedy algorithm

A well-known example for monotone submodular functions are coverage functions. They count how many elements are covered by a set of subsets of those elements.

Definition 4 Let $E \subseteq 2^U$ be a set of subsets of a finite ground set U. Then, $f\colon 2^E \to \mathbb{N}$ given by $f(S) := \left|\bigcup_{C \in S} C\right|$ for $S \subseteq E$ is a *coverage function*.

Submodularity of a coverage function f can be seen by observing that the marginal increase $f(C \mid S)$ of a set C on some set of subsets $S \subseteq E$ is exactly the number of elements in C that are not already covered by subsets in S. When more subsets are added to S, the set of elements already covered only grows. Therefore, the marginal increase of C on S decreases, when more subsets are added to S.

Proposition 19 *The competitive ratio of the double-greedy algorithm is exactly $(1 + \frac{e}{e-1})$ for h monotone and submodular and for g modular and monotone.*

Proof By Theorem 1 the competitive ratio is at most $\left(1+\frac{\mathrm{e}}{\mathrm{e}-1}\right)$. It is left to show that the algorithm is not ρ-competitive for any $\rho < \left(1+\frac{\mathrm{e}}{\mathrm{e}-1}\right)$.

To this end, consider the following instance. Let $U \coloneqq \{1,\dots,k\}^k$ be the ground set of k-tuples with entries $1,\dots,k$. For $i \in \{1,\dots,k\}$ we define sets $A_i \coloneqq \{1,\dots,k\}^{k-1} \times \{i\}$ to partition the ground set into k pairwise disjoint sets of equal size by fixing the last entry to i. We define sets $B_i \coloneqq \{1,\dots,k\}^{i-1} \times \{k\} \times \{1,\dots,k\}^{k-i}$ of the same size by fixing the i-th entry to k for $i \in \{1,\dots,k\}$. Note that the sets A_i and B_i all have k^{k-1} elements for $i \in \{1,\dots,k\}$. Let $h\colon 2^E \to \mathbb{N}$ be a coverage function on U with $E \coloneqq \{A_1,\dots,A_k,B_1,\dots,B_k\}$. Then, for $S \subseteq E$ the value $h(S) \coloneqq |\bigcup_{C\in S} C|$ is the number of elements covered by the sets in S. Since h is a coverage function, it is monotone and submodular. Let $g\colon 2^E \to \mathbb{N}$ be the modular function with $g(\{B_i\}) = h(B_i \mid \{B_1,\dots,B_{i-1}\})$ and $g(\{A_i\}) = 0$ for $i \in \{1,\dots,k\}$. Note that, $g(\{B_1,\dots B_k\}) = h(\{B_1,\dots,B_k\})$ by definition of g.

The optimum solution of size k for f is $S_k^* \coloneqq \{A_1,\dots,A_k\}$ with complement $(S_k^*)^{\mathrm{c}} \coloneqq \{B_1,\dots,B_k\}$. Since S_k^* covers the ground set completely and $(S_k^*)^{\mathrm{c}}$ covers all elements apart from the $(k-1)^k$ elements with no entry equal to k,

$$f(S_k^*) = h(S_k^*) + g((S_k^*)^{\mathrm{c}}) = k^k + h\left((S_k^*)^{\mathrm{c}}\right) = k^k + k^k - (k-1)^k = 2k^k - (k-1)^k.$$

On the other hand $B_1,\dots,B_k$ is a possible solution of the double-greedy algorithm of size k. By modifying the values of h and g by ε one can force the algorithm to resolve the tie-breaking accordingly and reach the same lower bound for $\varepsilon \to 0$. In the following, we choose how the ties are resolved by the algorithm for simplicity. Consider iteration $\ell \le k$ of the algorithm under the assumption that $H_{\ell-1} = \{B_1,\dots,B_{\ell-1}\}$ and $G_{\ell-1} = \emptyset$ are already computed. Then, for $i \in \{1,\dots,k\}$ and $j \in \{\ell,\dots,k\}$ the sets A_i and B_j cover exactly $(k-1)^{\ell-1}k^{k-\ell}$ elements which are not already covered by $H_{\ell-1}$. These are the elements $(x_1,\dots,x_k) \in \{1,\dots,k\}^k$ with $x_k = i$ for A_i and $x_j = k$ for B_j, respectively, and $x_r \neq k$ for $r \in \{1,\dots,\ell-1\}$. Therefore, B_ℓ maximizes $h(B_\ell \mid H_{\ell-1})$ in $H_{\ell-1}^{\mathrm{c}}$. Since $g(\{B_i\}) = h(B_i \mid \{B_1,\dots,B_{i-1}\}) = (k-1)^{i-1}k^{k-i}$ is decreasing in $i \in \{\ell,\dots,k\}$, B_ℓ also maximizes $g(\{B_\ell\})$ in $H_{\ell-1}^{\mathrm{c}}$. Because of $h(B_\ell \mid H_{\ell-1}) = g(\{B_\ell\})$ the algorithm can choose B_ℓ as the next element setting $H_\ell \coloneqq H_{\ell-1} \cup \{B_\ell\}$ and $G_\ell \coloneqq G_{\ell-1} = \emptyset$. Thus, $B_1,\dots,B_k$ is a possible solution of the double-greedy algorithm. In this case, since H_k covers all elements apart from the $(k-1)^k$ elements with no entry equal to k, we have with $g(H_k^{\mathrm{c}}) = 0$ that

$$f(H_k) = h(H_k) + g(H_k^{\mathrm{c}}) = k^k - (k-1)^k.$$

Thus, we conclude

$$\lim_{k\to\infty} \frac{f(S_k^*)}{f(H_k)} = \lim_{k\to\infty} \frac{2-\left(\frac{k-1}{k}\right)^k}{1-\left(\frac{k-1}{k}\right)^k} = \frac{2-\mathrm{e}^{-1}}{1-\mathrm{e}^{-1}} = \frac{2\mathrm{e}-1}{\mathrm{e}-1} = 1+\frac{\mathrm{e}}{\mathrm{e}-1},$$

which shows the competitive ratio of the algorithm is at least $1+\frac{\mathrm{e}}{\mathrm{e}-1}$ and therefore by Theorem 1 exactly $1+\frac{\mathrm{e}}{\mathrm{e}-1}$. □

Since modular functions are submodular, we obtain the following Corollary.

Corollary 20 *The competitive ratio of the double-greedy algorithm is exactly* $(1+\frac{\mathrm{e}}{\mathrm{e}-1})$ *for* h *and* g *monotone and submodular.*

Acknowledgements We are grateful to Martin Knaack, Jannik Matuschke, and David Weckbecker for helpful discussions on functions of bounded curvature, gross substitutes functions and incremental maximization, respectively. We also thank Júlia Baligács and Nils Mosis for an initial exchange about the topic.

Author Contributions All authors contributed equally.

Funding Open Access funding enabled and organized by Projekt DEAL. This work is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through subprojects A07 and A09 of CRC/TRR154.

Data Availability No datasets were generated or analysed during the current study.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Bernstein, A., Disser, Y., Groß, M., Himburg, S.: General bounds for incremental maximization. Math. Program. **191**(2), 953–979 (2022)
2. Disser, Y., Klimm, M., Schewior, K., Weckbecker, D.: Incremental maximization via continuization. In: Proceedings of the 50th International Colloquium on Automata, Languages and Programming (ICALP) **47**(1–54), 17 (2023)
3. Disser, Y., Weckbecker, D.: Unified greedy approximability beyond submodular maximization. SIAM J. Discret. Math. **38**(1), 348–379 (2024)
4. Disser, Y., Weckbecker, D.: Incremental maximization for a broad class of objectives. In: Proceedings of the 33rd Annual European Symposium on Algorithms (ESA) **92**(1–92), pp. 13 (2025)
5. Megiddo, N.: Optimal flows in networks with multiple sources and sinks. Math. Program. **7**(1), 97–107 (1974). https://doi.org/10.1007/BF01585506
6. Bian, A.A., Buhmann, J.M., Krause, A., Tschiatschek, S.: Guarantees for greedy maximization of non-submodular functions with applications. In: Proceedings of the 34th International Conference on Machine Learning (ICML), pp. 498–507 (2017)
7. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions—i. Math. Program. **14**, 265–294 (1978)
8. Paes Leme, R.: Gross substitutability: an algorithmic survey. Games Econom. Behav. **106**, 294–316 (2017)
9. Feige, U.: A threshold of $\ln n$ for approximating set cover. J. ACM **45**(4), 634–652 (1998)
10. Conforti, M., Cornuéjols, G.: Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. Discret. Appl. Math. **7**(3), 251–274 (1984)
11. Das, A., Kempe, D.: Approximate submodularity and its applications: subset selection, sparse approximation and dictionary selection. J. Mach. Learn. Res. **19**(3), 1–34 (2018)
12. Goemans, M.X., Unda, F.: Approximating incremental combinatorial optimization problems. In: Proceedings of the 20th/21st International Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. (APPROX/RANDOM), **81**, 6:1–6:14 (2017)
13. Buchbinder, N., Feldman, M., Naor, J., Schwartz, R.: A tight linear time (1/2)-approximation for unconstrained submodular maximization. SIAM J. Comput. **44**(5), 1384–1402 (2015)
14. Feige, U., Mirrokni, V.S., Vondrak, J.: Maximizing non-monotone submodular functions. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 461–471 (2007)
15. Chekuri, C., Vondrák, J., Zenklusen, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. SIAM J. Comput. **43**(6), 1831–1879 (2014)
16. Fadaei, S., Fazli, M., Safari, M.: Maximizing non-monotone submodular set functions subject to different constraints: combined algorithms. Opererat. Res. Lett. **39**(6), 447–451 (2011)
17. Kulik, A., Shachnai, H., Tamir, T.: Approximations for monotone and nonmonotone submodular maximization with knapsack constraints. Math. Oper. Res. **38**(4), 729–739 (2013)
18. Lee, J., Mirrokni, V.S., Nagarajan, V., Sviridenko, M.: Maximizing nonmonotone submodular functions under matroid or knapsack constraints. SIAM J. Discret. Math. **23**(4), 2053–2078 (2010)

19. Gong, S., Nong, Q., Sun, T., Fang, Q., Du, D., Shao, X.: Maximize a monotone function with a generic submodularity ratio. Theoret. Comput. Sci. **853**, 16–24 (2021)
20. Kelso Jr, A.S., Crawford, V.P.: Job matching, coalition formation, and gross substitutes. Econometrica J Econ Soc, 1483–1504 (1982)
21. Disser, Y., Klimm, M., Lutz, A., Strubberg, L.: Incremental–decremental maximization. In: Proceedings of the 23rd Workshop on Approximation and Online Algorithms (WAOA), pp. 97–110 (2025)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.