Weakly Submodular Maximization Beyond Cardinality Constraints: Does Randomization Help Greedy?

Lin Chen [1 2] **Moran Feldman** [3] **Amin Karbasi** [1 2]

Abstract

Submodular functions are a broad class of set functions that naturally arise in many machine learning applications. Due to their combinatorial structures, there has been a myriad of algorithms for maximizing such functions under various constraints. Unfortunately, once a function deviates from submodularity (even slightly), the known algorithms may perform arbitrarily poorly. Amending this issue, by obtaining approximation results for functions obeying properties that generalize submodularity, has been the focus of several recent works. One such class, known as weakly submodular functions, has received a lot of recent attention from the machine learning community due to its strong connections to restricted strong convexity and sparse reconstruction. In this paper, we prove that a randomized version of the greedy algorithm achieves an approximation ratio of $(1+1/\gamma)^{-2}$ for weakly submodular maximization subject to a general matroid constraint, where γ is a parameter measuring the distance from submodularity. To the best of our knowledge, this is the first algorithm with a non-trivial approximation guarantee for this constrained optimization problem. Moreover, our experimental results show that our proposed algorithm performs well in a variety of real-world problems, including regression, video summarization, splice site detection, and black-box interpretation.

1. Introduction

Motivated by the frequent appearances of submodular functions in both theoretical and practical settings, the last decade has seen a proliferation of works on maximization of submodular functions. In particular, algorithms for maximizing a submodular function subject to various constraints have found many applications in machine learning and data mining, including data summarization (Mirzasoleiman et al., 2016a; Wei et al., 2013), document summarization (Lin & Bilmes, 2010; 2011), sensor placement (Krause et al., 2008; Krause & Guestrin, 2005), network reconstruction (Chen et al., 2016; Gomez Rodriguez et al., 2010), crowd teaching (Singla et al., 2014), spread of influence (Kempe et al., 2003) and article recommendation (El-Arini & Guestrin, 2011; Mirzasoleiman et al., 2016b).

Despite the above mentioned abundance of settings which give raise to submodular functions, it has been observed that there are also many settings inducing functions that are *close* to submodular (in some sense), but not strictly submodular. Unfortunately, algorithms that have been developed for maximization of true submodular functions often fail miserably when given a function which is only close to submodular (Hassidim & Singer, 2017). This hurdle has motivated the development of algorithms whose guarantee degrades gracefully with the distance of the function from submodularity. In particular, such algorithms have been developed for functions that are: close to submodular under a distance measure known as the supermodular degree (Feige & Izsak, 2013; Feldman & Izsak, 2014; 2017), close to a submodular function up to a multiplicative factor (Horel & Singer, 2016), noisy versions of submodular functions under various noise models (Hassidim & Singer, 2017), almost submodular in the sense that they satisfy the submodularity inequality $f(A)+f(B) \geq f(A \cup B)+f(A \cap B)$ up to a fixed constant (Bateni et al., 2013) or belong to a class of functions known as hypergraph-r valuations which restricts the kinds of interplay between elements that can affect a function's values (Abraham et al., 2012).

A particularly important class of close to submodular functions, known as γ-weakly submodular functions (where γ is a parameter measuring the distance of the function from being submodular), has received a lot of attention from the machine learning community. Weakly submodular functions were originally introduced by Das & Kempe (2011), who showed that the standard greedy algorithm achieves a

Authors are listed in alphabetical order. [1]Yale Institute for Network Science, Yale University, New Haven, CT, USA [2]Department of Electrical Engineering, Yale University [3] Department of Mathematics and Computer Science, Open University of Israel, Ra'anana, Israel. Correspondence to: Lin Chen <lin.chen@yale.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

good approximation ratio of $1-e^{-\gamma}$ for the problem of maximizing such functions subject to a cardinality constraint. Further works developed more sophisticated algorithms for the same maximization problem and demonstrated a large repertoire of applications captured by it. For example, Elenberg et al. (2017) described a streaming algorithm for the above maximization problem and used it to get a faster algorithm for interpreting outputs of neural networks. By relying on previous works which showed that submodular functions can be maximized by faster versions of the standard greedy algorithm that are either stochastic or distributed (Mirzasoleiman et al., 2015; 2013), Khanna et al. (2017) showed that these faster versions of the greedy algorithm can also be used for maximizing weakly submodular functions. Finally, Qian et al. (Qian et al., 2016) leveraged weak submodularity in the design of an approach for the parallel Pareto optimization problem for subset selection.

To the best of our knowledge, all existing works regarding the maximization of γ-weakly submodular functions assume a simple cardinality constraint, and thus, cannot be applied to applications which require more involved constraints. In this paper we make a first step towards amending this situation. Specifically, we show that RESIDUAL RANDOM GREEDY (originally proposed by Buchbinder et al. (2014) for submodular maximization) yields—through a more involved analysis—the first non-trivial approximation ratio for maximizing a γ-weakly submodular function subject to a general matroid constraint.

The above result has two important implications. First, as explained above, it opens the door to more involved applications. The second implication is related to the fact that RESIDUAL RANDOM GREEDY can be viewed as a randomized version of the greedy algorithm. This makes it possible to view our analysis of RESIDUAL RANDOM GREEDY as an evidence that the standard greedy algorithm works most of time. In other words, we expect the greedy algorithm to produce a good approximation ratio on instances that are not specifically engineered to make it perform poorly. To see whether this is indeed the case, we have conducted four sets of experiments. The first set studies the linear regression problem on synthetic data, and the other sets correspond to real-world application scenarios and use real data (the three real-world application scenarios are video summarization, splice site detection and black-box interpretation of images). As expected, our experiments show that RESIDUAL RANDOM GREEDY and the greedy algorithm have comparable performance on real-world instances.

1.1. Preliminaries and Results

In this section we present the notation and definitions we use in this paper, including the definition of γ-weak submodularity. We then use these notation and definitions to present our result formally.

We say that a set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ over a ground set $\mathcal{N}$ is monotone if $f(A) \leq f(B)$ for every two sets $A \subseteq B \subseteq \mathcal{N}$. Furthermore, given two subsets $A, B \subseteq \mathcal{N}$, we denote by $f(B \mid A)$ the marginal contribution of adding B's elements to A. More formally, $f(B \mid A) = f(A \cup B) - f(A)$. In many cases the subset B in the above definition will be a singleton set $\{u\}$. In these cases we write, for simplicity, $f(u \mid A)$ instead of $f(\{u\} \mid A)$. Additionally, we occasionally use $A + u$ and $A - u$ as shorthands for the union $A \cup \{u\}$ and the expression $A \setminus \{u\}$, respectively.

Using this notation we can now define γ-weak submodularity as follows (this definition differs slightly from the original definition of γ-weakly submodular functions by Das & Kempe (2011). We discuss this in more detail later in this section). A set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ is γ-weakly submodular for some $\gamma \in (0, 1]$ if

$$\sum_{u \in B} f(u \mid A) \geq \gamma \cdot f(B \mid A) \qquad (1)$$

for every two sets $A, B \subseteq \mathcal{N}$.

Next, we would like to remind the reader of the formal definition of a matroid. Consider a ground set $\mathcal{N}$ and a non-empty collection $\mathcal{I} \subseteq 2^{\mathcal{N}}$ of subsets of $\mathcal{N}$. The pair $(\mathcal{N}, \mathcal{I})$ is a matroid if for every two sets $A, B \subseteq \mathcal{N}$:

- $A \subseteq B$ and $B \in \mathcal{I}$ imply $A \in \mathcal{I}$.
- $|A| < |B|$ and $B \in \mathcal{I}$ imply the existence of an element $u \in B \setminus A$ such that $A \cup \{u\} \in \mathcal{I}$.

Furthermore, the sets in $\mathcal{I}$ are called the *independent* sets of the matroid. Matroids are important because they capture many natural structures. For example, the set of forests of a graph form a matroid known as the graphical matroid of this graph. Consequently, the maximization of various set functions subject to a general matroid constraint has been studied extensively (see, for example, (Călinescu et al., 2011; Feldman & Izsak, 2014; Feldman et al., 2011)).

In this paper we are interested in the problem of maximizing a non-negative monotone γ-weakly submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ subject to a matroid $\mathcal{M} = (\mathcal{N}, \mathcal{I})$ constraint. In other words, we want to find an independent set of the matroid maximizing f. Our main result for this problem is given by the following theorem.

Theorem 1.1. *The* RESIDUAL RANDOM GREEDY *algorithm of Buchbinder et al. (Buchbinder et al., 2014) has an approximation ratio of at least $(1 + 1/\gamma)^{-2}$ for the problem of maximizing a non-negative monotone γ-weakly submodular function subject to a matroid constraint.*

Two remarks about this result are now in place. First, we would like to point out that RESIDUAL RANDOM GREEDY is quite efficient. In the analysis of such algorithms it is standard practice to assume that the algorithm has access to two oracles: a value oracle that given a set $S \subseteq \mathcal{N}$ returns the value of the objective function for that set, and an independence oracle that given S determines whether it is independent. Given such oracles, RESIDUAL RANDOM GREEDY requires only $O(nk)$ queries to each one of them, where n is the size of the ground set $\mathcal{N}$ and k is the rank of the matroid (*i.e.*, the size of the largest independent set in it). In the rest of this paper we often use n and k to denote their values as defined here.

Our second remark regarding Theorem 1.1 is related to the definition of γ-weak submodularity given above. As mentioned, this definition is slightly different from the original definition of γ-weakly submodular functions by Das & Kempe (2011). The original definition was weaker in the sense that it required Inequality (1) to hold only for small sets, *i.e.*, sets whose size is at most comparable to the size of the largest possible feasible solution. For the sake of keeping the definition as clean as possible, we dropped this extra complication from our definition of weak submodularity. However, when one employs algorithms for weak submodular optimization to solve real-world problems, it is often useful to have the weakest possible definition because this makes it more likely for the real-world objective function to fall into the definition. Thus, we would like to point out that Theorem 1.1 applies even when the objective function only obeys the following weaker definition (with respect to the matroid $\mathcal{M}$ defining the constraint). Interestingly, this weaker definition is even weaker than the original definition of Das & Kempe (2011) for weak submodularity. A set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ is $(\gamma, \mathcal{M})$-restricted weakly submodular for some $\gamma \in (0, 1]$ and matroid $\mathcal{M} = (\mathcal{N}, \mathcal{I})$ if

$$\sum_{u \in B} f(u \mid A) \geq \gamma \cdot f(B \mid A)$$

for every two sets $A, B \subseteq \mathcal{N}$ such that $A \cup B \in \mathcal{I}$.

2. Related Work

The study of the maximization of monotone submodular functions subject to a matroid constraint can be traced back to the 1970's. Nemhauser et al. (1978) and Fisher et al. (1978) proved that the standard greedy algorithm achieves approximation ratios of $1 - 1/e \approx 0.632$ and $1/2$ for this problem when the matroid is a uniform matroid and a general matroid, respectively. The approximation ratio for uniform matroid constraints was discovered, at roughly the same time, to be optimal (Nemhauser & Wolsey, 1978). However, the question regarding the optimality of the $(1/2)$-approximation algorithm for general matroid constraints remained open for many years. A decade ago, this question was finally solved by a celebrated result of Călinescu et al. (2011) who described a $(1 - 1/e)$-approximation algorithm for maximizing a monotone submodular function subject to a general matroid constraint. The result of Călinescu et al. (2011) proved that exactly the same approximation ratio can be achieved for the maximization of monotone submodular functions subject to uniform and general matroid constraints, which implies that in some sense general matroid constraints are no more difficult than uniform matroid constraints. Nevertheless, there is still a significant gap between the time complexities of the fastest algorithms known for the two types of constraints (Buchbinder et al., 2017; Mirzasoleiman et al., 2015), and closing this gap (or proving that it cannot be done) remains an open question.

The result of Călinescu et al. (2011) has motivated a long series of works on the maximization of non-monotone submodular functions subject to a matroid constraint (Buchbinder et al., 2014; Chekuri et al., 2014; Ene & Nguyen, 2016; Feldman et al., 2011; Gharan & Vondrák, 2011). The currently best algorithm of this kind achieves an approximation ratio of 0.385 (Buchbinder & Feldman, 2016) for general matroid constraints, and no better approximation guarantee is known for uniform matroid constraints. On the inapproximability side, it is known that no polynomial time algorithm can achieve approximation ratios better than 0.491 and 0.478 for the maximization of non-monotone submodular functions subject to uniform and general matroid constraints, respectively.

3. Algorithm

In this section we present the RESIDUAL RANDOM GREEDY algorithm, originally proposed by Buchbinder et al. (2014), and prove Theorem 1.1. The pseudocode of this algorithm is given as Algorithm 1. In this pseudocode we use the notation $\mathcal{M}/S$ to denote the matroid obtained from the input matroid $\mathcal{M}$ by contracting a set S. Informally, Algorithm 1 grows a solution S in k rounds, where each round consists of two steps. In the first step, the algorithm assigns to each element a weight which is equal to the marginal contribution of this element to the current solution S. Then, in the second step of the round, the algorithm finds a set M of maximum weight among all sets whose union with the current solution S is independent, and adds a uniformly random element from M to S. We begin the analysis of Algorithm 1 with the following simple observation.

Observation 3.1. *Algorithm 1 always outputs a feasible set while using $O(nk)$ value and independence oracle queries.*

Proof. The first part of the observation follows immediately from the properties of a matroid. Specifically, it follows from the fact that for any independent set S of size less than k there must exist a non-empty set M such that $S \cup M$ is a base, and moreover, for such a set M any subset of $S \cup M$

Algorithm 1 Residual Random Greedy for Matroids

```
1: Initialize: $S_0 \leftarrow \varnothing$.
2: for $i \leftarrow 1, 2, \ldots, k$ do
3:     Let $M_i$ be a base of $\mathcal{M}/S_{i-1}$ maximizing the sum $\sum_{u \in M_i} f(u \mid S_{i-1})$.
4:     Let $u_i$ be a uniformly random element from $M_i$.
5:     $S_i \leftarrow S_{i-1} + u_i$.
6: end for
7: Return $S_k$.
```

is independent.

To see why the second part of the observation holds, we observe that the only line of Algorithm 1 which requires access to the oracles is Line 3. This line can be viewed as having two parts. The first part defines a weight $f(u \mid S_{i-1})$ for every element of $\mathcal{N} \setminus S$ (which requires $O(n)$ value oracle queries), while the second part finds a maximum weight independent set in $\mathcal{M}/S_{i-1}$ subject to these weights (which requires $O(n)$ independence oracle queries when done using the greedy algorithm). Thus, we get that each execution of Line 3 requires $O(n)$ oracle queries, and the observation follows since this line is executed k times. □

In the rest of this section we analyze the approximation ratio of Algorithm 1. Let us denote by OPT a set which maximizes f among the independent sets of $\mathcal{M}$. Since f is monotone, we may assume that OPT is a base of $\mathcal{M}$. We need to construct, for every $0 \leq i \leq k$, a random set OPT_i for which $S_i \cup OPT_i$ is a base. For the construction we use the following lemma from Brualdi (1969).

Lemma 3.2. *If A and B are two bases of a matroid $\mathcal{M} = (\mathcal{N}, \mathcal{I})$, then there exists a one to one function $g : A \setminus B \to B \setminus A$ such that for every $u \in A \setminus B$, $(B + u) - g(u) \in \mathcal{I}$.*

For $i = 0$, we define $OPT_0 = OPT$. For $i > 0$, OPT_i is constructed recursively based on the algorithm's behavior. Assume that OPT_{i-1} is already constructed, and let $g_i \colon M_i \to OPT_{i-1}$ be a one to one function mapping every element $u \in M_i$ to an element of OPT_{i-1} in such a way that $S_{i-1} \cup OPT_{i-1} - g_i(u) + u$ is a base. Observe that the existence of such a function follows immediately from Lemma 3.2 since both $S_{i-1} \cup OPT_{i-1}$ and $S_{i-1} \cup M_i$ are bases of $\mathcal{M}$. We now set $OPT_i = OPT_{i-1} - g_i(u_i)$, which guarantees that $S_i \cup OPT_i$ is a base, as promised. For this construction to be useful, it is important that the choice of g_i (among the possibly multiple functions obeying the required properties) is done independently of the random choice of u_i, which guarantees that $g_i(u_i)$ is a uniformly random sample from OPT_{i-1}.

The next lemma proves a lower bound on the expected values of the sets we have constructed. Due to space constraints, the proof of this observation (and most other proofs in this section) has been **deferred to Appendix A** in the supplementary material. However, we note that this proof is similar to the proof of Lemma A.1 of (Elenberg et al., 2017), but the current lemma obtains a tighter bound.

Lemma 3.3. *For every $0 \leq i \leq k$, $\mathbb{E}[f(OPT_i)] \geq \left[1 - \left(\frac{i+1}{k+1}\right)^{\gamma}\right] \cdot f(OPT)$.*

The next observation gives a lower bound on the increase in $\mathbb{E}[f(S_i)]$ as a function of i. Note that this bound uses the sets $\{OPT_i\}_{i=0}^{k}$ that we have constructed above.

Observation 3.4. *For every $1 \leq i \leq k$, $\mathbb{E}[f(S_i)] \geq \mathbb{E}[f(S_{i-1})] + \gamma \cdot \frac{\mathbb{E}[f(OPT_{i-1} \cup S_{i-1})] - \mathbb{E}[f(S_{i-1})]}{k-i+1}$.*

Combining the last lemma and observation, we get the next corollary.

Corollary 3.5. *For every $1 \leq i \leq k$, $\mathbb{E}[f(S_i)] \geq \mathbb{E}[f(S_{i-1})] + \gamma \cdot \frac{\{1-[i/(k+1)]^{\gamma}\} \cdot f(OPT) - \mathbb{E}[f(S_{i-1})]}{k-i+1}$.*

We are now ready to prove an approximation ratio for Algorithm 1. The next theorem proves that the approximation ratio of Algorithm 1 can be smaller than the approximation ratio guaranteed by Theorem 1.1 only by a low order term of $O(k^{-1})$. In **Appendix B**, we show that this low order term can be dropped, which implies Theorem 1.1.

Theorem 3.6. *The approximation ratio of Algorithm 1 is at least $(1 + 1/\gamma)^{-2} - O(k^{-1})$.*

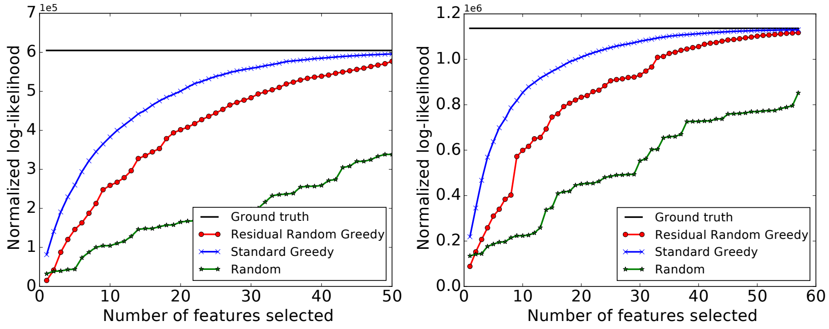
The proof of Theorem 3.6 consists mostly of solving the recursive formula given by Corollary 3.5.

4. Experiments

We conducted four sets of experiments. In the first set, we studied linear regression on synthetic data (Section 4.1), and in the other three, we investigated real-world application scenarios using real data. These scenarios include video summarization (Section 4.2), splice site detection (Section 4.3) and black-box interpretation for images (Section 4.4).

4.1. Linear Regression

In this set of experiments, we are given an $n \times p$ matrix $\boldsymbol{X}$ and a vector $\boldsymbol{y} \in \mathbb{R}^n$ which is a noisy version of the product of the matrix $\boldsymbol{X}$ and an unknown vector $\boldsymbol{\beta} \in \mathbb{R}^p$. More formally, $\boldsymbol{y} \triangleq \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where the coefficients of the noise vector $\boldsymbol{\varepsilon}$ are i.i.d. standard Gaussian random variables. In general, we are interested in the problem of, given such a matrix $\boldsymbol{X}$ and a vector $\boldsymbol{y}$, recovering β under the assumption that it is sparse in some sense. In the current set of experiment, we call a vector $\boldsymbol{\beta}$ sparse if and only if its support $\operatorname{supp}(\boldsymbol{\beta})$ is independent in some input matroid M. In other words, we want to find among the vectors whose support is independent in M, the vector $\boldsymbol{\beta}$ which is the most likely to be the vector which has been used to generate $\boldsymbol{y}$.



(a) Under a graphic matroid constraint (b) Under a partition matroid constraint

Figure 1. Normalized log-likelihood vs. the number of features selected in linear regression.

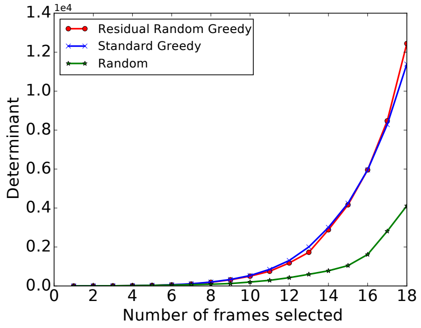


Figure 2. Determinant vs. number of frames selected in the video summarization problem.

The log-likelihood function of this problem for vectors is given by $l(\boldsymbol{\beta}) = -\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2 + C$, where C is a constant, and this yields the following log-likelihood function for support vectors $g(S) = \max_{\text{supp}(\boldsymbol{\beta}) \subseteq S} l(\boldsymbol{\beta}) = -\|\boldsymbol{y} - \boldsymbol{X}_S(\boldsymbol{X}_S^T\boldsymbol{X}_S)^{-1}\boldsymbol{X}_S^T\boldsymbol{y}\|^2 + C$, which was shown to be weakly submodular by Elenberg et al. (2016). Thus, our objective is to find a set S which is independent in M and approximately maximizes this weakly submodular function (given such a set S, one can calculate the vector $\boldsymbol{\beta}$ that we look for). Towards this goal, we have applied RESIDUAL RANDOM GREEDY to the matroid M and the normalized log-likelihood function $f(S) \triangleq g(S) - g(\varnothing)$ (we do not apply RESIDUAL RANDOM GREEDY directly to the log-likelihood function g since the last function is not guaranteed to be non-negative). We then compare the performance of RESIDUAL RANDOM GREEDY on this optimization problem with the following baselines.

- RANDOM. The RANDOM algorithm samples an independent set in an iterative manner. Throughout its execution, RANDOM maintains an independent set S, which is originally initialized to be the empty set. In each iteration, RANDOM adds to S a uniformly random element from the set of elements in $\mathcal{N}$ whose addition to S keeps S independent. This process continues until S becomes a base, at which point no more elements can be added to S.

- STANDARD GREEDY. Like the previous algorithm, the STANDARD GREEDY algorithm maintains an independent set S, which is originally initialized to be the empty set and grows iteratively. In each iteration, STANDARD GREEDY adds to S the element with the largest marginal contribution (with respect to the objective function) among the elements of $\mathcal{N}$ whose addition to S keeps S independent. Once more, the process terminates when S becomes a base.

Before we present the results obtained by this set of experiments, we would like to explain the way we used to generate the inputs for the experiments. We chose $n = 100$ and $p = 200$, and constructed each row of the $n \times p$ matrix $\boldsymbol{X}$ independently according to an autoregressive (AR) process with $\alpha = 0.5$ and noise variance $\sigma^2 = 10$ (in this generation process, each entry of the row is a function of the last few entries appearing before it in the row and a few random bits). The support of the vector $\boldsymbol{\beta}$ was chosen randomly using the above mentioned algorithm RANDOM (notice that this algorithm does not use the objective function, and thus, can be viewed as a way to sample an independent set from a matroid), and each non-zero value of $\boldsymbol{\beta}$ was assigned a uniformly random value from the set $\{-1, 1\}$. It remains to explain the way we used to construct the matroid M itself, which differs between the two experiments we conducted.

In one experiment we have used a graphic matroid M. To generate the graph underlying this matroid, we started with an empty graph over n vertices and added to it p random edges, where each edge was chosen independently and connected a uniformly random pair of distinct vertices. In the other experiment we have used a partition matroid M with 10 partitions, which we denote by $B_1, B_2, \ldots, B_{10}$. To generate this partition matroid we used a few steps. First, we uniformly sampled a random distribution out of the set of all possible distributions over the 10 partitions (*i.e.*, the sampled distribution is a uniformly random point from the standard 9-simplex). Then, we created p elements, and assigned each one of them to one of the partitions according to the above mentioned distribution. Finally, for every partition B_i, we sampled its capacity—*i.e.*, the maximum number of elements of this partition that can appear in an independent set—from the binomial distribution $B(|B_i|, 0.25)$.

The results of the two experiments are illustrated in Figure 1. The plots in this figure show how the normalized log-likelihood varies as the algorithms select more elements (also called "features") under the constraints corresponding to the above matroids. In both plots, the black line denotes the normalized log-likelihood achieved by the ground truth (*i.e.*, the vector $\boldsymbol{\beta}$ used to generate $\boldsymbol{y}$). We observe that RESIDUAL RANDOM GREEDY and STANDARD GREEDY yield comparable performance and both outperform RANDOM. In particular, when they terminate, the normalized likelihoods attained by RESIDUAL RANDOM GREEDY and STANDARD GREEDY are almost equal.

4.2. Video Summarization

In this application our objective is to pick a few frames from a video which summarize it (in some sense). One can formalize the problem of selecting such a summary as selecting a set of frames maximizing the Determinantal Point Process (DPP) objective function, which is a computationally efficient tool that favors subsets of elements with higher diversity (Kulesza et al., 2012). More formally, given an n frames video, we have represented each frame by a p-dimensional vector. Let $X \in \mathbb{R}^{n \times n}$ be the Gramian matrix of the n resulting vectors and the Gaussian kernel; *i.e.*, X_{ij} is the value of the Gaussian kernel between the i-th and j-th vectors. The DPP objective function is now given as the determinant function $f : 2^{[n]} \to \mathbb{R}$: $f(S) = \det(I + X_S)$, where X_S is the principal submatrix of X indexed by S. We note that the identity matrix was added here to the objective to make sure that the function f is monotone. Moreover, this function was shown to be weakly submodular with parameter $\min_{1 \leq k \leq n} \frac{k(\lambda_n - 1)}{(\prod_{j=1}^{k} \lambda_j) - 1}$, where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 1$ are the eigenvalues of $I + X$ (Bian et al., 2017). In light of the non-submodularity of the determinant function f, rather than optimize it directly, prior works considered its log, which is known to be submodular (Kulesza et al., 2012; Xu et al., 2015). This allows the use of standard submodular optimization techniques, but does not guarantee any approximation ratio for the original objective function. Fortunately, with the help of RESIDUAL RANDOM GREEDY, we can maximize the determinant function f directly and get a guaranteed approximation ratio.

The video that we have selected for this experiment lasts for roughly 7 minutes and a half, and we chose to created a summary of it by extracting one representative frame from every 25 seconds. In other words, the constraint on the allowed summarization is given by a partition matroid in which a set S of frames is independent (*i.e.*, belongs to $\mathcal{I}$) if and only if $|S \cap [25(i-1)+1, 25i]| \leq 1, \forall 1 \leq i \leq \lceil n/25 \rceil$. Given this constraint, the optimization problem that we need to solve is $\max_{S \in \mathcal{I}} f(S)$. Figure 2 illustrates the performance of RESIDUAL RANDOM GREEDY and the two benchmark algorithms when they are applied to this problem. The frames selected by the three algorithms are shown in Figure 3. Each algorithm selects one frame per 25 seconds, and the selected 18 frames are arranged in these images in chronological order from left to right and from top to bottom. It is quite easy to observe that both RESIDUAL RANDOM GREEDY and STANDARD GREEDY produce summaries with higher diversity than RANDOM. For example, the first two frames selected by RANDOM are about the same young lady in red, while RESIDUAL RANDOM GREEDY and STANDARD GREEDY choose one about the young lady and the other one about the TV show studio; and again, the 12-th and 13-th frames selected by RANDOM are both about a lady in black, while RESIDUAL RANDOM GREEDY and STANDARD GREEDY do not produce duplications, which allows them to cover other content. Comparing the outputs of RESIDUAL RANDOM GREEDY and STANDARD GREEDY is more subtle, but the result of RESIDUAL RANDOM GREEDY seems to be slightly better. The 10-th and 14-th frames selected by RESIDUAL RANDOM GREEDY show two participants that are not recognized by the other two summaries; in contrast, STANDARD GREEDY chooses five frames about TV show guests sitting behind a long blue desk in the studio, which reduces the diversity of the frames.

4.3. Splice Site Detection

An important problem in computational biology is the identification of true splice sites from similar decoy splice sites in nascent precursor messenger RNA (pre-mRNA) transcripts. Splice sites are nucleotide sequences that mark the beginnings and ends of introns (nucleotide sequences removed by RNA splicing during maturation of mRNA). In general, the two ends of an RNA sequence are known as the 5'-end and the 3'-end. In the case of introns, these ends are also known as the splice donor site and the splice acceptor site, respectively. We are interested in the problem of identifying splice donor and splice acceptor sites. In other words, given a sequence of nucleotides, we want to determine whether this sequence represents a splice donor/acceptor site. A splice donor site always includes the nucleotide sequence "GT" at its 5'-end, while a splice acceptor site has the sequence "AG" at its 3'-end. However, both kinds of sites include additional nucleotides whose identity should be taken into account when deciding whether a given sequence of nucleotides is a splice donor/acceptor site

The MEMset dataset provides instances of true and false splice donor/acceptor sites. We note that false splice donor/acceptor sites also include the compulsory "GT"/"AG" sequences, but differ from true sites in their other nucleotides. A detailed description of this dataset is presented in (Yeo & Burge, 2004). In this set of experiments, we used logistic regression on the MEMset dataset to determine the nucleotide values that have the largest influence on the categorization of splice sites into true and false sites. As a preprocessing step, we removed the consensus "GT" and "AG" sequences. Then, we considered the natural explana-

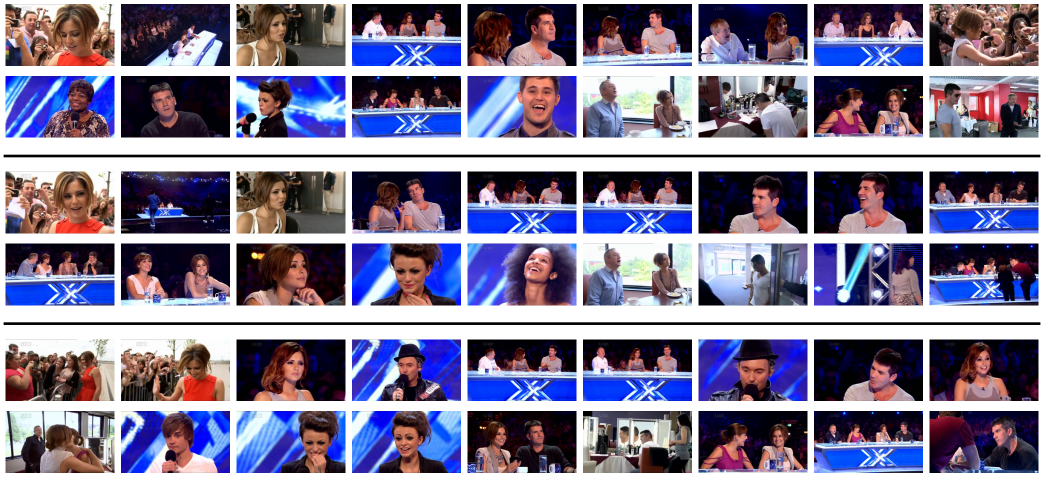
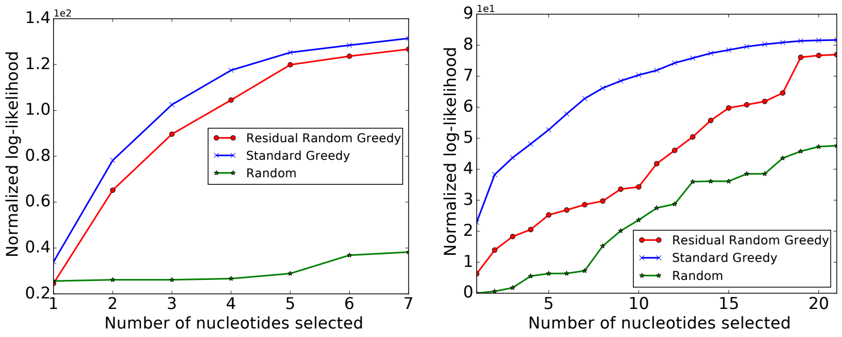


Figure 3. From top and bottom: frames selected by RESIDUAL RANDOM GREEDY, STANDARD GREEDY, and RANDOM as their video summaries.

tory variables for this problem, *i.e.*, a single variable taking the four values A, C, T and G for every nucleotide of the splice site. As these explanatory variables are categorical; we converted each of them into four binary variables via one-hot encoding. In other words, for each explanatory variable x_i (which takes values from {A, C, T, G}), we created four binary dummy variables $x'_{4i-3}, x'_{4i-2}, x'_{4i-1}, x'_{4i}$, where x'_{4i-3} (x'_{4i-2},x'_{4i-1} and x'_{4i}) takes the value one exactly when x_i is A (C, T and G, respectively). Given this encoding, a natural constraint is that at most one of the four binary variables can be set to one; which is a partition matroid constraint. Let us denote the j-th set of binary dummy variables and the corresponding outcome variable by $x'_{i,j}$ and y_j, respectively. As is standard in logistic regression, we assume that for all j, $\log\left(\frac{p_j}{1-p_j}\right) = \sum_i w_i x'_{i,j}$ and $y_j \mid \{x'_{i,j} : 1 \leq i \leq 4n\} \sim \text{Bernoulli}(p_j)$, where n is the total number of categorical explanatory variables (thus, we have $4n$ binary dummy variables in total). The log-likelihood function of logistic regression can now be written as $l(w) = \sum_j y_j(\sum_i w_i x'_{i,j}) - \log(1 + \exp(\sum_i w_i x'_{i,j}))$. As mentioned above, our objective is to find the set of nucleotide values that has the most influence on this log-likelihood function. Thus, the objective function we want to optimize is the normalized log-likelihood $f(S) \triangleq g(S) - g(\varnothing)$, where $g(S) = \max_{w:\text{supp}(w)\subseteq S} l(w)$. The weak submodularity of this objective function was shown in (Elenberg et al., 2016). In Figure 4, we present the result of applying RESIDUAL RANDOM GREEDY and the two benchmark algorithms mentioned above to this optimization problem. The ranks of the partition matroids for the donor and acceptor sites in Figure 4 are 7 and 21, respectively, because this is the number of nucleotides provided for each one of these kinds of sites by the MEMset dataset. One can note that STANDARD GREEDY and RESIDUAL RANDOM GREEDY exhibit comparable performance (especially at their termination point), and both consistently outperform RANDOM.

4.4. Black-Box Interpretation

In this set of experiments, we consider the problem of interpreting the predictions of black-box machine learning algorithms—*i.e.*, explaining the reasons for their prediction. Specifically, we follow the setting of (Elenberg et al., 2017; Ribeiro et al., 2016). Given an image I and a label l, the LIME framework (Ribeiro et al., 2016) outputs the likelihood that the image I has the label l. For example, the top five labels (in terms of the likelihood) assigned by the LIME framework to Figure 6(a) are *Bernese mountain dog* (with likelihood 0.44), *EntleBucher* (with likelihood 0.21), *Greater Swiss Mountain dog* (with likelihood 0.046), *Appenzeller* (with likelihood 0.033) and *Egyptian cat* (with likelihood 0.0044). Here we ask which parts of the image best explain the most likely label *Bernese mountain dog*; and let us denote this label by l_1 from now on. To this end, we applied the SLIC algorithm (Achanta et al., 2012) to the image, and this algorithm segmented the image into 25 superpixels (each superpixel is a tile of adjacent pixels of the image). Our task now is to select 10 superpixels that best explain the label l_1. We use $\mathcal{N}$ to denote a ground set consisting of all the superpixels. For any subset S of $\mathcal{N}$, let $I(S)$ denote the subimage where only superpixels in S are present, and let $f(S)$ be the likelihood that the subimage $I(S)$ has the label l_1. Using this notation, our task can be formulated as the following maximization problem: $\max_{|S|\leq k} f(S)$, where $k = 10$. We have applied RESIDUAL RANDOM GREEDY, STANDARD GREEDY and RANDOM to this optimization problem; and the superpixels selected by the three algorithms are visualized in Figures 6(b) to 6(d), respectively. We note that the set function $f(S)$ depends on the black-box machine learning algorithm, and thus, and may not be



(a) Splice donor site detection

(b) Splice acceptor site detection

Figure 4. Normalized log-likelihood vs. the number of nucleotides selected for splice donor and acceptor sites.

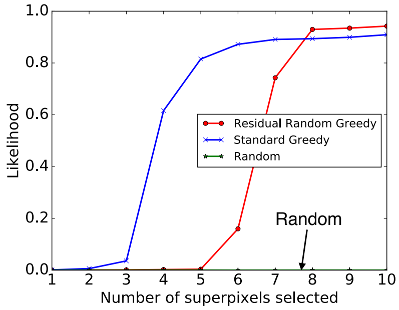
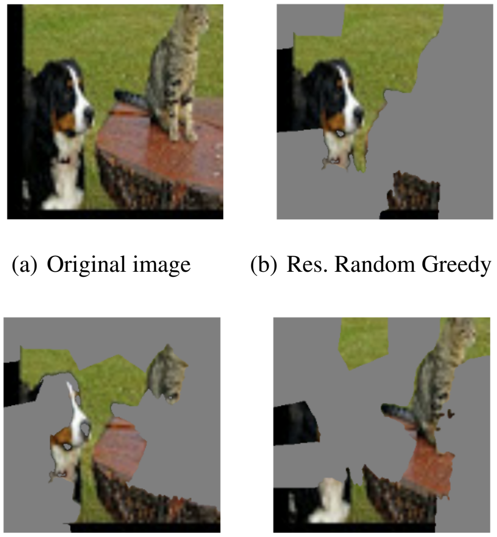


Figure 5. Likelihood that the subimage induced by the selected superpixels has the label vs. number of superpixels selected.



(a) Original image

(b) Res. Random Greedy

(c) Standard Greedy

(d) Random

Figure 6. Original image and visualization of the superpixels selected by the three algorithms to explain the label *Bernese mountain dog*.

weakly submodular, or even monotone, in general. Nevertheless, Residual Random Greedy and our benchmark algorithms still produce interesting results when used to optimize it.

Recall that the label that we try to explain is *Bernese mountain dog*. The superpixels selected by Residual Random Greedy (see Figure 6(b)) include all parts of the image that form the head of a Bernese mountain dog, while the superpixels selected by Standard Greedy (see Figure 6(c)) only cover the nose of the dog and a small portion of its body. Additionally, they also incorrectly include part of the cat. The performance of Random is the worst (see Figure 6(d)) as it mostly selects superpixels which are irrelevant to the dog. We also illustrate in Figure 5 the likelihood that the subimage induced by the selected superpixels has the label l_1 versus the number of superpixels selected. It can be observed that Residual Random Greedy outperforms Standard Greedy when ten superpixels are selected. It is also noteworthy to observe that the likelihood achieved by Random remains almost zero when the number of selected superpixels varies from 1 to 10, reaching only the value 4.39×10^{-4} at its highest point.

5. Conclusion

In this paper we have proved the first non-trivial approximation ratio for maximizing a γ-weakly submodular function subject to a general matroid constraint. Our result opens the door for new applications and also suggests that the greedy algorithm performs well in practice for this problem. Moreover, we were able to demonstrate experimentally, on multiple applications, this suggested good behavior of the greedy algorithm.

The most significant question that we leave open is whether the greedy algorithm has a good *provable* approximation ratio for the above problem. We note that this is not necessarily implied by the good practical behavior of the greedy algorithm. For example, on the closely related problem of maximizing a non-monotone submodular function, the greedy algorithm performs well in practice despite having an unbounded theoretical approximation ratio (Hassidim & Singer, 2017). Personally, we tend to believe that the greedy algorithm does have a good provable approximation ratio for the problem because we were unable to design any example on which the approximation ratio of the greedy algorithm is non-constant (for a constant γ). However, proving this formally is likely to require new ideas, and is thus, a very interesting area for future work.

Acknowledgements

MF was supported by Israel Science Foundation (ISF) grant number 1357/16 and AK was supported by AFOSR YIP (FA9550-18-1-0160).

References

Abraham, I., Babaioff, M., Dughmi, S., and Roughgarden, T. Combinatorial auctions with restricted complements. In *EC*, pp. 3–16, 2012.

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

Bateni, M., Hajiaghayi, M., and Zadimoghaddam, M. Submodular secretary problem and extensions. *ACM Transactions on Algorithms (TALG)*, 9(4):32, 2013.

Bian, A. A., Buhmann, J. M., Krause, A., and Tschiatschek, S. Guarantees for greedy maximization of non-submodular functions with applications. *CoRR*, abs/1703.02100, 2017.

Brualdi, R. A. Comments on bases in dependence structures. *Bull. of the Australian Math. Soc.*, 1(02):161–167, 1969.

Buchbinder, N. and Feldman, M. Constrained submodular maximization via a non-symmetric technique. *CoRR*, abs/1611.03253, 2016. URL http://arxiv.org/abs/1611.03253.

Buchbinder, N., Feldman, M., Naor, J., and Schwartz, R. Submodular maximization with cardinality constraints. In *SODA*, pp. 1433–1452, 2014.

Buchbinder, N., Feldman, M., and Schwartz, R. Comparing apples and oranges: Query tradeoff in submodular maximization. *Mathematics of Operations Research*, 42: 308–329, May 2017.

Călinescu, G., Chekuri, C., Pál, M., and Vondrák, J. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.

Chekuri, C., Vondrák, J., and Zenklusen, R. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM J. Comput.*, 43(6): 1831–1879, 2014.

Chen, L., Karbasi, A., and Crawford, F. W. Submodular variational inference for network reconstruction. *CoRR*, abs/1603.08616, 2016.

Das, A. and Kempe, D. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *ICML*, pp. 1057–1064, 2011.

El-Arini, K. and Guestrin, C. Beyond keyword search: Discovering relevant scientific literature. In *SIGKDD*, pp. 439–447, 2011.

Elenberg, E. R., Khanna, R., Dimakis, A. G., and Negahban, S. Restricted strong convexity implies weak submodularity. *CoRR*, abs/1612.00804, 2016.

Elenberg, E. R., Dimakis, A. G., Feldman, M., and Karbasi, A. Streaming weak submodularity: Interpreting neural networks on the fly. *CoRR*, abs/1703.02647, 2017.

Ene, A. and Nguyen, H. L. Constrained submodular maximization: Beyond 1/e. In *FOCS*, pp. 248–257, 2016.

Feige, U. and Izsak, R. Welfare maximization and the supermodular degree. In *ITCS*, pp. 247–256, 2013.

Feldman, M. and Izsak, R. Constrained monotone function maximization and the supermodular degree. In *APPROX*, pp. 160–175, 2014.

Feldman, M. and Izsak, R. Building a good team: Secretary problems and the supermodular degree. In *SODA*, pp. 1651–1670, 2017.

Feldman, M., Naor, J., and Schwartz, R. A unified continuous greedy algorithm for submodular maximization. In *FOCS*, pp. 570–579, 2011.

Fisher, M. L., Nemhauser, G. L., and Wolsey, L. A. An analysis of approximations for maximizing submodular set functions – II. *Mathematical Programming Study*, 8: 73–87, 1978.

Gharan, S. O. and Vondrák, J. Submodular maximization by simulated annealing. In *SODA*, pp. 1098–1116, 2011.

Gomez Rodriguez, M., Leskovec, J., and Krause, A. Inferring networks of diffusion and influence. In *SIGKDD*, pp. 1019–1028, 2010.

Hassidim, A. and Singer, Y. Submodular optimization under noise. In *COLT*, pp. 1069–1122, 2017.

Horel, T. and Singer, Y. Maximization of approximately submodular functions. In *NIPS*, pp. 3045–3053, 2016.

Kempe, D., Kleinberg, J., and Tardos, E. Maximizing the spread of influence through a social network. In *SIGKDD*, pp. 137–146, 2003.

Khanna, R., Elenberg, E. R., Dimakis, A. G., Negahban, S., and Ghosh, J. Scalable greedy feature selection via weak submodularity. *CoRR*, abs/1703.02723, 2017.

Krause, A. and Guestrin, C. Near-optimal nonmyopic value of information in graphical models. In *UAI*, pp. 5, 2005.

Krause, A., AjitSingh, and Guestrin, C. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.*, 9:235–284, January 2008.

Kulesza, A., Taskar, B., et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

Lin, H. and Bilmes, J. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL/HLT*, Los Angeles, CA, June 2010.

Lin, H. and Bilmes, J. A class of submodular functions for document summarization. In *HLT*, pp. 510–520, 2011.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pp. 2049–2057, 2013.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., and Krause, A. Lazier than lazy greedy. In *AAAI*, pp. 1812–1818, 2015.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization. *Journal of Machine Learning Research (JMLR)*, 2016a.

Mirzasoleiman, B., Zadimoghaddam, M., and Karbasi, A. Fast distributed submodular cover: Public-private data summarization. In *Advances in Neural Information Processing Systems*, pp. 3594–3602, 2016b.

Nemhauser, G. L. and Wolsey, L. A. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14:265–294, 1978.

Qian, C., Shi, J.-C., Yu, Y., Tang, K., and Zhou, Z.-H. Parallel pareto optimization for subset selection. In *IJCAI*, pp. 1939–1945, 2016.

Ribeiro, M. T., Singh, S., and Guestrin, C. Why should I trust you?: Explaining the predictions of any classifier. In *SIGKDD*, pp. 1135–1144, 2016.

Singla, A., Bogunovic, I., Bartók, G., Karbasi, A., and Krause, A. Near-optimally teaching the crowd to classify. In *ICML*, 2014.

Wei, K., Liu, Y., Kirchhoff, K., and Bilmes, J. Using document summarization techniques for speech data subset selection. In *HLT-NAACL*, pp. 721–726, 2013.

Xu, J., Mukherjee, L., Li, Y., Warner, J., Rehg, J. M., and Singh, V. Gaze-enabled egocentric video summarization via constrained submodular maximization. In *CVPR*, pp. 2235–2244, 2015.

Yeo, G. and Burge, C. B. Maximum entropy modeling of short sequence motifs with applications to rna splicing signals. *Journal of computational biology*, 11(2-3):377–394, 2004.