SIAM J. COMPUT.
Vol. 47, No. 3, pp. 703–754
© 2018 Society for Industrial and Applied Mathematics

# SUBMODULAR FUNCTIONS: LEARNABILITY, STRUCTURE, AND OPTIMIZATION*

MARIA-FLORINA BALCAN† AND NICHOLAS J. A. HARVEY‡

**Abstract.** Submodular functions are discrete functions that model laws of diminishing returns and enjoy numerous algorithmic applications. They have been used in many areas, including combinatorial optimization, machine learning, and economics. In this work we study submodular functions from a learning theoretic angle. We provide algorithms for learning submodular functions, as well as lower bounds on their learnability. In doing so, we uncover several novel structural results revealing ways in which submodular functions can be both surprisingly structured and surprisingly unstructured. We provide several concrete implications of our work in other domains including algorithmic game theory and combinatorial optimization. At a technical level, this research combines ideas from many areas, including learning theory (distributional learning and PAC-style analyses), combinatorics and optimization (matroids and submodular functions), and pseudorandomness (lossless expander graphs).

**Key words.** submodular functions, distributional learning, matroids, algorithmic game theory, optimization, PAC model, gross substitutes

**AMS subject classifications.** 68Q01, 68P01

**DOI.** 10.1137/120888909

## 1. Introduction.

Submodular functions are a discrete analogue of convex functions that enjoy numerous applications and have structural properties that can be exploited algorithmically. They arise naturally in the study of graphs, matroids, covering problems, facility location problems, etc., and they have been extensively studied in operations research and combinatorial optimization for many years [22]. More recently, submodular functions have become key concepts in other areas including machine learning, algorithmic game theory, and social sciences. For example, submodular functions have been used to model bidders' valuation functions in combinatorial auctions [41, 67, 20, 4, 95] and for solving several machine learning problems, including feature selection problems in graphical models [61] and various clustering problems [76].

In this work we use a learning theory perspective to uncover new structural properties of submodular functions. In addition to providing algorithms and lower bounds for learning submodular functions, we discuss numerous implications of our work in algorithmic game theory, economics, matroid theory, and combinatorial optimization.

One of our foremost contributions is to provide the first known results about learnability of submodular functions in a distributional (i.e., PAC-style) learning setting. Informally, such a setting has a fixed but unknown submodular function $f^*$ and

*Received by the editors August 22, 2012; accepted for publication (in revised form) February 20, 2018; published electronically May 31, 2018. A preliminary version of this paper appeared in the *Proceedings of the 43rd ACM Symposium on Theory of Computing* under the title "Learning Submodular Functions."

http://www.siam.org/journals/sicomp/47-3/88890.html

**Funding:** This work was supported in part by NSF grants CCF-0953192 and CCF-1101283, AFOSR grant FA9550-09-1-0538, ONR grant N00014-09-1-0751, an NSERC Discovery Grant, and a Microsoft Research Faculty Fellowship.

†School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 (ninamf@cs.cmu.edu).

‡Department of Computer Science, University of British Columbia, Vancouver, BC, V6T 1Z1, Canada (nickhar@cs.ubc.ca).

a fixed but unknown distribution over the domain of $f^*$. The goal is to design an efficient algorithm which provides a good approximation of $f^*$ with respect to that distribution, given only a small number of samples from the distribution.

Formally, let $[n] = \{1, \ldots, n\}$ denote a ground set of items and let $2^{[n]}$ be the power set of $[n]$. A function $f : 2^{[n]} \to \mathbb{R}$ is submodular if it satisfies

$$f(T \cup \{i\}) - f(T) \quad \leq \quad f(S \cup \{i\}) - f(S) \qquad \forall S \subseteq T \subseteq [n],\, i \in [n].$$

The goal is to output a function $f$ that, with probability $1 - \delta$ over the samples, is a good approximation of $f^*$ on most of the sets coming from the distribution. Here "most" means a $1 - \epsilon$ fraction and "good approximation" means that $f(S) \leq f^*(S) \leq \alpha \cdot f(S)$ for some approximation factor $\alpha$. We prove $\alpha = O(n^{1/2})$ upper and $\alpha = \tilde{\Omega}(n^{1/3})$ lower bounds on the approximation factor achievable when the algorithm receives only $\mathrm{poly}(n, 1/\epsilon, 1/\delta)$ examples from an arbitrary (fixed but unknown) distribution. We additionally provide a learning algorithm with constant approximation factor for the case that the underlying distribution is a product distribution. This is based on strong concentration properties of submodular functions, a topic that has independently been studied in prior work [40, Lemma 9], [14, Theorem 1.4], [97].

To prove the $\tilde{\Omega}(n^{1/3})$ lower bound for learning under arbitrary distributions, we construct a new family of matroids whose rank functions are fiendishly unstructured. Since matroid rank functions are submodular, this shows unexpected extremal properties of submodular functions and gives new insights into their complexity. This construction also provides a general tool for proving lower bounds in several areas where submodular functions arise. We derive and discuss such implications in two areas:

- *Algorithmic game theory and economics:* An important consequence of our construction is that matroid rank functions do not have a "sketch," i.e., a concise, approximate representation. As matroid rank functions are known to satisfy the gross substitutes property [75], our work implies that gross substitutes functions also do not have a concise, approximate representation. This provides a surprising answer to an open question in algorithmic game theory and economics [8], [9, section 6.2.1], [10, section 2.2].
- *Combinatorial optimization:* Many optimization problems involving submodular functions, such as submodular function minimization, are very well behaved and their optimal solutions have a rich structure. In contrast, we show that, for several other submodular optimization problems which have been considered recently in the literature, including submodular $s$-$t$ min cut and submodular vertex cover, their optimal solutions are very unstructured, in the sense that the optimal solutions do not have a succinct representation, or even a succinct, approximate representation.

Although our new family of matroids proves that matroid rank functions (and more generally submodular functions) are surprisingly unstructured, the concentration result for submodular functions shows that, in a different sense, matroid rank functions (and other sufficiently "smooth" submodular functions) are surprisingly structured.

Submodularity has been an increasingly useful tool in machine learning in recent years. For example, it has been used for feature selection problems in graphical models [61] and various clustering problems [76]. In fact, submodularity has been the topic of several tutorials and workshops at recent major conferences in machine learning [78, 62, 63, 79]. Nevertheless, our work is the first to use a learning theory

perspective to derive new structural results for submodular functions and related structures (including matroids), thereby yielding implications in many other areas. Our work also potentially has useful applications—our learning algorithms can be employed in many areas where submodular functions arise (e.g., medical decision making and economics). We discuss such applications in section 1.2. Furthermore, our work defines a new learning model for approximate distributional learning that could be useful for analyzing learnability of other interesting classes of real-valued functions. In fact, this model has already been used to analyze the learnability of several classes of set functions widely used in economics—see sections 1.1.2 and 8.1.

**1.1. Our results and techniques.** The central topic of this paper is proving new structural results for submodular functions, motivated by learnability considerations. In the following we provide a more detailed description of our results. For ease of exposition, we start by describing our new structural results, then we present our learning model and our learnability results within this model, and finally we describe implications of our results in various areas.

**1.1.1. New structural results.**

*A new matroid construction.* The first result in this paper is the construction of a family of submodular functions with interesting technical properties. These functions are the key ingredient in our lower bounds for learning submodular functions, inapproximability results for submodular optimization problems, and the nonexistence of succinct, approximate representations for gross substitutes functions.

Designing submodular functions directly is difficult because there is very little tangible structure to work with. It turns out to be more convenient to work with *matroids*,[1] because every matroid has an associated submodular function (its *rank function*) and because matroids are a very rich class of combinatorial objects with numerous well-understood properties.

Our goal is to find a collection of subsets of $[n]$ and two values $r_{\text{high}}$ and $r_{\text{low}}$ such that for *any* labeling of these subsets as either HIGH or LOW, we can construct a matroid for which each set labeled HIGH has rank value $r_{\text{high}}$ and each set labeled LOW has rank value $r_{\text{low}}$. We would like both the size of the collection and the ratio $r_{\text{high}}/r_{\text{low}}$ to be as large as possible.

Unfortunately existing matroid constructions can only achieve this goal with very weak parameters; for further discussion of existing matroids, see section 1.3. Our new matroid construction, which involves numerous technical steps, achieves this goal with the collection of size superpolynomial in $n$ and the ratio $r_{\text{high}}/r_{\text{low}} = \tilde{\Omega}(n^{1/3})$. This shows that matroid rank functions can be fiendishly unstructured—in our construction, knowing the value of the rank function on all but one of the sets in the collection does not determine the rank value on the remaining set, even to within a multiplicative factor $\tilde{\Omega}(n^{1/3})$.

More formally, let the collection of sets be $A_1, \ldots, A_k \subseteq [n]$ where each $|A_i| = r_{\text{high}}$. For every set of indices $B \subseteq \{1, \ldots, k\}$ there is a matroid $\mathbf{M}_B$ whose associated rank function $r_B : 2^{[n]} \to \mathbb{R}$ has the form

(1.1)
$$r_B(S) \;=\; \max\left\{ |I \cap S| : \begin{array}{r} \left|I \cap \bigcup_{j\in J} A_j\right| \le r_{\text{low}} \cdot |J| - \sum_{j\in J}|A_j| + \left|\bigcup_{j\in J} A_j\right| \\ \forall J \subseteq B,\ |J| < \tau \end{array} \right\}.$$

[1] For the reader unfamiliar with matroids, a brief introduction to them is given in section 2.2. For the present discussion, the only fact that we need about matroids is that the rank function of a matroid on $[n]$ is a submodular function on $2^{[n]}$.

We show that if the sets $A_i$ satisfy a strong *expansion* property, in the sense that any small collection of $A_i$ has small overlap, and the parameters $r_{\text{high}}, r_{\text{low}}, \tau$ are carefully chosen, then this function satisfies $r_B(A_i) = r_{\text{low}}$ whenever $i \in B$ and $r_B(A_i) = r_{\text{high}}$ whenever $i \notin B$.

*Concentration of submodular functions.* A major theme in probability theory is proving concentration bounds for a function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ under product distributions.[2] For example, when $f$ is linear, the Chernoff–Hoeffding bound is applicable. For arbitrary $f$, the McDiarmid inequality is applicable. The quality of these bounds also depends on the "smoothness" of $f$, which can be quantified using the Lipschitz constant $L := \max_{S,i} |f(S \cup \{i\}) - f(S)|$.

For a 1-Lipschitz function (i.e., $L = 1$), McDiarmid's inequality gives concentration comparable to that of a Gaussian random variable with standard deviation $\sqrt{n}$. For example, the probability that the value of $f$ is $\sqrt{n}$ less than its expectation is bounded above by a constant. Such a bound is quite weak when the expectation of $f$ is significantly less than $\sqrt{n}$, because it says that the probability of $f$ being negative is at most a constant, even though that probability is actually zero.

Several groups of researchers have independently shown that 1-Lipschitz, monotone, submodular functions are extremely tightly concentrated around their expected value. The quality of concentration is similar to Chernoff–Hoeffding bounds—importantly, it depends only on the expected value of the function and not on the dimension $n$. This phenomenon seems to have been first noticed by Hajiaghayi et al. [40], who used the self-bounding function technique of Boucheron, Lugosi, and Massart [11] to prove this sort of concentration for a particular subadditive function. Independently, Chekuri and Vondrák [14, Theorem 1.4] used the Fortuin–Kasteleyn–Ginibre (FKG) inequality to prove this sort of concentration for arbitrary nonnegative, monotone, submodular functions. A subsequent note of Vondrák [97] explains how the technique of Boucheron, Lugosi, and Massart easily yields such concentration for arbitrary nonnegative submodular functions (without requiring monotonicity). In section 3.2 we show that Talagrand's concentration inequality can also be used to prove such concentration for nonnegative, monotone, submodular functions.

*Approximate characterization of matroids.* Our new matroid construction described above can be viewed at a high level as saying that matroids can be surprisingly unstructured. One can pick numerous large regions of the matroid (namely, the sets $A_i$) and arbitrarily decide whether each region should have large rank or small rank. Thus the matroid's structure is very unconstrained.

Our next result shows that, in a different sense, a matroid's structure is actually very constrained. If one fixes any integer $k$ and looks at the rank values among all sets of size $k$, then those values are extremely tightly concentrated around their average—almost all sets of size $k$ have nearly the same rank value. Moreover, these averages are concave as a function of $k$. That is, there exists a concave function $h : [0, n] \to \mathbb{R}_{\geq 0}$ such that almost all sets $S$ have rank approximately $h(|S|)$.

This provides an interesting converse to the well-known fact that the function $f : 2^{[n]} \to \mathbb{R}$ defined by $f(S) = h(|S|)$ is a submodular function whenever $h : \mathbb{R} \to \mathbb{R}$ is concave. Our proof uses the aforementioned result on concentration for submodular functions under product distributions, and the multilinear extension [13] of submodular functions, which has been of great value in recent work.

[2] A random set $S \subseteq 2^{[n]}$ is said to have a *product distribution* if the events $i \in S$ and $j \in S$ are independent for every $i \neq j$.

### 1.1.2. Learning submodular functions.

*The learning model.* To study the learnability of submodular functions, we extend Valiant's classic PAC model [92], which captures settings where the learning goal is to predict the future based on past observations. The abbreviation PAC stands for "probably approximately correct." The PAC model, however, is primarily designed for learning *Boolean-valued functions*, such as linear threshold functions, decision trees, and low-depth circuits [92, 58]. For *real-valued functions*, it is more meaningful to change the model by ignoring small-magnitude errors in the predicted values. Our results on learning submodular functions are presented in this new model, which we call the probably mostly approximately correct (PMAC) model.

In this model, a learning algorithm is given a collection $\mathcal{S} = \{S_1, S_2, \ldots\}$ of polynomially many sets drawn independent and identically distributed (i.i.d.) from some fixed, but unknown, distribution $D$ over sets in $2^{[n]}$. There is also a fixed but unknown function $f^* : 2^{[n]} \to \mathbb{R}_+$, and the algorithm is given the value of $f^*$ at each set in $\mathcal{S}$. The goal is to design a polynomial-time algorithm that outputs a polynomial-time-evaluatable function $f$ such that, with large probability over $\mathcal{S}$, the set of sets for which $f$ is a good approximation for $f^*$ has large measure with respect to $D$. More formally,

$$\Pr_{S_1,S_2,\ldots\sim D}\left[\Pr_{S\sim D}\left[\, f(S) \le f^*(S) \le \alpha f(S)\,\right] \ge 1-\epsilon\right] \quad \ge \quad 1-\delta,$$

where $f$ is the output of the learning algorithm on inputs $\{\,(S_i, f^*(S_i))\,\}_{i=1,2,\ldots}$. The approximation factor $\alpha \ge 1$ allows for multiplicative error in the function values. Thus, whereas the PMAC model requires one to *approximate* the value of a function on a set of large measure and with high confidence, the traditional PAC model requires one to predict the value *exactly* on a set of large measure and with high confidence. The PAC model is the special case of our model with $\alpha = 1$.

An alternative approach for dealing with real-valued functions in learning theory is to consider other loss functions such as the squared-loss or the L1-loss. However, this approach does not distinguish between the case of having low error on most of the distribution and high error on just a few points, versus moderately high error everywhere. In comparison, the PMAC model allows for more fine-grained control with separate parameters for the amount and extent of errors, and in addition it allows for consideration of multiplicative error which is often more natural in this context. We discuss this further in section 1.3.

Within the PMAC model we prove several algorithmic and hardness results for learning submodular functions, as follows.

*Algorithm for product distributions.* Our first learning result concerns product distributions. This is a natural first step when studying learnability of various classes of functions, particularly when the class of functions has high complexity [53, 54, 69, 87]. By making use of the strong concentration results for monotone, submodular functions under product distributions, we show that if the underlying distribution is a product distribution, then sufficiently "smooth" (formally, 1-Lipschitz) submodular functions can be PMAC-learned with a constant approximation factor $\alpha$ by a very simple algorithm.

*Inapproximability for general distributions.* Although submodular functions that are 1-Lipschitz can be PMAC-learned with a constant approximation factor under product distributions, this result does not generalize to arbitrary distributions. By making use of our new matroid construction, we show that every algorithm for PMAC-learning monotone, submodular functions under arbitrary distributions must have approximation factor $\tilde{\Omega}(n^{1/3})$ even for constant $\epsilon$ and $\delta$, and even if the functions

are matroid rank functions. Moreover, this lower bound holds even if the algorithm knows the underlying distribution and it can adaptively query the given function at points of its choice.

*Algorithm for general distributions.* Our $\tilde{\Omega}(n^{1/3})$ inapproximability result for general distributions turns out to be close to optimal. We give an algorithm to PMAC-learn an arbitrary nonnegative, monotone, submodular function with approximation factor $O(\sqrt{n})$ for any $\epsilon$ and $\delta$ by using a number of samples $\tilde{O}(n/\epsilon \log(1/\delta))$.

This algorithm is based on a recent structural result which shows that any monotone, nonnegative, submodular function can be approximated within a factor of $\sqrt{n}$ on every point by the square root of a linear function [33]. We leverage this result to reduce the problem of PMAC-learning a submodular function to learning a linear separator in the usual PAC model. We remark that an improved structural result for any subclass of submodular functions would yield an improved analysis of our algorithm for that subclass. Moreover, the algorithmic approach we provide is quite robust and can be extended to handle more general scenarios, including forms of noise.

*The PMAC model.* Although this paper focuses only on learning submodular functions, the PMAC model that we introduce is interesting in its own right and can be used to study the learnability of other real-valued functions. Subsequent work by Badanidiyuru et al. [3] and Balcan et al. [5] has used this model for studying the learnability of other classes of real-valued set functions that are widely used in algorithmic game theory. See section 1.3 for further discussion.

### 1.1.3. Other hardness implications of our matroid construction.

*Algorithmic game theory and economics.* An important consequence of our matroid construction is that matroid rank functions do not have a "sketch," i.e., a concise, approximate representation. Formally, there exist matroid rank functions on $2^{[n]}$ that do not have any poly$(n)$-space representation which approximates every value of the function to within a $o(n^{1/3}/\log n)$ factor.

In fact, as matroid rank functions are known to satisfy the gross substitute property [75], our work implies that gross substitutes do not have a concise, approximate representation, or, in game theoretic terms, gross substitutes do not have a bidding language. This provides a surprising answer to an open question in economics [8] [9, section 6.2.1], [10, section 2.2].

*Implications for submodular optimization.* Many optimization problems involving submodular functions, such as linear optimization over a submodular base polytope, submodular function minimization, and submodular flow, are very well behaved and their optimal solutions have a rich structure. We consider several other submodular optimization problems which have been considered recently in the literature, specifically submodular function minimization under a cardinality constraint, submodular $s$-$t$ min cut, and submodular vertex cover. These are difficult optimization problems, in the sense that the optimum value is hard to compute. We show that they are also difficult in the sense that their optimal solutions are very unstructured: the optimal solutions do not have a succinct representation, or even a succinct, approximate representation.

Formally, the problem of submodular function minimization under a cardinality constraint is

$$\min\{\, f(A) \,:\, A \subseteq [n],\, |A| \geq d \,\},$$

where $f$ is a monotone, submodular function. We show that there is no representation in poly$(n)$ bits for the minimizers of this problem, even allowing a factor $o(n^{1/3}/\log n)$ multiplicative error. In contrast, a much simpler construction [34, 89, 33] shows

that no deterministic algorithm performing poly($n$) queries to $f$ can approximate the minimum value to within a factor $o(n^{1/2}/\log n)$, but that construction implies nothing about small-space representations of the minimizers.

For the submodular $s$-$t$ min cut problem, which is a generalization of the classic $s$-$t$ min cut problem in network flow theory, we show that there is no representation in poly($n$) bits for the minimizers, even allowing a factor $o(n^{1/3}/\log n)$ multiplicative error. Similarly, for the submodular vertex cover problem, which is a generalization of the classic vertex cover problem, we show that there is no representation in poly($n$) bits for the minimizers, even allowing a factor $4/3$ multiplicative error. In contrast, the minimizers for the classic $s$-$t$ min cut problem and the classic vertex cover problem trivially have a representation in poly($n$) bits as they are determined by a graph on $n$ vertices. Moreover, for the $s$-$t$ min cut problem [83] and the vertex cover problem in bipartite graphs, the minimizers can be expressed as the minimizers of a submodular function, and so they can be efficiently represented as a lattice [71, section 5.1].

**1.2. Applications.** Algorithms for learning submodular functions could be very useful in some of the applications where these functions arise. For example, in the context of economics, our work provides useful tools for learning the valuation functions of (typical) customers, with applications such as bundle pricing, predicting demand, advertisement, etc. Our algorithms are also useful in settings where one would like to predict the value of some function over objects described by features, where the features have positive but decreasing marginal impact on the function's value. Examples include predicting the rate of growth of jobs in cities as a function of various amenities or enticements that the city offers, predicting the sales price of a house as a function of features (such as an updated kitchen, extra bedrooms, etc.) that it might have, and predicting the demand for a new laptop as a function of various add-ons that might be included. In all of these settings (and many others) it is natural to assume diminishing returns, making them well-suited to a formulation as a problem of learning a submodular function.

**1.3. Related work.** This section focuses primarily on prior work. Section 8.1 discusses subsequent work that was directly motivated by this paper.

*Submodular optimization.* Optimization problems involving submodular functions have long played a central role in combinatorial optimization. Recently there have been many applications of these optimization problems in machine learning, algorithmic game theory, and social networks.

The past decade has seen significant progress in algorithms for solving submodular optimization problems. There have been improvements in both the conceptual understanding and the running time of algorithms for submodular function minimization [45, 47, 85]. There has also been much progress on approximation algorithms for various problems. For example, there are now optimal approximation algorithms for submodular maximization subject to a matroid constraint [13, 29, 95], nearly optimal algorithms for nonmonotone submodular maximization [24, 25, 81], and algorithms for submodular maximization subject to a wide variety of constraints [15, 16, 26, 64, 65, 66, 81, 96].

Approximation algorithms for submodular analogues of several other optimization problems have been studied, including load balancing [89], set cover [46, 98], shortest path [32], sparsest cut [89], $s$-$t$ min cut [50], vertex cover [32, 46], etc. In this paper we provide several new results on the difficulty of such problems. Most of these previous papers on submodular optimization prove inapproximability results using matroids whose rank function has the same form as (1.1), but only for the drastically simpler

case of $k = 1$. Our construction is much more intricate since we must handle the case $k = n^{\omega(1)}$.

Recent work of Dobzinski and Vondrák [21] proves inapproximability of welfare maximization in combinatorial auctions with submodular valuations. Their proof is based on a collection of submodular functions that take high values on every set in a certain exponential-sized family and low values on sets that are far from that family. Their proof is in the same spirit as our inapproximability result, although their construction is technically very different from ours. In particular, our result uses a special family of submodular functions and family of sets for which the sets are *local minima* of the functions, whereas their result uses a different family of submodular functions and family of sets for which the sets are *local maxima* of the functions.

*Learning real-valued functions and the PMAC model.* In the machine learning literature [42, 93], learning real-valued functions in a distributional setting is often addressed by considering loss functions such as the $L_2$-loss or the $L_1$-loss, where the loss incurred by predicting according to hypothesis $f$ on a given example $x$ is $l_f(x, f^*) = (f(x) - f^*(x))^2$ for $L_2$-loss and $l_f(x, f^*) = |f(x) - f^*(x)|$ for $L_1$-loss. In this context, one typically normalizes the function to be in $[0, 1]$, and the aim is to achieve low expected loss $\mathbf{E}_x [\, l_f(x, f^*) \,]$. However, lower bounds on expected loss do not distinguish between the case of achieving low loss on most of the distribution and high loss on just a few points, versus moderately high loss everywhere.

For example, consider a function $f$ with codomain $\{0, 1, \ldots, n\}$. Here we would normalize by a factor $1/n$, so a lower bound of $\Omega(n^{1/3})$ on expected $L_1$-loss before normalizing is equivalent to a lower bound of $\Omega(n^{-2/3})$ after normalizing. But such a lower bound would not distinguish between the following two scenarios: (1) one where any hypothesis $f$ produced by the algorithm has $L_1$-loss of $\Omega(n^{-2/3})$ on a $1/2$ fraction of the points, and (2) one where an algorithm can output a hypothesis $f$ that is exactly correct on a $1{-}O(n^{-2/3})$ fraction of the points but has high loss on the rest.

In comparison, the PMAC model provides more fine-grained control, with separate parameters for the amount $\epsilon$ and extent $\alpha$ of errors. For instance, in scenario (1) if the normalized function has $f(x) = 1/n$ on the points of high $L_1$-loss, then this would correspond to a lower bound of $\alpha = \Omega(n^{1/3})$ and $\epsilon = 1/2$ in the PMAC model. In contrast, scenario (2) would correspond to having an upper bound of $\alpha = 1$ and $\epsilon = n^{-2/3}$.

Another advantage of the PMAC model is that, since it uses multiplicative error, an algorithm in the PMAC model provides good approximations *uniformly at all scales*.[3] Existing work in the learning theory literature has also considered guarantees that combine both multiplicative and additive aspects, in the context of sample complexity bounds [43, 68]. However, this work did not consider the development of efficient algorithms or learnability of submodular functions.

We remark that our construction showing the $\tilde{\Omega}(n^{1/3})$ inapproximability in the PMAC model immediately implies a lower bound of $\tilde{\Omega}(n^{-2/3})$ for the $L_1$-loss and $\tilde{\Omega}(n^{-4/3})$ for the $L_2$-loss (after normalization).

*Learning submodular functions.* To our knowledge, there is no prior work on learning submodular functions in a distributional, PAC-style learning setting. The most relevant work is a paper of Goemans et al. [33], which considers the problem of "approximating submodular functions everywhere." That paper considers the algorithmic

[3] For example, if $f$ is a good hypothesis in the PMAC model, and one focuses on points $x$ such that $f^*(x) \leq c$ and rescales, then the multiplicative approximation guarantee provided by $f$ remains true in the restricted and rescaled domain, as long as this set has sufficiently large probability mass.

problem of efficiently finding a function which approximates a submodular function at *every* set in its domain. They give an algorithm which achieves an approximation factor $\tilde{O}(\sqrt{n})$, and they also show $\tilde{\Omega}(\sqrt{n})$ inapproximability. Their algorithm adaptively queries the given function *on sets of its choice*, and their output function must approximate the given function on *every* set.[4] In contrast, our PMAC model falls into the more widely studied passive, supervised learning setting [2, 58, 92, 93], which is more relevant for our motivating applications discussed in section 1.2.

Our algorithm for PMAC-learning under general distributions and the Goemans et al. algorithm both rely on the structural result (due to Goemans et al.) that monotone, submodular functions can be approximated by the square root of a linear function to within a factor $\sqrt{n}$. In both cases, the challenge is to find this linear function. The Goemans et al. algorithm is very sophisticated: it gives an intricate combinatorial algorithm to approximately solve a certain convex program which produces the desired function. Their algorithm requires query access to the function and so it is not applicable in the PMAC model. Our algorithm, on the other hand, is very simple: given the structural result, we can reduce our problem to that of learning a linear separator, which is easily solved by linear programming. Moreover, our algorithm is noise-tolerant and more amenable to extensions; we elaborate on this in section 4.4.

On the other hand, our lower bound is significantly more involved than the lower bound of Goemans et al. [33] and the related lower bounds of Svitkina and Fleischer [89]. Essentially, the previous results show only *worst-case* inapproximability, whereas we need to show *average-case* inapproximability. A similar situation occurs with Boolean functions, where lower bounds for distributional learning are typically much harder to show than lower bounds for exact learning (i.e., learning everywhere). For instance, even conjunctions are hard to learn in the exact learning model (from random examples or via membership queries), and yet they are trivial to PAC-learn. Proving a lower bound for PAC-learning requires exhibiting some fundamental complexity in the class of functions. It is precisely this phenomenon which makes our lower bound challenging to prove.

*Learning valuation functions and other economic solutions concepts.* As discussed in section 1.2, one important application of our results on learning is for learning valuation functions. Kalai [56] considered the problem of learning *rational choice functions* from random examples. Here, the learning algorithm observes sets $S \subseteq [n]$ drawn from some distribution $D$, along with a choice $c(S) \in [n]$ for each $S$. The goal is then to learn a good approximation to $c$ under various natural assumptions on $c$. For the assumptions considered in [56], the choice function $c$ has a simple description as a linear ordering. In contrast, in our work we consider valuation functions that may be much more complex and for which the PAC model would not be sufficient to capture the inherent easiness or difficulty of the problem. Kalai briefly considers utility functions over bundles and remarks that "the PAC-learnability of preference relations and choice functions on commodity bundles ... deserves further study" [55].

**1.4. Structure of the paper.** We begin with background about matroids and submodular functions in section 2. In section 3 we present our new structural results:

[4] Technically speaking, their model can be viewed as "approximate learning everywhere with value queries," which is less natural in certain machine learning scenarios. In particular, in many applications arbitrary membership or value queries are undesirable because natural oracles, such as hired humans, have difficulty labeling synthetic examples [7]. Also, negative results for approximate learning everywhere do not necessarily imply hardness for learning in more widely used learning models. We discuss this in more detail below.

a new extremal family of matroids and the concentration result for submodular functions. We present our new framework for learning real-valued functions as well as our results for learning submodular functions within this framework in section 4. We further present implications of our matroid construction in optimization and algorithmic game theory in sections 6 and 7.

## 2. Preliminaries: Submodular functions and matroids.

**2.1. Notation.** Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. This will typically be used as the ground set for the matroids and submodular functions that we discuss. For any set $S \subseteq [n]$ and element $x \in [n]$, we let $S + x$ denote $S \cup \{x\}$. The indicator vector of a set $S \subseteq [n]$ is $\chi(S) \in \{0,1\}^n$, where $\chi(S)_i$ is 1 if $i$ is in $S$ and 0 otherwise. We frequently use this natural isomorphism between $\{0,1\}^n$ and $2^{[n]}$.

**2.2. Submodular functions and matroids.** In this section we give a brief introduction to matroids and submodular functions and discuss some standard facts that will be used throughout the paper. A more detailed discussion can be found in standard references [30, 31, 70, 82, 86]. The reader familiar with matroids and submodular functions may wish to skip to section 3.

Let $V = \{v_1, \ldots, v_n\}$ be a collection of vectors in some vector space $\mathbb{F}^m$. Roughly one century ago, several researchers observed that the linearly independent subsets of $V$ satisfy some interesting combinatorial properties. For example, if $B \subseteq V$ is a basis of $\mathbb{F}^m$ and $I \subseteq V$ is linearly independent but not a basis, then there is always a vector $v \in B$ which is not in the span of $I$, implying that $I + v$ is also linearly independent.

These combinatorial properties are quite interesting to study in their own right, as there are a wide variety of objects which satisfy these properties but (at least superficially) have no connection to vector spaces. A *matroid* is defined to be any collection of elements that satisfies these same combinatorial properties, without referring to any underlying vector space. Formally, a pair $\mathbf{M} = ([n], \mathcal{I})$ is called a matroid if $\mathcal{I} \subseteq 2^{[n]}$ is a nonempty family such that

- if $J \subseteq I$ and $I \in \mathcal{I}$, then $J \in \mathcal{I}$, and
- if $I, J \in \mathcal{I}$ and $|J| < |I|$, then there exists an $i \in I \setminus J$ such that $J + i \in \mathcal{I}$.

The sets in $\mathcal{I}$ are called *independent*.

Let us illustrate this definition with two examples:

Partition matroid. Let $V_1 \cup \cdots \cup V_k$ be a partition of $[n]$, i.e., $\bigcup_i V_i = [n]$ and $V_i \cap V_j = \emptyset$ whenever $i \neq j$. Define $\mathcal{I} \subseteq 2^{[n]}$ to be the family of partial transversals of $[n]$, i.e., $I \in \mathcal{I}$ iff $|I \cap V_i| \leq 1$ for all $i \in [k]$. It is easy to verify that the pair $([n], \mathcal{I})$ satisfies the definition of a matroid. This is called a *partition matroid*. This definition can be generalized slightly. Let $I \in \mathcal{I}$ iff $|I \cap V_i| \leq b_i$ for all $i = 1, \ldots, k$, where the $b_i$ values are arbitrary. The resulting pair $([n], \mathcal{I})$ is a (generalized) partition matroid.

Graphic matroid. Let $G$ be a graph with edge set $E$. Define $\mathcal{I} \subseteq 2^E$ to be the collection of all acyclic sets of edges. One can verify that the pair $([n], \mathcal{I})$ satisfies the definition of a matroid. This is called a *graphic matroid*.

One might wonder, Given an arbitrary matroid $([n], \mathcal{I})$, do there necessarily exist vectors $V = \{v_1, \ldots, v_n\}$ in some vector space for which the independent subsets of $V$ correspond to $\mathcal{I}$? Although this is true for partition matroids and graphic matroids, in general the answer is no. So matroids do not capture all properties of vector spaces. Nevertheless, many concepts from vector spaces do generalize to matroids.

For example, given vectors $V \subset \mathbb{F}^m$, all maximal linearly independent subsets of $V$ have the same cardinality, which is the dimension of the span of $V$. Similarly, given

a matroid $([n], \mathcal{I})$, all maximal sets in $\mathcal{I}$ have the same cardinality, which is called the *rank* of the matroid.

More generally, for any subset $V' \subseteq V$, we can define its rank to be the dimension of the span of $V'$; equivalently, this is the maximum size of any linearly independent subset of $V'$. This notion generalizes easily to matroids. The *rank function* of the matroid $([n], \mathcal{I})$ is the function $\text{rank}_{\mathbf{M}} : 2^{[n]} \to \mathbb{N}$ defined by

$$\text{rank}_{\mathbf{M}}(S) := \max\{\, |I| : I \subseteq S, I \in \mathcal{I} \,\}.$$

Rank functions also turn out to have numerous interesting properties, the most interesting of which is the *submodularity* property. Let us now illustrate this via an example. Let $V'' \subset V' \subset V$ be collections of vectors in some vector space. Suppose that $v \in V$ is a vector which does not lie in $\text{span}(V')$. Then it is clear that $v$ does not lie in $\text{span}(V'')$ either. Consequently,

$$\text{rank}(V' + v) - \text{rank}(V') = 1 \qquad \Longrightarrow \qquad \text{rank}(V'' + v) - \text{rank}(V'') = 1.$$

The submodularity property is closely related: it states that

$$\text{rank}_{\mathbf{M}}(T + i) - \text{rank}_{\mathbf{M}}(T) \quad \leq \quad \text{rank}_{\mathbf{M}}(S + i) - \text{rank}_{\mathbf{M}}(S) \qquad \forall S \subseteq T \subseteq [n],\, i \in [n].$$

The following properties of real-valued set functions play an important role in this paper. A function $f : 2^{[n]} \to \mathbb{R}$ is

- *normalized* if $f(\emptyset) = 0$,
- *nonnegative* if $f(S) \geq 0$ for all $S$,
- *monotone* (or *nondecreasing*) if $f(S) \leq f(T)$ for all $S \subseteq T$,
- *submodular* if it satisfies

$$f(T + i) - f(T) \quad \leq \quad f(S + i) - f(S) \qquad \forall S \subseteq T \subseteq [n],\, i \in [n], \tag{2.1}$$

  and an equivalent definition is

$$f(A) + f(B) \quad \geq \quad f(A \cup B) + f(A \cap B) \qquad \forall A, B \subseteq [n], \tag{2.2}$$

- *L-Lipschitz* if $|f(S + i) - f(S)| \leq L$ for all $S \subseteq [n]$ and $i \in [n]$.

Matroid rank functions are integer-valued, normalized, nonnegative, monotone, submodular, and 1-Lipschitz. The converse is also true: any function satisfying those properties is a matroid rank function.

The most interesting of these properties is submodularity. It turns out that there are a wide variety of set functions which satisfy the submodularity property but do not come from matroids. Let us mention two examples:

Coverage function. Let $S_1, \ldots, S_n$ be a subsets of a ground set $[m]$. Define the function $f : 2^{[n]} \to \mathbb{N}$ by

$$f(I) \;=\; \Big|\bigcup_{i \in I} S_i\Big|.$$

This is called a *coverage function*. It is integer-valued, normalized, nonnegative, monotone, and submodular, but it is not 1-Lipschitz.

Cut function. Let $G = ([n], E)$ be a graph. Define the function $f : 2^{[n]} \to \mathbb{N}$ by

$$f(U) \;=\; |\delta(U)|,$$

where $\delta(U)$ is the set of all edges that have exactly one endpoint in $U$. This is called a *cut function*. It is integer-valued, normalized, nonnegative, and submodular, but it is not monotone or 1-Lipschitz.

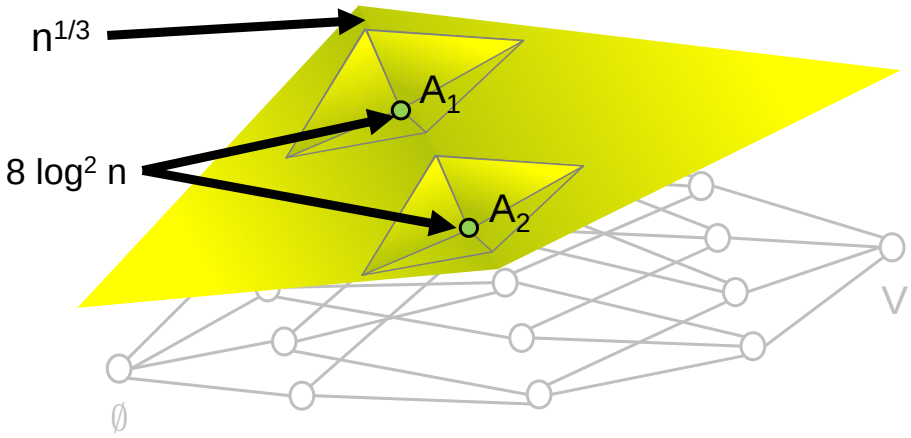


FIG. 1. *This figure aims to illustrate a function* $\text{rank}_{\mathbf{M}_\mathcal{B}}$ *that is constructed by* Theorem 1. *This is a real-valued function whose domain is the lattice of subsets of* $V$. *The family* $\mathcal{B}$ *contains the sets* $A_1$ *and* $A_2$, *both of which have size* $n^{1/3}$. *Whereas* $\text{rank}_{\mathbf{M}_\mathcal{B}}(S)$ *is large (close to* $n^{1/3}$*) for most sets* $S$ *of size* $n^{1/3}$, *we have* $\text{rank}_{\mathbf{M}_\mathcal{B}}(A_1) = \text{rank}_{\mathbf{M}_\mathcal{B}}(A_2) = 8\log^2 n$. *In order to ensure submodularity, sets near* $A_1$ *or* $A_2$ *also have low values.*

## 3. New structural results about matroids and submodular functions.

**3.1. A new family of extremal matroids.** In this section we present a new family of matroids whose rank functions take wildly varying values on many sets. The formal statement of this result is as follows.

THEOREM 1. *For any* $k \geq 8$ *with* $k = 2^{o(n^{1/3})}$, *there exists a family of sets* $\mathcal{A} \subseteq 2^{[n]}$ *and a family of matroids* $\mathcal{M} = \{ \mathbf{M}_\mathcal{B} : \mathcal{B} \subseteq \mathcal{A} \}$ *with the following properties:*

- $|\mathcal{A}| = k$ *and* $|A| = n^{1/3}$ *for every* $A \in \mathcal{A}$.
- *For every* $\mathcal{B} \subseteq \mathcal{A}$ *and every* $A \in \mathcal{A}$, *we have*

$$\text{rank}_{\mathbf{M}_\mathcal{B}}(A) \;=\; \begin{cases} 8\log k & \text{(if } A \in \mathcal{B}\text{)}, \\ |A| & \text{(if } A \in \mathcal{A} \setminus \mathcal{B}\text{)}. \end{cases}$$

Theorem 1 implies that there exists a superpolynomial-sized collection of subsets of $[n]$ such that, for *any* labeling of those sets as HIGH or LOW, we can construct a matroid where the sets in HIGH have rank $r_{\text{high}}$ and the sets in LOW have rank $r_{\text{low}}$, and the ratio $r_{\text{high}}/r_{\text{low}} = \tilde{\Omega}(n^{1/3})$. For example, by picking $k = n^{\log n}$, in the matroid $\mathbf{M}_\mathcal{B}$, a set $A$ has rank only $O(\log^2 n)$ if $A \in \mathcal{B}$, but has rank $n^{1/3}$ if $A \in \mathcal{A} \setminus \mathcal{B}$. In other words, as $\mathcal{B}$ varies, the rank of a set $A \in \mathcal{A}$ varies wildly, depending on whether $A \in \mathcal{B}$.

Later sections of the paper use Theorem 1 to prove various negative results. In section 4.3 we use the theorem to prove our inapproximability result for PMAC-learning submodular functions under arbitrary distributions. In section 6 we use the theorem to prove results on the difficulty of several submodular optimization problems.

In the remainder of section 3.1 we discuss Theorem 1 and give a detailed proof.

**3.1.1. Discussion of Theorem 1 and sketch of the construction.** We begin by discussing some set systems which give intuition on how Theorem 1 is proven. Let $\mathcal{A} = \{A_1, \ldots, A_k\}$ be a collection of subsets of $[n]$ and let $b_1, \ldots, b_k$ be positive integers.

Consider the set system

$$\mathcal{I} \;=\; \{\, I \,:\, |I| \le r \;\;\wedge\;\; |I \cap A_j| \le b_j \;\; \forall j \in [k] \,\}\,.$$

If $\mathcal{I}$ is the family of independent sets of a matroid $\mathbf{M}$, and if $\mathrm{rank}_{\mathbf{M}}(A_j) = b_j$ for each $j$, then perhaps such a construction can be used to prove Theorem 1.

Even in the case $k = 2$, understanding $\mathcal{I}$ is quite interesting. First of all, $\mathcal{I}$ typically is not a matroid. Consider taking $n = 5$, $r = 4$, $A_1 = \{1, 2, 3\}$, $A_2 = \{3, 4, 5\}$, and $b_1 = b_2 = 2$. Then both $\{1, 2, 4, 5\}$ and $\{2, 3, 4\}$ are maximal sets in $\mathcal{I}$ but their cardinalities are unequal, which violates a basic matroid property. However, one can verify that $\mathcal{I}$ is a matroid if we additionally require that $r \le b_1 + b_2 - |A_1 \cap A_2|$. In fact, we could place a constraint on $|I \cap (A_1 \cup A_2)|$ rather than on $|I|$, obtaining

$$\{\, I \,:\, |I \cap A_1| \le b_1 \;\;\wedge\;\; |I \cap A_2| \le b_2 \;\;\wedge\;\; |I \cap (A_1 \cup A_2)| \le b_1 + b_2 - |A_1 \cap A_2| \,\}\,,$$

which is the family of independent sets of a matroid. In the case that $A_1$ and $A_2$ are disjoint, the third constraint becomes $|I \cap (A_1 \cup A_2)| \le b_1 + b_2$, which is redundant because it is implied by the first two constraints. In the case that $A_1$ and $A_2$ are "nearly disjoint," this third constraint becomes necessary and it incorporates an "error term" of $-|A_1 \cap A_2|$.

To generalize to $k > 2$, we impose similar constraints for every subcollection of $\mathcal{A}$, and we must include additional "error terms" that are small when the $A_j$'s are nearly disjoint. First define

$$A(J) \;:=\; \bigcup_{j \in J} A_j.$$

Theorem 2 proves that

$$\mathcal{I} \;=\; \{\, I \,:\, |I \cap A(J)| \;\le\; g(J) \;\; \forall J \subseteq [k] \,\} \tag{3.1}$$

is a matroid, where the function $g : 2^{[k]} \to \mathbb{Z}$ is defined by

$$g(J) \;:=\; \sum_{j \in J} b_j \;-\; \left( \sum_{j \in J} |A_j| - |A(J)| \right). \tag{3.2}$$

The proof appears in section 3.1.3. In the definition of $g(J)$, we should think of $-(\sum_{j \in J} |A_j| - |A(J)|)$ as an "error term," since it is nonpositive, and it captures the "overlap" of the sets $\{\, A_j \,:\, j \in J \,\}$. In particular, in the case $J = \{1, 2\}$, this error term is $-|A_1 \cap A_2|$, as it was in our discussion of the case $k = 2$.

THEOREM 2. *The family $\mathcal{I}$ given in* (3.1) *is the family of independent sets of a matroid, if it is nonempty.*

Let us now consider a special case of this construction. If the $A_j$'s are all disjoint, then the error terms are all 0, so the family $\mathcal{I}$ reduces to

$$\{\, I \,:\, |I \cap A_j| \;\le\; b_j \;\; \forall j \in [k] \,\}\,,$$

which is a (generalized) partition matroid, regardless of the $b_j$ values. Unfortunately these matroids cannot achieve our goal of having superpolynomially many sets labeled HIGH or LOW. The reason is that, since the $A_j$'s must be disjoint, there can be at most $n$ of them.

In fact, it turns out that any matroid of the form (3.1) can have at most $n$ sets in the collection $\mathcal{A}$. To obtain a superpolynomially large $\mathcal{A}$ we must modify this construction slightly. Theorem 3 shows that, under certain conditions, the family

$$\bar{\mathcal{I}} \;=\; \{I \;:\; |I| \le d \;\;\wedge\;\; |I \cap A(J)| \le g(J) \;\; \forall J \subseteq [k],\, |J| < \tau\}$$

is also the family of independent sets of a matroid. Introducing the crucial parameter $\tau$ allows us to obtain a superpolynomially large $\mathcal{A}$.

There is an important special case of this construction. Suppose that $|A_j| = d$ and $b_j = d - 1$ for every $j$ and that $|A_i \cap A_j| \le 2$ for all $i \ne j$. The resulting matroid is called a *paving matroid*, a well-known type of matroid. These matroids are quite relevant to our goals of having superpolynomially many sets labeled HIGH and LOW. The reason is that the conditions on the $A_j$'s are equivalent to $\mathcal{A}$ being a constant-weight error-correcting code of distance 4, and it is well known that such codes can have superpolynomial size. Unfortunately this construction has $r_{\text{low}} = d - 1$ and $r_{\text{high}} = d$; this small, additive gap is much too weak for our purposes.

The high-level plan underlying Theorem 1 is to find a new class of matroids that somehow combines the positive attributes of both partition and paving matroids. From paving matroids we will inherit the large size of the collection $\mathcal{A}$, and from partition matroids we will inherit a large ratio $r_{\text{high}}/r_{\text{low}}$.

One of our key observations is that there is a commonality between partition and paving matroids: the collection $\mathcal{A}$ must satisfy an "expansion" property, which roughly means that the $A_j$'s cannot overlap too much. With partition matroids the $A_j$'s must be disjoint, which amounts to having "perfect" expansion. With paving matroids the $A_j$'s must have small pairwise intersections, which is a fairly weak sort of expansion.

It turns out that the "perfect" expansion required by partition matroids is too strong for $\mathcal{A}$ to have superpolynomial size, and the "pairwise" expansion required by paving matroids is too weak to allow a large ratio $r_{\text{high}}/r_{\text{low}}$. Fortunately, weakening the expansion from "perfect" to "nearly perfect" is enough to obtain a collection $\mathcal{A}$ of superpolynomial size. With several additional technical ideas, we show that these nearly perfect expansion properties can be leveraged to achieve our desired ratio $r_{\text{high}}/r_{\text{low}} = \tilde{\Omega}(n^{1/3})$. These ideas lead to a proof of Theorem 1.

**3.1.2. Our main matroid construction.** As mentioned above, our basic matroid construction (Theorem 2) does not suffice to prove Theorem 1. To see why, suppose that $|\mathcal{A}| = k > n$ and that $b_i < |A_i|$ for every $i$. Then $g([k]) \le n - k < 0$, and therefore $\mathcal{I}$ is empty. So the construction of Theorem 2 is applicable only when $k \le n$, which is insufficient for proving Theorem 1.

We now modify the preceding construction by introducing a sort of "truncation" operation which allows us to take $k \gg n$. We emphasize that this truncation is *not* ordinary matroid truncation. The ordinary truncation operation *decreases* the rank of the matroid, whereas we want to *increase* the rank by throwing away constraints in the definition of $\mathcal{I}$. We will introduce an additional parameter $\tau$ and only keep constraints for $|J| < \tau$. So long as $g$ is large enough for a certain interval, then we can truncate $g$ and still get a matroid.

DEFINITION 1. *Let $d$ and $\tau$ be nonnegative integers. A function $g : 2^{[k]} \to \mathbb{R}$ is called $(d, \tau)$-large if*

$$g(J) \;\ge\; \begin{cases} 0 & \forall J \subseteq [k],\; |J| < \tau, \\ d & \forall J \subseteq [k],\; \tau \le |J| \le 2\tau - 2. \end{cases} \tag{3.3}$$

*The truncated function* $\bar{g} : 2^{[k]} \to \mathbb{Z}$ *is defined by*

$$\bar{g}(J) := \begin{cases} g(J) & (\text{if } |J| < \tau), \\ d & (\text{otherwise}). \end{cases}$$

THEOREM 3. *Suppose that the function* $g$ *defined in* (3.2) *is* $(d, \tau)$*-large. Then the family*

$$\bar{\mathcal{I}} \;=\; \{\, I \,:\, |I \cap A(J)| \leq \bar{g}(J) \;\; \forall J \subseteq [k] \,\}$$

*is the family of independent sets of a matroid.*

Consequently, we claim that the family

$$\bar{\mathcal{I}} \;=\; \{I \;:\; |I| \leq d \;\;\wedge\;\; |I \cap A(J)| \leq g(J) \;\; \forall J \subseteq [k],\, |J| < \tau\}$$

is also the family of independent sets of a matroid. This claim follows immediately if the $A_i$'s cover the ground set (i.e., $A([k]) = [n]$), because the matroid definition in Theorem 3 includes the constraint $|I| = |I \cap A([k])| \leq \bar{g}([k]) = d$. Alternatively, if $A([k]) \neq [n]$, we may we apply the well-known matroid truncation operation which constructs a new matroid simply by removing all independent sets of size greater than $d$.

This construction yields quite a broad family of matroids. We list several interesting special cases in Appendix E. In particular, partition matroids and paving matroids are both special cases. Thus, our construction can produce "nonlinear" matroids (i.e., matroids that do not correspond to vectors over any field), as the Vámos matroid is a paving matroid that is nonlinear [82].

**3.1.3. Proofs of Theorems 2 and 3.** In this section, we will prove Theorems 2 and 3. We start with a simple but useful lemma which describes a general set of conditions that suffice to obtain a matroid.

Let $\mathcal{C} \subseteq 2^{[n]}$ be an arbitrary family of sets and let $g : \mathcal{C} \to \mathbb{Z}$ be a function. Consider the family

$$\mathcal{I} \;=\; \{\, I \,:\, |I \cap C| \leq g(C) \;\; \forall C \in \mathcal{C} \,\}\,. \tag{3.4}$$

For any $I \in \mathcal{I}$, define $T(I) = \{\, C \in \mathcal{C} \,:\, |I \cap C| = g(C) \,\}$ to be the set of constraints that are "tight" for the set $I$. Suppose that $g$ has the following property:

$$\forall I \in \mathcal{I}, \quad C_1, C_2 \in T(I) \quad \Longrightarrow \quad (C_1 \cup C_2 \in T(I)) \;\vee\; (C_1 \cap C_2 = \emptyset). \tag{3.5}$$

Properties of this sort are commonly called "uncrossing" properties. Note that we do not require that $C_1 \cap C_2 \in \mathcal{C}$. We show in the following lemma that this uncrossing property is sufficient[5] to obtain a matroid.

LEMMA 1. *Assume that* (3.5) *holds. Then* $\mathcal{I}$ *is the family of independent sets of a matroid if it is nonempty.*

*Proof.* We will show that $\mathcal{I}$ satisfies the required axioms of an independent set family. If $I \subseteq I' \in \mathcal{I}$, then clearly $I \in \mathcal{I}$ also. So suppose that $I \in \mathcal{I}$, $I' \in \mathcal{I}$, and $|I| < |I'|$. Let $C_1, \ldots, C_m$ be the maximal sets in $T(I)$ and let $C^* = \cup_i C_i$. Note

[5]There are general matroid constructions in the literature which are similar in spirit to Lemma 1, e.g., the construction of Edmonds [22, Theorem 15] and the construction of Frank and Tardos [86, Corollary 49.7a]. However, we were unable to use those existing constructions to prove Theorems 2 or 3.

that these maximal sets are disjoint; otherwise we could replace any intersecting sets with their union. In other words, $C_i \cap C_j = \emptyset$ for $i \neq j$; otherwise (3.5) implies that $C_i \cup C_j \in T(I)$, contradicting maximality. So

$$|I' \cap C^*| \;=\; \sum_{i=1}^{m} |I' \cap C_i| \;\leq\; \sum_{i=1}^{m} g(C_i) \;=\; \sum_{i=1}^{m} |I \cap C_i| \;=\; |I \cap C^*|.$$

Since $|I'| > |I|$ but $|I' \cap C^*| \leq |I \cap C^*|$, we must have that $|I' \setminus C^*| > |I \setminus C^*|$. The key consequence is that some element $x \in I' \setminus I$ is not contained in any tight set, i.e., there exists $x \in I' \setminus \big(C^* \cup I\big)$. Then $I + x \in \mathcal{I}$ because for every $C \in \mathcal{C}$ with $x \in C$ we have $|I \cap C| \leq g(C) - 1$. ∎

We now use Lemma 1 to prove Theorem 2, restated here.

THEOREM 2. *The family $\mathcal{I}$ defined in* (3.1)*, namely,*

$$\mathcal{I} \;=\; \{\, I \,:\, |I \cap A(J)| \;\leq\; g(J) \;\; \forall J \subseteq [k] \,\},$$

*where*

$$g(J) := \sum_{j \in J} b_j \;-\; \left( \sum_{j \in J} |A_j| - |A(J)| \right) \qquad \textit{and} \qquad A(J) := \bigcup_{j \in J} A_j,$$

*is the family of independent sets of a matroid if it is nonempty.*

This theorem is proven by showing that the constraints defining $\mathcal{I}$ can be "uncrossed" (in the sense that they satisfy (3.5)), then applying Lemma 1. It is not a priori obvious that these constraints can be uncrossed: in typical uses of uncrossing, the right-hand side $g(J)$ should be a submodular function of $J$ and the left-hand side $|I \cap A(J)|$ should be a *super*modular function of $J$. In our case both $g(J)$ and $|I \cap A(J)|$ are submodular functions of $J$.

*Proof of Theorem* 2. The proof applies Lemma 1 to the family

$$\mathcal{C} = \{\, A(J) \,:\, J \subseteq [k] \,\}.$$

We must also define a function $g' : \mathcal{C} \to \mathbb{Z}$ for which (3.5) will hold. However, there is a small issue: it is possible that there exist $J \neq J'$ with $A(J) = A(J')$ but $g(J) \neq g(J')$, so we cannot simply define $g'(A(J)) = g(J)$. Instead, we define the value of $g'(A(J))$ according the tightest constraint on $|I \cap A(J)|$, i.e.,

$$g'(C) := \min \{\, g(J) \,:\, A(J) = C \,\} \qquad \forall C \in \mathcal{C}.$$

Now fix $I \in \mathcal{I}$ and suppose that $C_1$ and $C_2$ are tight, i.e., $|I \cap C_i| = g'(C_i)$ for both $i = 1, 2$. Define $h_I : 2^{[k]} \to \mathbb{Z}$ by

$$h_I(J) \;:=\; g(J) - |I \cap A(J)| \;=\; |A(J) \setminus I| - \sum_{j \in J} (|A_j| - b_j).$$

We claim that $h_I$ is a submodular function of $J$. This follows because $J \mapsto |A(J) \setminus I|$ is a submodular function of $J$ (cf. Theorem 24 in Appendix A.1), and $J \mapsto \sum_{j \in J} (|A_j| - b_j)$ is a modular function of $J$.

Now choose $J_i$ satisfying $C_i = A(J_i)$ and $g'(C_i) = g(J_i)$ for both $i \in \{1, 2\}$. Then

$$h_I(J_i) = g(J_i) - |I \cap A(J_i)| = g'(C_i) - |I \cap C_i| = 0$$

for both $i \in \{1,2\}$. However, $h_I \geq 0$, since we assume $I \in \mathcal{I}$ and therefore $|I \cap A(J)| \leq g(J)$ for all $J$. So we have shown that $J_1$ and $J_2$ are both minimizers of $h_I$. It is well known that the minimizers of any submodular function are closed under union and intersection. (See Lemma 7 in Appendix A.1.) So $J_1 \cup J_2$ and $J_1 \cap J_2$ are also minimizers, implying that $A(J_1 \cup J_2) = A(J_1) \cup A(J_2) = C_1 \cup C_2$ is also tight.

This shows that (3.5) holds, so the theorem follows from Lemma 1. ∎

A similar approach is used for our second construction.

*Proof of* Theorem 3. Fix $I \in \bar{\mathcal{I}}$. Let $J_1$ and $J_2$ satisfy $|I \cap A(J_i)| = \bar{g}(J_i)$. By considering two cases, we will show that

$$|I \cap A(J_1 \cup J_2)| \;\geq\; \bar{g}(J_1 \cup J_2),$$

so the desired result follows from Lemma 1.

*Case* 1: $\max\{|J_1|, |J_2|\} \geq \tau$. Without loss of generality, $|J_1| \geq |J_2|$. Then

$$\bar{g}(J_1 \cup J_2) = d = \bar{g}(J_1) = |I \cap A(J_1)| \leq |I \cap A(J_1 \cup J_2)|.$$

*Case* 2: $\max\{|J_1|, |J_2|\} \leq \tau - 1$. So $|J_1 \cup J_2| \leq 2\tau - 2$. We have $|I \cap A(J_i)| = \bar{g}(J_i) = g(J_i)$ for both $i$. As argued in the proof of Theorem 2, we also have $|I \cap A(J_1 \cup J_2)| = g(J_1 \cup J_2)$. But $g(J_1 \cup J_2) \geq \bar{g}(J_1 \cup J_2)$ since $g$ is $(d,\tau)$-large, so $|I \cap A(J_1 \cup J_2)| \geq \bar{g}(J_1 \cup J_2)$, as desired. ∎

**3.1.4. Putting it all together: Proof of Theorem 1.** In this section we use the construction in Theorem 3 to prove Theorem 1, which is restated here.

THEOREM 1. *For any $k \geq 8$ with $k = 2^{o(n^{1/3})}$, there exists a family of sets $\mathcal{A} \subseteq 2^{[n]}$ and a family of matroids $\mathcal{M} = \{\, \mathbf{M}_{\mathcal{B}} : \mathcal{B} \subseteq \mathcal{A} \,\}$ with the following properties:*

- *$|\mathcal{A}| = k$ and $|A| = n^{1/3}$ for every $A \in \mathcal{A}$.*
- *For every $\mathcal{B} \subseteq \mathcal{A}$ and every $A \in \mathcal{A}$, we have*

$$\operatorname{rank}_{\mathbf{M}_{\mathcal{B}}}(A) \;=\; \begin{cases} 8\log k & \text{(if } A \in \mathcal{B}\text{)}, \\ |A| & \text{(if } A \in \mathcal{A} \setminus \mathcal{B}\text{)}. \end{cases}$$

To prove this theorem, we must construct a family of sets $\mathcal{A} = \{A_1, \ldots, A_k\}$ where each $|A_i| = n^{1/3}$, and for every $\mathcal{B} \subseteq \mathcal{A}$ we must construct a matroid $\mathbf{M}_{\mathcal{B}}$ with the desired properties: It will be convenient to let $d = n^{1/3}$ denote the size of the $A_i$'s, to let the index set of $\mathcal{A}$ be denoted by $U := [k]$, and to let the index set for $\mathcal{B}$ be denoted by $U_{\mathcal{B}} := \{\, i \in U : A_i \in \mathcal{B} \,\}$. Each matroid $\mathbf{M}_{\mathcal{B}}$ is constructed by applying Theorem 3 with the set family $\mathcal{B}$ instead of $\mathcal{A}$, so its independent sets are

$$\mathcal{I}_{\mathcal{B}} \;:=\; \left\{ I \;:\; |I| \leq d \;\;\wedge\;\; |I \cap A(J)| \leq g_{\mathcal{B}}(J) \;\; \forall J \subseteq U_{\mathcal{B}},\, |J| < \tau \right\},$$

where the function $g_{\mathcal{B}} : 2^{U_{\mathcal{B}}} \to \mathbb{R}$ is defined as in (3.2), taking all $b_i$'s to be equal to a common value $b$ (independent of $i$ and $\mathcal{B}$):

$$g_{\mathcal{B}}(J) \;:=\; \sum_{j \in J} b \;-\; \left( \sum_{j \in J} |A_j| - |A(J)| \right) \;=\; (b-d)|J| + |A(J)| \qquad \forall J \subseteq U_{\mathcal{B}}.$$

Several steps remain. We must choose the set family $\mathcal{A}$, then choose parameters carefully such that, for every $\mathcal{B} \subseteq \mathcal{A}$, we have as follows:

- P1: $\mathbf{M}_{\mathcal{B}}$ is indeed a matroid;
- P2: $\text{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) = 8 \log k$ for all $A_i \in \mathcal{B}$; and
- P3: $\text{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) = |A|$ for all $A_i \in \mathcal{A} \setminus \mathcal{B}$.

Let us start with P2. Suppose $A_i \in \mathcal{B}$. The definition of $\mathcal{I}_{\mathcal{B}}$ includes the constraint $|I \cap A_i| \leq g_{\mathcal{B}}(\{i\})$, which implies that $\text{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) \leq g_{\mathcal{B}}(\{i\}) = b$. This suggests that choosing $b := 8 \log k$ may be a good choice to satisfy P2.

On the other hand, if $A_i \notin \mathcal{B}$, then P3 requires that $A_i$ is independent in $\mathbf{M}_{\mathcal{B}}$. To achieve this, we need the constraints $|I \cap A(J)| \leq g_{\mathcal{B}}(J)$ to be as loose as possible, i.e., $g_{\mathcal{B}}(J)$ should be as large as possible. Notice that $g_{\mathcal{B}}(J)$ has two terms, $\sum_{j \in J} b$, which grows as a function of $J$, and $-\big(\sum_{j \in J} |A_j| - |A(J)|\big)$, which is nonpositive. So we desire that $|A(J)|$ should be as close as possible to $\sum_{j \in J} |A_j|$ for all $J$ with $|J| < \tau$. Set systems with this property can be constructed from expander graphs.

DEFINITION 2. *Let $G = (U \cup V, E)$ be a bipartite graph. For $J \subseteq U$, define*

$$\Gamma(J) := \{\, v : \exists u \in J \text{ such that } \{u, v\} \in E \,\}.$$

*The graph $G$ is called a $(d, L, \epsilon)$-expander if*

$$\begin{aligned} |\Gamma(\{u\})| &= d && \forall u \in U, \\ |\Gamma(J)| &\geq (1-\epsilon) \cdot d \cdot |J| && \forall J \subseteq U,\ |J| \leq L. \end{aligned}$$

*Additionally, $G$ is called a lossless expander if $\epsilon < 1/2$.*

Given such a graph $G$, we construct the set family $\mathcal{A} = \{A_1, \ldots, A_k\} \subseteq 2^{[n]}$ by identifying $U = [k]$, $V = [n]$, and for each vertex $i \in U$ defining $A_i := \Gamma(\{i\})$. The resulting sets satisfy

$$\begin{aligned} |A_i| &= d && \forall i \in U, \\ |A(J)| &\geq (1-\epsilon) \cdot d \cdot |J| && \forall J \subseteq U,\ |J| \leq L, \\ \implies \quad \textstyle\sum_{j \in J} |A_j| - |A(J)| &\leq \epsilon \cdot d \cdot |J| && \forall J \subseteq U,\ |J| \leq L. \end{aligned} \tag{3.6}$$

This last inequality will allow us to show that $g_{\mathcal{B}}(J)$ is sufficiently large.

To make things concrete, let us now state the expander construction that we will use. Lossless expanders are well-studied [39, 44], and several probabilistic constructions are known, both in folklore and in the literature [12, Lemma 3.10], [44, section 1.2], [88, Theorem 26], [91, Theorem 4.4]. The following construction has parameters that match our requirements.

THEOREM 4 (Buhrman et al. [12, Lemma 3.10]). *Suppose that $k \geq 8$, that $n \geq 25L \log(k)/\epsilon^2$, and that $d \geq \log(k)/2\epsilon$. Then there exists a graph $G = (U \cup V, E)$ with $|U| = k$ and $|V| = n$ that is a $(d, L, \epsilon)$-lossless expander.*

The next theorem states another (folklore) probabilistic construction that also matches our requirements. We include a proof in Appendix D for the sake of completeness and because we will require a slight variant in section 6.

THEOREM 5. *Let $G = (U \cup V, E)$ be a random multigraph where $|U| = k$, $|V| = n$, and every $u \in U$ has exactly $d$ incident edges, each of which has an endpoint chosen uniformly and independently from all nodes in $V$. Suppose that $k \geq 4$, $d \geq \log(k)/\epsilon$, and $n \geq 16Ld/\epsilon$. Then, with probability at least $1 - 2/k$,*

$$|\Gamma(J)| \geq (1-\epsilon) \cdot d \cdot |J| \qquad \forall J \subseteq U,\ |J| \leq L.$$

*If it is desired that $|\Gamma(\{u\})| = d$ for all $u \in U$, then this can be achieved by replacing any parallel edges incident on $u$ by new edges with distinct endpoints. This cannot decrease $|\Gamma(J)|$ for any $J$.*

We require an expander with the following parameters. Recall that $n$ is arbitrary and $k = 2^{o(n^{1/3})}$:

$$d := n^{1/3}, \qquad L := \frac{n^{1/3}}{2\log k}, \qquad \epsilon := \frac{2\log k}{n^{1/3}}.$$

These satisfy the hypotheses of Theorem 4 (and Theorem 5), so a $(d, L, \epsilon)$-expander exists, and a set family $\mathcal{A}$ satisfying (3.6) exists. Next we use these properties of $\mathcal{A}$ to show that P1, P2, and P3 hold.

The fact that P1 holds follows from Theorem 3 and the following claim. Recall that $b = 8\log k$.

CLAIM 1. *Set $\tau = n^{1/3}/4\log k$. Then $g_{\mathcal{B}}$ is $(d, \tau)$-large, as defined in* (3.3).

*Proof.* Consider any $J \subseteq U_{\mathcal{B}}$ with $|J| \le 2\tau - 2$. Then

$$\begin{aligned} g_{\mathcal{B}}(J) &= (b-d)|J| + |A(J)| \\ &\ge b|J| - \epsilon d|J| \qquad \text{(by (3.6), since } |J| \le 2\tau - 2 \le L) \\ &= \frac{3b}{4}|J| \qquad \text{(since } \epsilon = b/4d). \end{aligned} \tag{3.7}$$

This shows $g_{\mathcal{B}}(J) \ge 0$. If additionally $|J| \ge \tau$, then $g_{\mathcal{B}}(J) \ge (3/4)b\tau > d$. ∎

The following claim implies that P2 holds.

CLAIM 2. *For all $\mathcal{B} \subseteq \mathcal{A}$ and all $A_i \in \mathcal{B}$ we have* $\operatorname{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) = b$.

*Proof.* The definition of $\mathcal{I}_{\mathcal{B}}$ includes the constraint $|I \cap A_i| \le g_{\mathcal{B}}(\{i\}) = b$. This immediately implies $\operatorname{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) \le b$. To prove that equality holds, it suffices to prove that $g_{\mathcal{B}}(J) \ge b$ whenever $|J| \ge 1$, since this implies that every constraint in the definition of $\mathcal{I}_{\mathcal{B}}$ has a right-hand side at least $b$ (except for the constraint corresponding to $J = \emptyset$, which is vacuous). For $|J| = 1$ this is immediate, and for $|J| \ge 2$ we use (3.7) to obtain $g_{\mathcal{B}}(J) = 3b|J|/4 > b$. ∎

Finally, the following claim implies that P3 holds.

CLAIM 3. *For all $\mathcal{B} \subseteq \mathcal{A}$ and all $A_i \in \mathcal{A} \setminus \mathcal{B}$ we have* $\operatorname{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) = d$.

*Proof.* Since $d = |A_i|$, the condition $\operatorname{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) = d$ holds iff $A_i \in \mathcal{I}_{\mathcal{B}}$. So it suffices to prove that $A_i$ satisfies all constraints in the definition of $\mathcal{I}_{\mathcal{B}}$.

The constraint $|A_i| \le d$ is trivially satisfied. So it remains to show that for every $J \subseteq U_{\mathcal{B}}$ with $|J| < \tau$, we have

$$|A_i \cap A(J)| \le g_{\mathcal{B}}(J). \tag{3.8}$$

This is trivial if $J = \emptyset$, so assume $|J| \ge 1$. We have

$$\begin{aligned} |A_i \cap A(J)| &= |A_i| + |A(J)| - |A(J+i)| \\ &\le d + d|J| - (1-\epsilon)d|J+i| \qquad \text{(by (3.6))} \\ &= \frac{b\,|J+i|}{4} \qquad \text{(since } \epsilon = b/4d) \\ &\le \frac{b\,|J|}{2} \\ &\le g_{\mathcal{B}}(J) \qquad \text{(by (3.7)).} \end{aligned}$$

This proves (3.8), so $A_i \in \mathcal{I}_{\mathcal{B}}$, as desired. ∎

**3.2. Concentration properties of submodular functions.** In this section we prove the following strong concentration bound for submodular functions.

THEOREM 6. *Let $f : 2^{[n]} \to \mathbb{R}_+$ be a nonnegative, monotone, submodular, 1-Lipschitz function. Let the random variable $X \subseteq [n]$ have a product distribution. For any $b, t \geq 0$,*

$$\Pr\left[ f(X) \leq b - t\sqrt{b} \right] \cdot \Pr\left[ f(X) \geq b \right] \;\leq\; \exp(-t^2/4).$$

To understand Theorem 6, it is instructive to compare it with standard results. For example, the Chernoff bound is precisely a concentration bound for *linear*, Lipschitz functions. On the other hand, if $f$ is an arbitrary 1-Lipschitz function, then McDiarmid's inequality implies concentration, although of a much weaker form, with standard deviation roughly $\sqrt{n}$. If $f$ is additionally known to be submodular, then we can apply Theorem 6 with $b$ equal to a median, which can be much smaller than $n$. So Theorem 6 can be viewed as saying that McDiarmid's inequality can be significantly strengthened when the given function is known to be submodular.

Our proof of Theorem 6 is based on the Talagrand inequality [90, 1, 74, 49]. As mentioned above, similar results have previously been proven using the machinery of self-bounding functions [40, 97] and using the FKG inequality [14].

Theorem 6 most naturally implies concentration around a median of $f(X)$. As shown in the following corollary, this also implies concentration around the expected value. This corollary, with better constants, also follows from the results of Chekuri and Vondrák [14] and Vondrák [97]

COROLLARY 1. *Let $f : 2^{[n]} \to \mathbb{R}_+$ be a nonnegative, monotone, submodular, 1-Lipschitz function. Let the random variable $X \subseteq [n]$ have a product distribution. For any $0 \leq \alpha \leq 1$,*

$$\Pr\left[ |f(X) - \mathbf{E}\left[ f(X) \right]| > \alpha \, \mathbf{E}\left[ f(X) \right] \right] \;\leq\; 4 \exp\left( -\alpha^2 \, \mathbf{E}\left[ f(X) \right] / 422 \right).$$

As an interesting application of Corollary 1, let us consider the case where $f$ is the rank function of a linear matroid. Formally, fix a matrix $A$ over any field. Construct a random submatrix by selecting the $i$th column of $A$ with probability $p_i$, where these selections are made independently. Then Corollary 1 implies that the rank of the resulting submatrix is highly concentrated around its expectation, in a way that does not depend on the number of rows of $A$.

The proofs of this section are technical applications of Talagrand's inequality and are provided in Appendix B. Later sections of the paper use Theorem 6 and Corollary 1 to prove various results. In section 4.2 we use these theorems to analyze our algorithm for PMAC-learning submodular functions under product distributions. In section 5 we use these theorems to give an approximate characterization of matroid rank functions.

## 4. Learning submodular functions.

**4.1. A new learning model: The PMAC model.** In this section we introduce a new learning model for learning real-valued functions in the passive, supervised learning paradigm, which we call the PMAC model. In this model, a learning algorithm is given a collection $\mathcal{S} = \{x_1, x_2, \ldots, x_\ell\}$ of polynomially many sets drawn i.i.d. from some fixed, but unknown, distribution $D$ over an instance space $\mathcal{X}$. There is also a fixed but unknown function $f^* : \mathcal{X} \to \mathbb{R}_+$, and the algorithm is given the value of $f^*$ at each set in $\mathcal{S}$. The algorithm may perform an arbitrary polynomial time

computation on the examples $\{(x_i, f^*(x_i))\}_{1\le i\le \ell}$, then must output another function $f : \mathcal{X} \to \mathbb{R}_+$. This function is called a "hypothesis function." The goal is that, with high probability, $f$ is a good approximation of $f^*$ for most points in $D$. Formally, we have the following.

DEFINITION 3. *Let $\mathcal{F}$ be a family of nonnegative, real-valued functions with domain $\mathcal{X}$. We say that an algorithm $\mathcal{A}$ PMAC-learns $\mathcal{F}$ with approximation factor $\alpha$ if, for any distribution $D$ over $\mathcal{X}$, for any target function $f^* \in \mathcal{F}$, and for $\epsilon \ge 0$ and $\delta \ge 0$ sufficiently small,*

- *the input to $\mathcal{A}$ is a sequence of pairs $\{(x_i, f^*(x_i))\}_{1\le i\le \ell}$ where each $x_i$ is i.i.d. from $D$,*
- *the number of inputs $\ell$ provided to $\mathcal{A}$ and the running time of $\mathcal{A}$ are both at most $\mathrm{poly}(n, 1/\epsilon, 1/\delta)$,*
- *the output of $\mathcal{A}$ is a function $f : \mathcal{X} \to \mathbb{R}$ that can be evaluated in time $\mathrm{poly}(n, 1/\epsilon, 1/\delta)$ and that satisfies*

$$\Pr_{x_1,\ldots,x_\ell\sim D}\left[\Pr_{x\sim D}\left[\, f(x) \le f^*(x) \le \alpha \cdot f(x)\,\right] \ge 1-\epsilon\right] \;\ge\; 1-\delta.$$

PMAC is an extension of the PAC model to learning nonnegative, real-valued functions, allowing multiplicative error $\alpha$. The PAC model for learning Boolean functions is precisely the special case when $\alpha = 1$.

In this paper we focus on the PMAC-learnability of submodular functions. In this case $\mathcal{X} = \{0,1\}^n$ and $\mathcal{F}$ is the family of all nonnegative, monotone, submodular functions. We note that it is quite easy to PAC-learn the class of *Boolean* submodular functions. Details are given in Appendix C.1. The rest of this section considers the much more challenging task of PMAC-learning the general class of real-valued, submodular functions.

**4.2. Product distributions.** A first natural and common step in studying learning problems is to study learnability of functions when the examples are distributed according to the uniform distribution or a product distribution [53, 60, 69]. In this section we consider learnability of submodular functions when the underlying distribution is a product distribution. Building on the concentration results in section 3.2 we provide an algorithm that PMAC learns the class of Lipschitz submodular functions with a constant approximation factor.

We will let $L < M < H$ and $K$ be universal constants, whose values we can take to be $L = 10550$, $M = 11250$, $H = 12500$, and $K = 26000$. We begin with the following technical lemma, which states some useful concentration bounds.

LEMMA 2. *Let $f : 2^{[n]} \to \mathbb{R}$ be a nonnegative, monotone, submodular, 1-Lipschitz function. Suppose that $S_1, \ldots, S_l$ are drawn from a product distribution $D$ over $2^{[n]}$. Let $\mu$ the empirical average $\mu = \sum_{i=1}^{\ell} f^*(S_i)/\ell$, which is our estimate for $\mathbf{E}_{S\sim D}[\, f^*(S)\,]$. Let $\epsilon, \delta \le 1/5$. We have the following:*

(1) *If $\mathbf{E}[\, f^*(S)\,] > H\log(1/\epsilon)$ and $\ell \ge 16\log(1/\delta)$, then*

$$\Pr[\, \mu \ge M\log(1/\epsilon)\,] \;\ge\; 1-\delta/4.$$

(2) *If $\mathbf{E}[\, f^*(S)\,] > L\log(1/\epsilon)$ and $\ell \ge 16\log(1/\delta)$, then*

$$\Pr\left[\, \tfrac{5}{6}\,\mathbf{E}[\, f^*(S)\,] \le \mu \le \tfrac{4}{3}\,\mathbf{E}[\, f^*(S)\,]\,\right] \;\ge\; 1-\delta/4.$$

(3) *If $\mathbf{E}[\, f^*(S)\,] \le H\log(1/\epsilon)$, then*

$$\Pr[\, f^*(S) < K\log(1/\epsilon)\,] \;\ge\; 1-\epsilon.$$

**Algorithm 1.** An algorithm for PMAC-learning a nonnegative, monotone, 1-Lipschitz, submodular function $f^*$ with minimum nonzero value 1, when the examples come from a product distribution. Its input is a sequence of labeled training examples $(S_1, f^*(S_1)), \ldots, (S_\ell, f^*(S_\ell))$, parameters $\epsilon$ and $\ell$.

- Let $\mu = \sum_{i=1}^{\ell} f^*(S_i)/\ell$.
- *Case* 1: If $\mu \geq M \log(1/\epsilon)$, then return the constant function $f = \mu/4$.
- *Case* 2: If $\mu < M \log(1/\epsilon)$, then compute the set $U = \bigcup_{i\,:\,f^*(S_i)=0} S_i$. Return the function $f$ where $f(A) = 0$ if $A \subseteq U$ and $f(A) = 1$ otherwise.

(4) *If* $\mathbf{E}[f^*(S)] < L \log(1/\epsilon)$ *and* $\ell \geq 16 \log(1/\delta)$, *then*

$$\Pr[\mu < M \log(1/\epsilon)] \;\geq\; 1 - \delta/4.$$

The proof of Lemma 2, which is provided in Appendix C.2, follows easily from Theorem 6 and Corollary 1. We now present our main result in this section.

THEOREM 7. *Let $\mathcal{F}$ be the class of nonnegative, monotone, 1-Lipschitz, submodular functions with ground set $[n]$ and minimum nonzero value 1. Let $D$ be a product distribution on $\{0,1\}^n$. For any sufficiently small $\epsilon > 0$ and $\delta > 0$,* Algorithm 1 *PMAC-learns $\mathcal{F}$ with approximation factor $\alpha = K \log(1/\epsilon)$. The number of training examples used is $\ell = n \log(n/\delta)/\epsilon + 16 \log(1/\delta)$.*

*If it is known a priori that $\mathbf{E}[f^*(S)] \geq L \log(1/\epsilon)$, then the approximation factor improves to 8, and the number of examples can be reduced to $\ell = 16 \log(1/\delta)$, which is independent of $n$ and $\epsilon$.*

*Proof.* We begin with an overview of the proof. Consider the expected value of $f^*(S)$ when $S$ is drawn from distribution $D$. When this expected value of $f^*$ is large compared to $\log(1/\epsilon)$, we simply output a constant function given by the empirical average $\mu$ estimated by the algorithm. The concentration bounds for submodular functions (Theorem 6 and Corollary 1) allow us to show that this constant function provides a good estimate. However, when the expected value of $f^*$ is small, we must carefully handle the zeros of $f^*$, since they may have large measure under distribution $D$. The key idea here is to use the fact that the zeros of a nonnegative, monotone, submodular function have special structure: they are both union-closed and downward-closed, so it is sufficient to *PAC-learn* the Boolean NOR function which indicates the zeros of $f^*$.

We now present the proof formally. By Lemma 2, with probability at least $1 - \delta$ over the choice of examples, we may assume that the following implications hold:

(4.1)
$$\begin{aligned}
&\mu \geq M \log(1/\epsilon) \implies \mathbf{E}[f^*(S)] \geq L \log(1/\epsilon) \text{ and } \tfrac{5}{6}\mathbf{E}[f^*(S)] \leq \mu \leq \tfrac{4}{3}\mathbf{E}[f^*(S)],\\
&\mu < M \log(1/\epsilon) \implies \mathbf{E}[f^*(S)] \leq H \log(1/\epsilon).
\end{aligned}$$

Now we show that the function $f$ output by the algorithm approximates $f^*$ to within a factor $K \log(1/\epsilon)$.

*Case* 1. $\mu \geq M \log(1/\epsilon)$. Since we assume that (4.1) holds, we have $\frac{5}{6}\mathbf{E}[f^*(S)] \leq \mu \leq \frac{4}{3}\mathbf{E}[f^*(S)]$ and $\mathbf{E}[f^*(S)] \geq L \log(1/\epsilon)$. Using these together with Corollary 1 we obtain

(4.2)
$$\begin{aligned}
\Pr[\mu/4 \leq f^*(S) \leq 2\mu] &\geq \Pr\left[\tfrac{1}{3}\mathbf{E}[f^*(S)] \leq f^*(S) \leq \tfrac{5}{3}\mathbf{E}[f^*(S)]\right]\\
&\geq 1 - \Pr[\,|f^*(S) - \mathbf{E}[f^*(S)]| \geq (2/3)\mathbf{E}[f^*(S)]\,]\\
&\geq 1 - 4\exp\left(-\mathbf{E}[f^*(S)]/950\right) \;\geq\; 1 - \epsilon,
\end{aligned}$$

since $L \geq 4000$ and $\epsilon \leq 1/2$. Therefore, with confidence at least $1-\delta$, the constant function $f$ output by the algorithm approximates $f^*$ to within a factor 8 on all but an $\epsilon$ fraction of the distribution.

*Case* 2. $\mu < M \log(1/\epsilon)$. As mentioned above, we must separately handle the zeros and the nonzeros of $f^*$. To that end, define

$$\mathcal{P} = \{\, S \,:\, f^*(S) > 0 \,\} \qquad \text{and} \qquad \mathcal{Z} = \{\, S \,:\, f^*(S) = 0 \,\}.$$

Recall that the algorithm sets $U = \bigcup_{f^*(S_i)=0} S_i$. Monotonicity and submodularity imply that $f^*(U) = 0$. Furthermore, setting $\mathcal{L} = \{\, T \,:\, T \subseteq U \,\}$, monotonicity implies that

$$f^*(T) \;=\; 0 \qquad \forall T \in \mathcal{L}. \tag{4.3}$$

We wish to analyze the measure of the points for which the function $f$ output by the algorithm fails to provide a good estimate of $f^*$. So let $S$ be a new sample from $D$ and let $\mathcal{E}$ be the event that $S$ violates the inequality

$$f(S) \;\leq\; f^*(S) \;\leq\; \big(K \log(1/\epsilon)\big) \cdot f(S).$$

Our goal is to show that, with probability $1-\delta$ over the training examples, we have $\Pr[\,\mathcal{E}\,] \leq \epsilon$. Clearly

$$\Pr[\,\mathcal{E}\,] \;=\; \Pr[\,\mathcal{E} \,\wedge\, S \in \mathcal{P}\,] \;+\; \Pr[\,\mathcal{E} \,\wedge\, S \in \mathcal{Z}\,].$$

We will separately analyze these two probabilities.

First we analyze the nonzeros of $f^*$. So assume that $S \in \mathcal{P}$, which implies that $f^*(S) \geq 1$ by our hypothesis. Then $S \not\subseteq U$ (by (4.3)), and hence $f(S) = 1$ by the definition of $f$. Therefore the event $\mathcal{E} \wedge S \in \mathcal{P}$ can only occur when $f^*(S) > K \log(1/\epsilon)$. Since we assume that (4.1) holds, we have $\mathbf{E}[\,f^*(S)\,] \leq H \log(1/\epsilon)$, so we can apply Lemma 2, statement (3). This shows that

$$\Pr[\,\mathcal{E} \,\wedge\, S \in \mathcal{P}\,] \;\leq\; \Pr[\,f^*(S) > K \log(1/\epsilon)\,] \;\leq\; \epsilon.$$

It remains to analyze the zeros of $f^*$. Assume that $S \in \mathcal{Z}$, i.e., $f^*(S) = 0$. Since our hypothesis has $f(S) = 0$ for all $S \in \mathcal{L}$, the event $\mathcal{E} \wedge S \in \mathcal{Z}$ holds only if $S \in \mathcal{Z} \setminus \mathcal{L}$. The proof now follows from Claim 4. ∎

Claim 4. *With probability at least $1-\delta$, the set $\mathcal{Z} \setminus \mathcal{L}$ has measure at most $\epsilon$.*

*Proof.* The idea of the proof is as follows. At any stage of the algorithm, we can compute the set $U$ and the subcube $\mathcal{L} = \{\, T \,:\, T \subseteq U \,\}$. We refer to $\mathcal{L}$ as the algorithm's *null subcube*. Suppose that there is at least an $\epsilon$ chance that a new example is a zero of $f^*$ but does not lie in the null subcube. Then such an example should be seen in the next sequence of $\log(1/\delta)/\epsilon$ examples, with probability at least $1-\delta$. This new example increases the dimension of the null subcube by at least one, and therefore this can happen at most $n$ times.

Formally, for $k \leq \ell$, define

$$U_k \;=\; \bigcup_{\substack{i \leq k \\ f^*(S_i)=0}} S_i \qquad \text{and} \qquad \mathcal{L}_k \;=\; \{\, S \,:\, S \subseteq \mathcal{U}_k \,\}.$$

As argued above, we have $\mathcal{L}_k \subseteq \mathcal{Z}$ for any $k$. Suppose that, for some $k$, the set $\mathcal{Z} \setminus \mathcal{L}_k$ has measure at least $\epsilon$. Define $k' = k + \log(n/\delta)/\epsilon$. Then among the subsequent examples $S_{k+1}, \ldots, S_{k'}$, the probability that none of them lie in $\mathcal{Z} \setminus \mathcal{L}_k$ is at most

$$(1-\epsilon)^{\log(n/\delta)/\epsilon} \leq \delta/n.$$

On the other hand, if one of them does lie in $\mathcal{Z} \setminus \mathcal{L}_k$, then $|\mathcal{U}_{k'}| > |\mathcal{U}_k|$. But $|\mathcal{U}_k| \leq n$ for all $k$, so this can happen at most $n$ times. Since $\ell \geq n \log(n/\delta)/\epsilon$, with probability at least $\delta$ the final set $\mathcal{Z} \setminus \mathcal{L}_\ell$ has measure at most $\epsilon$. ∎

The class $\mathcal{F}$ defined in Theorem 7 contains the class of matroid rank functions. We remark that Theorem 7 can be easily modified to handle the case where the minimum nonzero value for functions in $\mathcal{F}$ is $\eta < 1$. To do this, we simply modify step 2 of the algorithm to output $f(A) = \eta$ for all $A \not\subseteq U$. The same proof shows that this modified algorithm has an approximation factor of $K \log(1/\epsilon)/\eta$.

**4.3. Inapproximability under arbitrary distributions.** The simplicity of Algorithm 1 might raise one's hopes that a constant-factor approximation is possible under arbitrary distributions. However, we show in this section that no such approximation is possible. In particular, by making use of the new family of matroids we presented in section 3.1, we show that no algorithm can PMAC-learn the class of nonnegative, monotone, submodular functions with approximation factor $o(n^{1/3}/\log n)$. Formally, we have the following.

THEOREM 8. *Let $\mathcal{ALG}$ be an arbitrary learning algorithm that uses only a polynomial number of training examples drawn i.i.d. from the underlying distribution. There exists a distribution $D$ and a submodular target function $f^*$ such that, with probability at least $1/8$ (over the draw of the training samples), the hypothesis function $f$ output by $\mathcal{ALG}$ does not approximate $f^*$ within an $o(n^{1/3}/\log n)$ factor on at least a $1/4$ fraction of the examples under $D$. This holds even for the subclass of matroid rank functions.*

*Proof.* To show the lower bound, we use the family of matroids from Theorem 1 in section 3.1.4, whose rank functions take wildly varying values on a large set of points. The high-level idea is to show that for a superpolynomial-sized set of $k$ points in $\{0,1\}^n$, and for *any* partition of those points into HIGH and LOW, we can construct a matroid where the points in HIGH have rank $r_{\text{high}}$ and the points in LOW have rank $r_{\text{low}}$, and the ratio $r_{\text{high}}/r_{\text{low}} = \tilde{\Omega}(n^{1/3})$. This then implies hardness for learning over the uniform distribution on these $k$ points from any polynomial-sized sample, even with value queries.

To make the proof formal, we use the probabilistic method. Assume that $\mathcal{ALG}$ uses $\ell \leq n^c$ training examples for some constant $c$. To construct a hard family of submodular functions, we will apply Theorem 1 with $k = 2^t$, where $t = c\log(n) + 3$. Let $\mathcal{A}$ and $\mathcal{M}$ be the families that are guaranteed to exist by Theorem 1. Let the underlying distribution $D$ on $2^{[n]}$ be the uniform distribution on $\mathcal{A}$. (We note that $D$ is *not* a product distribution.) Choose a matroid $\mathbf{M}_{\mathcal{B}} \in \mathcal{M}$ uniformly at random and let the target function be $f^* = \text{rank}_{\mathbf{M}_{\mathcal{B}}}$. Clearly $\mathcal{ALG}$ does not know $\mathcal{B}$.

Assume that $\mathcal{ALG}$ uses a set $\mathcal{S}$ of $\ell$ training examples. For any $A \in \mathcal{A}$ that is not a training example, the algorithm $\mathcal{ALG}$ has *no information* about $f^*(A)$; in particular, the conditional distribution of its value, given $\mathcal{S}$, remains uniform in $\{8t, |A|\}$. So $\mathcal{ALG}$ cannot determine its value better than randomly guessing between the two possible

values $8t$ and $|A|$. The set of nontraining examples has measure $1 - 2^{-t+\log \ell}$. Thus

$$\mathbf{E}_{f^*,\mathcal{S}}\left[\Pr_{A\sim D}\left[f^*(A) \notin \left[f(A), \tfrac{n^{1/3}}{16t} f(A)\right]\right]\right] \;\geq\; \frac{1 - 2^{-t+\log \ell}}{2} \;\geq\; 7/16.$$

Therefore, there exists $f^*$ such that

$$\Pr_{\mathcal{S}}\left[\Pr_{A\sim D}\left[f^*(A) \notin \left[f(A), \tfrac{n^{1/3}}{16t} f(A)\right]\right] \geq 1/4\right] \;\geq\; 1/8.$$

That is, there exists $f^*$ such that with probability at least $1/8$ (over the draw of the training samples) we have that the hypothesis function $f$ output by $\mathcal{ALG}$ does not approximate $f^*$ within a $o(n^{1/3}/\log n)$ factor on at least $1/4$ fraction of the examples under $D$. ∎

We can further show that the lower bound in Theorem 8 holds even if the algorithm is told the underlying distribution, even if the algorithm can query the function on inputs of its choice, and even if the queries are adaptive. In other words, this inapproximability still holds in the PMAC model augmented with value queries. Specifically, we have the following.

THEOREM 9. *Let $\mathcal{ALG}$ be an arbitrary learning algorithm that uses only a polynomial number of training examples, which can be either drawn i.i.d. from the underlying distribution or value queries. There exists a distribution $D$ and a submodular target function $f^*$ such that, with probability at least $1/4$ (over the draw of the training samples), the hypothesis function output by $\mathcal{ALG}$ does not approximate $f^*$ within an $o(n^{1/3}/\log n)$ factor on at least a $1/4$ fraction of the examples under $D$. This holds even for the subclass of matroid rank functions.*

Theorem 8 is an information-theoretic hardness result. A slight modification yields Corollary 2, which is a complexity-theoretic hardness result.

COROLLARY 2. *Suppose one-way functions exist. For any constant $\epsilon > 0$, no algorithm can PMAC-learn the class of nonnegative, monotone, submodular functions with approximation factor $O(n^{1/3-\epsilon})$, even if the functions are given by polynomial-time algorithms computing their value on the support of the distribution.*

The proofs of Theorem 9 and Corollary 2 are given in Appendix C.3. The lower bound in Corollary 2 gives a family of submodular functions that are hard to learn, even though the functions can be evaluated by polynomial-time algorithms on the *support of the distribution*. However, we do not prove that the functions can be evaluated by polynomial-time algorithms at *arbitrary points*, and we leave it as an open question whether such a construction is possible.

**4.4. An $O(\sqrt{n})$-approximation algorithm.** In this section we discuss our most general upper bound for efficiently PMAC-learning the class of nonnegative, monotone, submodular functions with an approximation factor of $O(\sqrt{n})$.

We start with a useful structural lemma concerning submodular functions.

LEMMA 3 (Goemans et al. [33]). *Let $f : 2^{[n]} \to \mathbb{R}_+$ be a normalized, nonnegative, monotone, submodular function. Then there exists a function $\hat{f}$ of the form $\hat{f}(S) = \sqrt{w^T \chi(S)}$ where $w \in \mathbb{R}^n_+$ such that for all $S \subseteq [n]$ we have*

$$\hat{f}(S) \;\leq\; f(S) \;\leq\; \sqrt{n}\hat{f}(S).$$

This result, proven by Goemans et al. [33], follows from properties of submodular polyhedra and John's theorem on approximating centrally symmetric convex bodies by ellipsoids [52]. We now use it in proving our main algorithmic result.

**Algorithm 2.** Algorithm for PMAC-learning the class of nonnegative, monotone, submodular functions.

**Input:** A sequence of labeled training examples $\mathcal{S} = \{(S_1, f^*(S_1)), (S_2, f^*(S_2)), \ldots (S_\ell, f^*(S_\ell))\}$.

- Let $\mathcal{S}_{\neq 0} = \{(A_1, f^*(A_1)), \ldots, (A_a, f^*(A_a))\}$ be the subsequence of $\mathcal{S}$ with $f^*(A_i) \neq 0 \ \ \forall i$. Let $\mathcal{S}_0 = \mathcal{S} \setminus \mathcal{S}_{\neq 0}$. Let $\mathcal{U}_0$ be the set of indices defined as

$$\mathcal{U}_0 \ = \ \bigcup_{\substack{i \leq \ell \\ f^*(S_i)=0}} S_i.$$

- For each $i \in [a]$, let $y_i$ be the outcome of flipping a fair $\{+1, -1\}$-valued coin, each coin flip independent of the others. Let $x_i \in \mathbb{R}^{n+1}$ be the point defined by

$$x_i \ = \ \begin{cases} \left( \chi(A_i),\, {f^*}^2(A_i) \right) & (\text{if } y_i = +1), \\ \left( \chi(A_i),\, (n+1) \cdot {f^*}^2(A_i) \right) & (\text{if } y_i = -1). \end{cases}$$

- Find a linear separator $u = (w, -z) \in \mathbb{R}^{n+1}$, where $w \in \mathbb{R}^n$ and $z \in \mathbb{R}$, such that $u$ is consistent with the labeled examples $(x_i, y_i) \ \ \forall i \in [a]$, and with the additional constraint that $w_j = 0 \ \ \forall j \in \mathcal{U}_0$.

**Output:** The function $f$ defined as $f(S) = \left( \frac{w^\mathsf{T} \chi(S)}{(n+1)z} \right)^{1/2}$.

THEOREM 10. *Let $\mathcal{F}$ be the class of nonnegative, monotone, submodular functions over $\mathcal{X} = 2^{[n]}$. There is an algorithm that PMAC-learns $\mathcal{F}$ with approximation factor $\sqrt{n+1}$. That is, for any distribution $D$ over $\mathcal{X}$, for any $\epsilon$, $\delta$ sufficiently small, with probability $1 - \delta$, the algorithm produces a function $f$ that approximates $f^*$ within a multiplicative factor of $\sqrt{n+1}$ on a set of measure $1 - \epsilon$ with respect to $D$. The algorithm uses $\ell = \frac{48n}{\epsilon} \log(\frac{9n}{\delta\epsilon})$ training examples and runs in time* $\text{poly}(n, 1/\epsilon, 1/\delta)$.

*Proof.* As in Theorem 7, because of the multiplicative error allowed by the PMAC-learning model, we will separately analyze the subset of the instance space where $f^*$ is zero and the subset of the instance space where $f^*$ is nonzero. For convenience, let us define

$$\mathcal{P} = \{\, S \,:\, f^*(S) \neq 0 \,\} \qquad \text{and} \qquad \mathcal{Z} = \{\, S \,:\, f^*(S) = 0 \,\}.$$

The main idea of our algorithm is to reduce our learning problem to the standard problem of learning a binary classifier (in fact, a linear separator) from i.i.d. samples in the passive, supervised learning setting [58, 93] with a slight twist in order to handle the points in $\mathcal{Z}$. The problem of learning a linear separator in the passive supervised learning setting is one where the instance space is $\mathbb{R}^m$, the samples are independently drawn from some fixed and unknown distribution $D'$ on $\mathbb{R}^m$, and there is a fixed but unknown target function $c^* : \mathbb{R}^m \to \{-1, +1\}$ defined by $c^*(x) = \text{sgn}(u^\mathsf{T} x)$ for some vector $u \in \mathbb{R}^m$. The examples induced by $D'$ and $c^*$ are called *linearly separable*.

The linear separator learning problem we reduce to is defined as follows. The instance space is $\mathbb{R}^m$ where $m = n + 1$ and the distribution $D'$ is defined by the following procedure for generating a sample from it. Repeatedly draw a sample $S \subseteq [n]$ from the distribution $D$ until $f^*(S) \neq 0$. Next, flip a fair coin. The sample from $D'$ is

$$\begin{array}{ll} \left( \chi(S),\, f^*(S)^2 \right) & (\text{if the coin is heads}), \\ \left( \chi(S),\, (n+1) \cdot f^*(S)^2 \right) & (\text{if the coin is tails}). \end{array} \tag{4.4}$$

The function $c^*$ defining the labels is as follows: samples for which the coin was heads are labeled $+1$, and the others are labeled $-1$.

We claim that the distribution over labeled examples induced by $D'$ and $c^*$ is linearly separable in $\mathbb{R}^m$. To prove this we use Lemma 3, which says that there exists a linear function $\hat{f}(S) = w^{\mathsf{T}}\chi(S)$ such that

$$\hat{f}(S) \;\le\; f^*(S)^2 \;\le\; n \cdot \hat{f}(S) \qquad \forall S \subseteq [n]. \tag{4.5}$$

Let $u = \big( (n+1/2) \cdot w,\, -1 \big) \in \mathbb{R}^m$. For any point $x$ in the support of $D'$ we have

$$\begin{aligned} x \;=\; \big( \chi(S),\, f^*(S)^2 \big) \quad &\Longrightarrow \quad u^{\mathsf{T}}x = (n+\tfrac{1}{2}) \cdot \hat{f}(S) - f^*(S)^2 \;>\; 0, \\ x \;=\; \big( \chi(S),\, (n+1) \cdot f^*(S)^2 \big) \quad &\Longrightarrow \quad u^{\mathsf{T}}x = (n+\tfrac{1}{2}) \cdot \hat{f}(S) - (n+1) \cdot f^*(S)^2 \;<\; 0. \end{aligned}$$

This proves the claim.

Moreover, due to (4.5), the linear function $\hat{f}$ also satisfies $\hat{f}(S) = 0$ for every $S \in \mathcal{Z}$. In particular, every training example $S_i$ satisfies $\hat{f}(S_i) = 0$ whenever $S_i \in \mathcal{Z}$, and moreover

$$\hat{f}(\{j\}) = w_j = 0 \quad \forall j \in \mathcal{U}_D, \qquad \text{where} \qquad \mathcal{U}_D \;=\; \bigcup_{S_i \in \mathcal{Z}} S_i.$$

Our algorithm is now as follows. It first partitions the training set

$$\mathcal{S} \;=\; \{(S_1, f^*(S_1)), \ldots, (S_\ell, f^*(S_\ell))\}$$

into two sets $\mathcal{S}_0$ and $\mathcal{S}_{\neq 0}$, where $\mathcal{S}_0$ is the subsequence of $\mathcal{S}$ with $f^*(S_i) = 0$, and $\mathcal{S}_{\neq 0} = \mathcal{S} \setminus \mathcal{S}_0$. For convenience, let us denote the sequence $\mathcal{S}_{\neq 0}$ as

$$\mathcal{S}_{\neq 0} \;=\; \big( \big(A_1, f^*(A_1)\big), \ldots, \big(A_a, f^*(A_a)\big) \big).$$

Note that $a$ is a random variable and we can think of the sets the $A_i$ as drawn independently from $D$, conditioned on belonging to $\mathcal{P}$. Let

$$\mathcal{U}_0 \;=\; \bigcup_{S_i \,:\, f^*(S_i)=0} S_i \qquad \text{and} \qquad \mathcal{L}_0 \;=\; \{\, S \,:\, S \subseteq \mathcal{U}_0 \,\}.$$

Using $\mathcal{S}_{\neq 0}$, the algorithm then constructs a sequence $\mathcal{S}'_{\neq 0} = \big((x_1, y_1), \ldots, (x_a, y_a)\big)$ of training examples for the binary classification problem. For each $i \in [a]$, let $y_i$ be $-1$ or $1$, each with probability $1/2$. Define $x_i$ as in (4.4):

$$x_i \;=\; \begin{cases} \big( \chi(A_i),\, f^*(A_i)^2 \big) & (\text{if } y_i = +1), \\ \big( \chi(A_i),\, (n+1) \cdot f^*(A_i)^2 \big) & (\text{if } y_i = -1). \end{cases}$$

The last step of our algorithm is to solve a linear program in order to find a linear separator $u = (w, -z)$ where $w \in \mathbb{R}^n$, $z \in \mathbb{R}$, and

- $u$ is consistent with the labeled examples $(x_i, y_i)$ for all $i = 1, \ldots, a$, and
- $w_j = 0$ for all $j \in \mathcal{U}_0$.

The output hypothesis is $f(S) = (\frac{w^{\mathsf{T}}\chi(S)}{(n+1)z})^{1/2}$.

To prove correctness, note first that the linear program is feasible; this follows from our earlier discussion using the facts that (1) $\mathcal{S}'_{\neq 0}$ is a set of labeled examples drawn from $D'$ and labeled by $c^*$, and (2) $\mathcal{U}_0 \subseteq \mathcal{U}_D$. It remains to show that $f$

approximates the target on most of the points. Let $\mathcal{Y}$ denote the set of points $S \in \mathcal{P}$ such that both of the points $(\chi(S), f^{*2}(S))$ and $(\chi(S), (n+1) \cdot f^{*2}(S))$ are correctly labeled by $\mathrm{sgn}(u^{\mathsf{T}}x)$, the linear separator found by our algorithm. It is easy to see that the function $f$ approximates $f^*$ to within a factor $\sqrt{n+1}$ on all the points in the set $\mathcal{Y}$: for any point $S \in \mathcal{Y}$, we have

$$w^{\mathsf{T}}\chi(S) - z f^*(S)^2 > 0 \qquad \text{and} \qquad w^{\mathsf{T}}\chi(S) - z(n+1) f^*(S)^2 < 0$$
$$\implies \quad \left(\frac{w^{\mathsf{T}}\chi(S)}{(n+1)z}\right)^{1/2} \;<\; f^*(S) \;<\; \sqrt{n+1}\left(\frac{w^{\mathsf{T}}\chi(S)}{(n+1)z}\right)^{1/2}.$$

So, for any point in $S \in \mathcal{Y}$, the function $f(S) = (\frac{w^{\mathsf{T}}\chi(S)}{(n+1)z})^{1/2}$ approximates $f^*$ to within a factor $\sqrt{n+1}$.

Moreover, by design the function $f$ correctly labels as 0 all the examples in $\mathcal{L}_0$. To finish the proof, we now note two important facts: for our choice of $\ell = \frac{16n}{\epsilon}\log(\frac{n}{\delta\epsilon})$, with high probability both $\mathcal{P} \setminus \mathcal{Y}$ and $\mathcal{Z} \setminus \mathcal{L}_0$ have small measure.

Claim 5. *With probability at least $1-\delta$, the set $\mathcal{Z} \setminus \mathcal{L}_0$ has measure at most $\epsilon$ under $D$.*

*Proof.* The proof is very similar to Claim 4. For $1 \le k \le \ell$, define

$$U_k \;=\; \bigcup_{\substack{i \le k \\ f^*(S_i)=0}} S_i \qquad \text{and} \qquad \mathcal{L}_k \;=\; \{\, S \,:\, S \subseteq \mathcal{U}_k \,\}.$$

For any $k$, monotonicity and submodularity imply that $f^*(U_k) = 0$, so monotonicity again implies that $\mathcal{L}_k \subseteq \mathcal{Z}$. Suppose that, for some $k$, the set $\mathcal{Z} \setminus \mathcal{L}_k$ has measure at least $\epsilon$. Define $k' = k + \log(n/\delta)/\epsilon$. Then among the subsequent examples $S_{k+1}, \ldots, S_{k'}$, the probability that none of them lie in $\mathcal{Z} \setminus \mathcal{L}_k$ is at most $(1-\epsilon)^{\log(n/\delta)/\epsilon} \le \delta/n$. On the other hand, if one of them does lie in $\mathcal{Z} \setminus \mathcal{L}_k$, then $|\mathcal{U}_{k'}| > |\mathcal{U}_k|$. But $|\mathcal{U}_k| \le n$ for all $k$, so this can happen at most $n$ times. Since $\ell \ge n\log(n/\delta)/\epsilon$, with probability at least $\delta$ the final set $\mathcal{Z} \setminus \mathcal{L}_\ell = \mathcal{Z} \setminus \mathcal{L}_0$ has measure at most $\epsilon$. ∎

Claim 6. *With probability at least $1-2\delta$, the set $\mathcal{P} \setminus \mathcal{Y}$ has measure at most $2\epsilon$ under $D$.*

*Proof.* Let $q = 1 - p = \Pr_{S\sim D}[\, S \in \mathcal{P}\,]$. If $q < \epsilon$, then the claim is immediate, since $\mathcal{P}$ has measure at most $\epsilon$. So assume that $q \ge \epsilon$. Let $\mu = \mathbf{E}[\, a\,] = q\ell$. By assumption $\mu > 16n\log(n/\delta\epsilon)\frac{q}{\epsilon}$. Then Chernoff bounds give that

$$\Pr\left[\, a < 8n\log(n/\delta\epsilon)\frac{q}{\epsilon}\,\right] \;<\; \exp(-n\log(n/\delta)q/\epsilon) \;<\; \delta.$$

So with probability at least $1-\delta$, we have $a \ge 8n\log(qn/\delta\epsilon)\frac{q}{\epsilon}$. By a standard sample complexity argument [93] (which we reproduce in Theorem 25 in Appendix A.2), with probability at least $1-\delta$, any linear separator consistent with $\mathcal{S}'$ will be inconsistent with the labels on a set of measure at most $\epsilon/q$ under $D'$. In particular, this property holds for the linear separator computed by the linear program. So for any set $S$, the conditional probability that either $(\chi(S), f^*(S)^2)$ or $(\chi(S), (n+1)\cdot f^*(S)^2)$ is incorrectly labeled, given that $S \in \mathcal{P}$, is at most $2\epsilon/q$. Thus

$$\Pr[\, S \in \mathcal{P} \;\wedge\; S \notin \mathcal{Y}\,] \;=\; \Pr[\, S \in \mathcal{P}\,] \cdot \Pr[\, S \notin \mathcal{Y} \;\mid\; S \in \mathcal{P}\,] \;\le\; q \cdot (2\epsilon/q),$$

as required. ∎

In summary, our algorithm produces a hypothesis $f$ that approximates $f^*$ to within a factor $\sqrt{n+1}$ on the set $\mathcal{Y} \cup \mathcal{L}_\ell$. The complement of this set is $(\mathcal{Z} \setminus \mathcal{L}_\ell) \cup (\mathcal{P} \setminus \mathcal{Y})$, which has measure at most $3\epsilon$, with probability at least $1 - 3\delta$. ∎

*Remark.* Our algorithm proving Theorem 10 is significantly simpler than the algorithm of Goemans et al. [33], which achieves a slightly worse approximation factor in the model of approximately learning everywhere with value queries.

**4.4.1. Extensions.** Our algorithm for learning submodular functions is quite robust and can be extended to handle more general scenarios, including forms of noise. In this section we discuss several such extensions.

It is clear from the proofs of Theorem 10 that any improvements in the approximation factor for approximating submodular functions by linear functions (i.e., Lemma 3) for specific subclasses of submodular functions yield PMAC-learning algorithms with improved approximation factors.

Next, let us consider the more general case where we do not even assume that the target function is submodular but that it is within a factor $\alpha$ of a submodular function on every point in the instance space. Under this relaxed assumption we are able to achieve the approximation factor $\alpha\sqrt{n+1}$. Specifically, we have the following.

THEOREM 11. *Let $\mathcal{F}$ be the class of nonnegative, monotone, submodular functions over $\mathcal{X} = 2^{[n]}$ and let*

$$\mathcal{F}' \;=\; \{\, f \,:\, \exists g \in \mathcal{F},\; g(S) \le f(S) \le \alpha \cdot g(S) \;\; \forall\; S \subseteq [n] \,\}$$

*for some known $\alpha > 1$. There is an algorithm that PMAC-learns $\mathcal{F}'$ with approximation factor $\alpha\sqrt{n+1}$. The algorithm uses $\ell = \frac{48n}{\epsilon}\log(\frac{9n}{\delta\epsilon})$ training examples and runs in time* $\mathrm{poly}(n, 1/\epsilon, 1/\delta)$.

*Proof.* By assumption, there exists $g \in \mathcal{F}$ such that $g(S) \le f^*(S) \le \alpha \cdot g(S)$. Combining this with Lemma 3, we get that there exists $\hat{f}(S) = w^{\mathsf{T}}\chi(S)$ such that

$$w^{\mathsf{T}}\chi(S) \;\le\; f^{*2}(S) \;\le\; n \cdot \alpha^2 \cdot w^{\mathsf{T}}\chi(S) \qquad \forall\; S \subseteq [n].$$

We then apply the algorithm described in Theorem 10 with the following modifications: (1) in the second step if $y_i = +1$ we set $x_i = (\chi(S), f^{*2}(S))$ and if $y_i = -1$ we set $x_i = (\chi(S), \alpha^2(n+1)\cdot f^*(S))$; (2) we output the function $f(S) = (\frac{1}{\alpha^2(n+1)z} w^{\mathsf{T}}\chi(S))^{1/2}$. It is then easy to show that the distribution over labeled examples induced by $D'$ and $c^*$ is linearly separable in $\mathbb{R}^{n+1}$; in particular, $u = (\alpha^2(n+1/2)\cdot w, -1) \in \mathbb{R}^{n+1}$ defines a good linear separator. The proof then proceeds as in Theorem 10. ∎

We can also extend the result in Theorem 10 to the *agnostic case* where we assume that there exists a submodular function that agrees with the target on all but an $\eta$ fraction of the points; note that on the $\eta$ fraction of the points the target can be arbitrarily far from a submodular function. In this case we can still PMAC-learn with a polynomial number of samples $O(\frac{n}{\epsilon^2}\log(\frac{n}{\delta\epsilon}))$, but using a potentially computationally inefficient procedure.

THEOREM 12. *Let $\mathcal{F}$ be the class of nonnegative, monotone, submodular functions over $\mathcal{X} = 2^{[n]}$. Let*

$$\mathcal{F}' \;=\; \{\, f \,:\, \exists g \in \mathcal{F} \text{ s.t. } f(S) = g(S) \;\; \text{on more than } 1-\eta \text{ fraction of the points} \,\}.$$

*There is an algorithm that PMAC-learns $\mathcal{F}'$ with approximation factor $\sqrt{n+1}$. That is, for any distribution $D$ over $\mathcal{X}$, for any $\epsilon$, $\delta$ sufficiently small, with probability*

*$1-\delta$, the algorithm produces a function $f$ that approximates $f^*$ within a multiplicative factor of $\sqrt{n+1}$ on a set of measure $1-\epsilon-\eta$ with respect to $D$. The algorithm uses $O(\frac{n}{\epsilon^2}\log(\frac{n}{\delta\epsilon}))$ training examples.*

*Proof sketch.* The proof proceeds as in Theorem 10. The main difference is that in the new feature space $\mathbb{R}^m$, the best linear separator has error (fraction of mistakes) $\eta$. It is well known that even in the agnostic case the number of samples needed to learn a separator of error at most $\eta+\epsilon$ is $O(\frac{n}{\epsilon^2}\log(\frac{n}{\delta\epsilon}))$ (see Theorem 26 in Appendix A.2). However, it is NP-hard to minimize the number of mistakes, even approximately [38], so the resulting procedure uses a polynomial number of samples, but it is computationally inefficient. ∎

**5. An approximate characterization of matroid rank functions.** We now present an interesting structural result that is an application of the ideas in section 4.2. The statement is quite surprising: matroid rank functions are very well approximated by *univariate*, concave functions. The proof is also based on Theorem 6. To motivate the result, consider the following easy construction of submodular functions, which can be found in Lovász's survey [70, p. 251]

PROPOSITION 1. *Let $h : \mathbb{R} \to \mathbb{R}$ be concave. Then $f : 2^{[n]} \to \mathbb{R}$ defined by $f(S) = h(|S|)$ is submodular.*

Surprisingly, we now show that a partial converse is true.

THEOREM 13. *There is an absolute constant $c > 1$ such that the following is true. Let $f : 2^{[n]} \to \mathbb{Z}_+$ be the rank function of a matroid with no loops, i.e., $f(S) \geq 1$ whenever $S \neq \emptyset$. Fix any $\epsilon > 0$, sufficiently small. There exists a concave function $h : [0,n] \to \mathbb{R}$ such that, for* every *$k \in [n]$, and for a $1-\epsilon$ fraction of the sets $S \in \binom{[n]}{k}$,*

$$h(k)/(c\log(1/\epsilon)) \;\leq\; f(S) \;\leq c\log(1/\epsilon)h(k).$$

The idea behind this theorem is as follows. For $x \in [0,n]$, we define $h(x)$ to be the expected value of $f$ under the product distribution which samples each element independently with probability $x/n$. The value of $f$ under this distribution is tightly concentrated around $h(x)$, by the results of sections 3.2 and 4.2. For any $k \in [n]$, the distribution defining $h(k)$ is very similar to the uniform distribution on sets of size $k$, so $f$ is also tightly concentrated under the latter distribution. So the value of $f$ for most sets of size $k$ is roughly $h(k)$. The concavity of this function $h$ is a consequence of submodularity of $f$.

Henceforth, we will use the following notation. For $p \in [0,1]$, let $R(p) \subseteq [n]$ denote the random variable obtained by choosing each element of $[n]$ independently with probability $p$. For $k \in [n]$, let $S(k) \subseteq [n]$ denote a set of cardinality $k$ chosen uniformly at random. Define the function $h' : [0,1] \to \mathbb{R}$ by

$$h'(p) \;=\; \mathbf{E}\left[\, f(R(p)) \,\right].$$

For any $\tau \in \mathbb{R}$, define the functions $g_\tau : [0,1] \to \mathbb{R}$ and $g'_\tau : [n] \to \mathbb{R}$ by

$$\begin{aligned} g_\tau(p) &= \Pr\left[\, f(R(p)) > \tau \,\right], \\ g'_\tau(k) &= \Pr\left[\, f(S(k)) > \tau \,\right]. \end{aligned}$$

Finally, let us introduce the notation $X \cong Y$ to denote that random variables $X$ and $Y$ are identically distributed.

LEMMA 4. *$h'$ is concave.*

*Proof.* One way to prove this is by appealing to the *multilinear extension* of $f$, which has been of great value in recent work [13]. This is the function $F : [0,1]^{[n]} \to \mathbb{R}$ defined by $F(y) = \mathbf{E}[\, f(\hat{y})\,]$, where $\hat{y} \in \{0,1\}^{[n]}$ is a random variable obtained by independently setting $\hat{y}_i = 1$ with probability $y_i$, and $\hat{y}_i = 0$ otherwise. Then $h'(p) = F(p, \ldots, p)$. It is known [13] that $\frac{\partial^2 F}{\partial y_i \partial y_j} \leq 0$ for all $i, j$. By basic calculus, this implies that the second derivative of $h'$ is nonpositive, and hence $h'$ is concave. ∎

LEMMA 5. *$g'_\tau$ is a monotonically nondecreasing function.*

*Proof.* Fix $k \in [n-1]$ arbitrarily. Pick a set $S = S(k)$. Construct a new set $T$ by adding to $S$ a uniformly chosen element of $V \setminus S$. By monotonicity of $f$ we have $f(S) > \tau \implies f(T) > \tau$. Thus $\Pr[\, f(S) > \tau\,] \leq \Pr[\, f(T) > \tau\,]$. Since $T \cong S(k+1)$, this implies that $g_\tau(k) \leq g_\tau(k+1)$, as required. ∎

LEMMA 6. *$g'_\tau(k) \leq 2 \cdot g_\tau(k/n)$ for all $\tau \in \mathbb{R}$ and $k \in [n]$.*

*Proof.* This lemma is reminiscent of a well-known property of the Poisson approximation [73, Theorem 5.10], and the proof is also similar. Let $p = k/n$. Then

$$\begin{aligned}
g_\tau(p) &= \Pr[\, f(R(p)) > \tau\,] \\
&= \sum_{i=0}^{n} \Pr[\, f(R(p)) > \tau \mid |R(p)| = i\,] \cdot \Pr[\, |R(p)| = i\,] \\
&= \sum_{i=0}^{n} g'_\tau(i) \cdot \Pr[\, |R(p)| = i\,] \\
&\geq \sum_{i=k}^{n} g'_\tau(k) \cdot \Pr[\, |R(p)| = i\,] \qquad \text{(by Lemma 5)} \\
&= g'_\tau(k) \cdot \Pr[\, |R(p)| \geq k\,] \\
&\geq g'_\tau(k)/2,
\end{aligned}$$

since the mean $k$ of the binomial distribution $B(n, k/n)$ is also a median. ∎

*Proof of* Theorem 13. For $x \in [0,n]$, define $h(x) = h'(x/n) = \mathbf{E}[\, f(R(x/n))\,]$. Fix $k \in [n]$ arbitrarily. We use the same constants $L < M < H$ and $K$ as in section 4.2.

*Case* 1. Suppose that $h(k) \geq L \log(1/\epsilon)$. As argued in (4.2), since $L \geq 4000$ and $\epsilon \leq 1/2$ we have

$$\Pr\left[ f(R(k/n)) < \frac{1}{3} h(k) \right] \leq \epsilon \qquad \text{and} \qquad \Pr\left[ f(R(k/n)) > \frac{5}{3} h(k) \right] \leq \epsilon.$$

By Lemma 6, $\Pr\left[\, f(S(k)) > \frac{5}{3} h(k)\,\right] \leq 2\epsilon$. By a symmetric argument, which we omit, one can show that $\Pr\left[\, f(S(k)) < \frac{1}{3} h(k)\,\right] \leq 2\epsilon$. Thus,

$$\Pr\left[\; \tfrac{1}{3} h(k) \leq f(S(k)) \leq \tfrac{5}{3} h(k) \;\right] \geq 1 - 4\epsilon.$$

This completes the proof of Case 1.

*Case* 2. Suppose that $h(k) < L \log(1/\epsilon)$. This immediately implies that

$$\Pr\left[ f(S(k)) < \frac{h(k)}{L \log(1/\epsilon)} \right] \leq \Pr[\, f(S(k)) < 1\,] = 0 \tag{5.1}$$

since $k \geq 1$ and since we assume that $f(S) \geq 1$ whenever $S \neq \emptyset$. These same assumptions lead to the following lower bound on $h$:

$$h(k) \;\geq\; \Pr\left[\, f(R(k/n)) \geq 1 \,\right] \;=\; \Pr\left[\, R(k/n) \neq \emptyset \,\right] \;=\; 1-(1-k/n)^n \;\geq\; 1-1/e. \tag{5.2}$$

Thus

$$\begin{aligned}
&\Pr\left[\, f(S(k)) > \big(2K\log(1/\epsilon)\big) \cdot h(k) \,\right] \\
&\quad\leq\; 2 \cdot \Pr\left[\, f(R(k/n)) > \big(2K\log(1/\epsilon)\big) \cdot h(k) \,\right] \qquad \text{(by Lemma 6)} \\
&\quad\leq\; 2 \cdot \Pr\left[\, f(R(k/n)) > K\log(1/\epsilon) \,\right] \qquad \text{(by (5.2))} \\
&\quad\leq\; 2 \cdot \epsilon,
\end{aligned}$$

by Lemma 2, statement (3), since $\mathbf{E}\left[\, f(R(k/n)) \,\right] = h(k) < L\log(1/\epsilon)$. Thus,

$$\Pr\left[\; \frac{h(k)}{L\log(1/\epsilon)} \;\leq\; f(S(k)) \;\leq\; \big(2K\log(1/\epsilon)\big)h(k) \;\right] \;\geq\; 1 - 2\epsilon,$$

completing the proof of Case 2. ∎

**6. Implications of our matroid construction for submodular optimization.** The original motivation of our matroid construction in section 4.3 is to show hardness of learning in the PMAC model. In this section we show that this construction has implications beyond learning theory; it reveals interesting structure of matroids and submodular functions. We illustrate this interesting structure by using it to show strong inapproximability results for several submodular optimization problems.

**6.1. Submodular minimization under a cardinality constraint.** Minimizing a submodular function is a fundamental problem in combinatorial optimization. Formally, the problem is

$$\min\left\{\; f(S) \;:\; S \subseteq [n] \;\right\}. \tag{6.1}$$

There exist efficient algorithms to solve this problem exactly.

THEOREM 14 (Grötschel, Lovász, and Schrijver [35], Iwata, Fleischer, and Fujishige [45], Schrijver [85]). *Let $f : 2^{[n]} \to \mathbb{R}$ be any submodular function.*

(a) *There is an algorithm with running time* $\mathrm{poly}(n)$ *that computes the minimum value of* (6.1).

(b) *There is an algorithm with running time* $\mathrm{poly}(n)$ *that constructs a lattice which represents all minimizers of* (6.1). *This lattice can be represented in space* $\mathrm{poly}(n)$.

The survey of McCormick [71, section 5.1] contains further discussion about algorithms to construct the lattice of minimizers. This lattice efficiently encodes a lot of information about the minimizers. For example, given any set $S \subseteq [n]$, one can use the lattice to efficiently determine whether $S$ is a minimizer of (6.1). Also, the lattice can be used to efficiently find the inclusionwise-minimal and the inclusionwise-maximal minimizer of (6.1). In summary, submodular function minimization is a very tractable optimization problem, and its minimizers have a rich combinatorial structure.

The submodular function minimization problem becomes much harder when we impose some simple constraints. In this section we consider submodular function minimization under a cardinality constraint:

$$\min\left\{\; f(S) \;:\; S \subseteq [n], |S| \geq d \;\right\}. \tag{6.2}$$

This problem, which was considered in previous work [89], is a minimization variant of submodular function maximization under a cardinality constraint [77] and is a submodular analogue of the minimum coverage problem [94]. Unfortunately, (6.2) is not a tractable optimization problem. We show that, in a strong sense, its minimizers are very unstructured.

The main result of this section is that the minimizers of (6.2) do not have a succinct, approximate representation.

THEOREM 15. *There exists a randomly chosen nonnegative, monotone, submodular function* $f : 2^{[n]} \to \mathbb{R}$ *such that, for any algorithm that performs any number of queries to* $f$ *and outputs a data structure of size* $\mathrm{poly}(n)$*, that data structure cannot represent the minimizers of* (6.2) *to within an approximation factor* $o(n^{1/3}/\log n)$*. Moreover, any algorithm that performs* $\mathrm{poly}(n)$ *queries to* $f$ *cannot compute the minimum value of* (6.2) *to within a* $o(n^{1/3}/\log n)$ *factor.*

Here, a "data structure representing the minimizers to within a factor $\alpha$" is a program of size $\mathrm{poly}(n)$ that, given a set $S$, returns "yes" if $S$ is a minimizer, returns "no" if $f(S)$ is at least $\alpha$ times larger than the minimum, and otherwise can return anything.

Previous work [34, 89, 33] showed that there exists a randomly chosen nonnegative, monotone, submodular function $f : 2^{[n]} \to \mathbb{R}$ such that any algorithm that performs $\mathrm{poly}(n)$ queries to $f$ cannot approximate the minimum value of (6.2) to within a $o(n^{1/2}/\log n)$ factor. Also, implicit in the work of Jensen and Korte [51, p. 186] is the fact that no data structure of size $\mathrm{poly}(n)$ can *exactly* represent the minimizers of (6.2). In contrast, Theorem 15 is much stronger because it implies that no data structure of size $\mathrm{poly}(n)$ can even *approximately* represent the minimizers of (6.2).

To prove Theorem 15 we require the matroid construction of section 3.1.4, which we restate as follows.

THEOREM 16. *Let* $n$ *be a sufficiently large integer and let* $h(n)$ *be any function satisfying* $h(n) = o(n^{1/3}/\log n)$*. Define* $k = n^{h(n)} + 1$*,* $d = n^{1/3}$*,* $b = 8\log k$*,* $\tau = d/4\log k$*, and* $\epsilon = 2\log(k)/n^{1/3}$*.*

*Set* $U = \{u_1, \ldots, u_k\}$ *and* $V = \{v_1, \ldots, v_n\}$*. Suppose that* $H = (U \cup V, E)$ *is a* $(d, L, \epsilon)$*-lossless expander. We construct a family* $\mathcal{A} = \{A_1, \ldots, A_k\}$ *of subsets of* $[n]$*, each of size* $d$*, by setting*

$$A_i \;=\; \{\, j \in [n] \,:\, v_j \in \Gamma(\{u_i\}) \,\} \qquad \forall i = 1, \ldots, k. \tag{6.3}$$

*As before,* $\Gamma(J)$ *denotes the neighbors of the vertex set* $J \subseteq U$*.*

*For every* $\mathcal{B} \subseteq U$ *there is a matroid* $\mathbf{M}_{\mathcal{B}} = ([n], \mathcal{I})$ *whose rank function satisfies*

$$\mathrm{rank}_{\mathbf{M}_{\mathcal{B}}}(A_i) \;=\; \begin{cases} b & \text{(if } u_i \in \mathcal{B}\text{)}, \\ d & \text{(if } u_i \in U \setminus \mathcal{B}\text{)}. \end{cases}$$

*Furthermore, every set* $S \subseteq [n]$ *with* $|S| \geq b$ *has* $\mathrm{rank}_{\mathbf{M}_{\mathcal{B}}}(S) \geq b$*.*

*Proof of* Theorem 6.2. Fix any function $h(n) = \omega(1)$. Let $d = n^{1/3}$, $k = n^{h(n)}+1$, and let $U = \{u_1, \ldots, u_k\}$ be a set of vertices. Pick a subset $\mathcal{B} \subseteq U \setminus \{u_k\}$ randomly. We now define a submodular function on the ground set $[n]$. We apply Theorem 5 with $L = d/2\log k$ and $\epsilon = 1/L$ to obtain a random bipartite multigraph $H$. With probability at least $1 - 2/k$, the resulting graph $H$ is a $(d, L, \epsilon)$-lossless expander (after eliminating parallel edges). In this case, we can apply Theorem 16 to obtain the matroid $\mathbf{M}_{\mathcal{B}}$, which we emphasize does not depend on $\Gamma(\{u_k\})$. We let $f = \mathrm{rank}_{\mathbf{M}_{\mathcal{B}}}$.

Define $A_i$ as in (6.3) for $i = 1, \ldots, k-1$. The last sentence of Theorem 16 ensures that $f(S) \geq b$ whenever $|S| \geq b$, and therefore $A_i$ is a minimizer of (6.2) whenever $u_i \in \mathcal{B}$.

First consider an algorithm $\mathcal{ALG}$ that performs any number of queries to $f$ and attempts to represent $\mathcal{B}$ in poly$(n)$ bits. Since $\mathcal{B}$ is a random subset of $U \setminus \{u_k\}$, which has cardinality $n^{h(n)}$, the probability that $\mathcal{B}$ can be represented in poly$(n)$ bits is $o(1)$. If $\mathcal{B}$ cannot be exactly represented by $\mathcal{ALG}$, then, with probability $1/2$, there is some set $A_i$ whose value is not correctly represented. The multiplicative error in the value of $A_i$ is $d/b = \Theta(n^{1/3}/h(n) \log n)$. This proves the first statement of the theorem.

Next we will argue that any algorithm $\mathcal{ALG}$ performing $m = \text{poly}(n)$ queries to $f = \text{rank}_{\mathbf{M}_\mathcal{B}}$ has low probability of determining whether $\mathcal{B} = \emptyset$. If $\mathcal{B} = \emptyset$, then $f(S) = \min\{|S|, d\}$, so the minimum value of (6.2) is $d = n^{1/3}$. On the other hand, if $\mathcal{B} \neq \emptyset$, then the minimum value of (6.2) is $b = O(h(n) \log n)$. Therefore this will establish the second part of the theorem.

Suppose the algorithm $\mathcal{ALG}$ queries the value of $f$ on the sets $S_1, \ldots, S_m \subseteq [n]$. Consider the $i$th query and suppose inductively that $\text{rank}_{\mathbf{M}_\mathcal{B}}(S_j) = \text{rank}_{\mathbf{M}_\emptyset}(S_j)$ for all $j < i$. Thus $\mathcal{ALG}$ has not yet distinguished between the cases $f = \text{rank}_{\mathbf{M}_\mathcal{B}}$ and $f = \text{rank}_{\mathbf{M}_\emptyset}$. Consequently the set $S_i$ used in the $i$th query is independent of $A_1, \ldots, A_{k-1}$.

Let $S_i'$ be a set of size $|S_i'| = d$ obtained from $S_i$ by either adding (if $|S_i| < d$) or removing (if $|S_i| > d$) arbitrary elements of $[n]$, or setting $S_i' = S_i$ if $|S_i| = d$. We will apply Theorem 5 again, but this time we make an additional observation. Since the definition of expansion does not depend on the labeling of the ground set, one may assume in Theorem 5 that one vertex in $U$, say, $u_k$, chooses its neighbors deterministically and that all remaining vertices in $U$ choose their neighbors at random. Specifically, we will set

$$\Gamma(\{u_k\}) \;=\; \{\, v_j \,:\, j \in S_i' \,\}.$$

The neighbors $\Gamma(\{u_i\})$ for $i < k$ are not randomly rechosen; they are chosen to be the same as they were in the first invocation of Theorem 5. With probability at least $1 - 2/k$ we again obtain a $(d, L, \epsilon)$-lossless expander, in which case Theorem 16 shows that $\text{rank}_{\mathbf{M}_\mathcal{B}}(S_i') = d = |S_i'|$. That event implies

$$\text{rank}_{\mathbf{M}_\mathcal{B}}(S_i) \;=\; \begin{cases} |S_i| = \text{rank}_{\mathbf{M}_\emptyset}(S_i) & (\text{if } |S_i| < d), \\ d = \text{rank}_{\mathbf{M}_\emptyset}(S_i) & (\text{if } |S_i| \geq d), \end{cases}$$

and hence the inductive hypothesis holds for $i$ as well.

By a union bound over all $m$ queries, the probability of distinguishing whether $B = \emptyset$ is at most $2m/k = o(1)$. ∎

**6.2. Submodular *s*-*t* min cut.** Let $G$ be an undirected graph with edge set $E$ and $n = |E|$. Let $s$ and $t$ be distinct vertices of $G$. A set $C \subseteq E$ is called an $s$-$t$ cut if every $s$-$t$ path intersects $C$. Let $\mathcal{C} \subset 2^E$ be the collection of all $s$-$t$ cuts. The *submodular s-t min cut* problem [50] is

$$\min\{\, f(C) \,:\, C \in \mathcal{C} \,\}, \tag{6.4}$$

where $f : 2^E \to \mathbb{R}$ is a nonnegative, monotone, submodular function.

THEOREM 17 (Jegelka and Bilmes [50]). *Any algorithm for the submodular s-t min cut problem with approximation ratio $o(n^{1/3})$ must perform exponentially many queries to $f$.*

Modifying their result to incorporate our matroid construction in section 4.3, we obtain the following theorem.

THEOREM 18. *Let $d = n^{1/3}$. Let $G$ be a graph with edge set $E$ consisting of $d$ internally-vertex-disjoint $s$-$t$ paths, each of length exactly $n/d$. Assume that $f : 2^E \to \mathbb{R}$ is a nonnegative, monotone, submodular function. For any algorithm that performs any number of queries to $f$ and outputs a data structure of size* $\mathrm{poly}(n)$*, that data structure cannot represent the minimizers of* (6.4) *to within an approximation factor $o(n^{1/3}/\log n)$. Moreover, any algorithm that performs* $\mathrm{poly}(n)$ *queries to $f$ cannot compute the minimum value of* (6.4) *to within a $o(n^{1/3}/\log n)$ factor.*

The proof of this theorem is almost identical to the proof of Theorem 15. All that we require is a slightly different expander construction.

THEOREM 19. *Let $U = \{u_1, \dots, u_k\}$ and $V$ be disjoint vertex sets, where $|V| = n$ and $n$ is a multiple of $d$. Write $V$ as the disjoint union $V = V_1 \cup \cdots \cup V_d$, where each $|V_i| = n/d$.*

*Generate a random bipartite multigraph $H$ with left-vertices $U$ and right-vertices $V$ as follows. The vertex $u_k$ has exactly $d$ neighbors in $V$, chosen deterministically and arbitrarily. For each vertex $u_\ell$ with $\ell \le k-1$, pick exactly one neighbor from each $V_i$, uniformly and independently at random. Thus each vertex in $U$ has degree exactly $d$.*

*Suppose that $k \ge 4$, $L \ge d$, $d \ge \log(k)/\epsilon$, and $n \ge 22Ld/\epsilon$. Then, with probability at least $1 - 2/k$, the multigraph $H$ has no parallel edges and satisfies*

$$\begin{aligned} |\Gamma(\{u\})| &= d \qquad \forall u \in U, \\ |\Gamma(J)| &\ge (1-\epsilon)\cdot d \cdot |J| \qquad \forall J \subseteq U,\ |J| \le L. \end{aligned}$$

*Proof.* The proof is nearly identical to the proof of Theorem 5 in Appendix D. The only difference is in analyzing the probability of a repeat when sampling the neighbors of a set $J \subseteq U$ with $|J| = j$. First consider the case that $u_k \in J$. When sampling the neighbors $\Gamma(J)$, an element $v_i$ is considered a repeat if $v_i \in \{v_1, \dots, v_{i-1}\}$ or if $v_i \in \Gamma(\{u_k\})$. Conditioned on $v_1, \dots, v_{i-1}$, the probability of a repeat is at most $\frac{j+d}{n/d}$. If $u_k \notin J$, then this probability is at most $jd/n$. Consequently, the probability of having more than $\epsilon jd$ repeats is at most

$$\binom{jd}{\epsilon jd}\left(\frac{(j+d)d}{n}\right)^{\epsilon jd} \le \left(\frac{e}{\epsilon}\right)^{\epsilon jd}\left(\frac{(j+d)d}{n}\right)^{\epsilon jd} \le (1/4)^{\epsilon jd}.$$

The last inequality follows from $j + d \le 2L$ and our hypothesis $n \ge 22Ld/\epsilon$. The remainder of the proof is identical to the proof of Theorem 5. ∎

*Proof of* Theorem 18. Let $V_i$ be the edges of the $i$th $s$-$t$ path. The minimal $s$-$t$ cuts are those which choose exactly one edge from each $s$-$t$ path; in other words, they are the transversals of the $V_i$'s. Let $V = V_1 \cup \cdots \cup V_d$; this is also the edge set of the graph $G$.

As in Theorem 15, fix $h(n) = \omega(1)$, $k = n^{h(n)} + 1$, and $U = \{u_1, \dots, u_k\}$. Pick a subset $\mathcal{B} \subseteq U \setminus \{u_k\}$ randomly. We now define a submodular function on the ground set $V = [n]$. We apply Theorem 19 with $L = d$ and $\epsilon = 2\log(k)/d$ to obtain a random bipartite multigraph $H$. With probability at least $1 - 2/k$, the resulting graph $H$ is a $(d, L, \epsilon)$-lossless expander. In this case we can apply Theorem 16 to obtain a matroid $\mathbf{M}_{\mathcal{B}}$. We let $f = \mathrm{rank}_{\mathbf{M}_{\mathcal{B}}}$.

Define $A_i$ as in (6.3) for $i = 1, \ldots, k-1$. Because the expander construction of Theorem 19 ensures that each vertex $u_\ell$ has exactly one neighbor in each $V_i$, the corresponding set $A_\ell = \Gamma(u_\ell)$ is a minimal $s$-$t$ cut. Consequently, $A_i \in \mathcal{C}$. The last sentence of Theorem 16 ensures that $f(S) \geq b$ whenever $|S| \geq b$, and therefore $A_i$ is a minimizer of (6.4) whenever $u_i \in \mathcal{B}$.

First consider an algorithm $\mathcal{ALG}$ that performs any number of queries to $f$ and attempts to represent $\mathcal{B}$ in $\text{poly}(n)$ bits. As in Theorem 15, with probability $1/2$ there is some set $A_i$ whose value is not correctly represented, with multiplicative error $\Theta(n^{1/3}/h(n)\log n)$. This proves the first statement of the theorem.

Next we will argue that any algorithm $\mathcal{ALG}$ performing $m = \text{poly}(n)$ queries to $f$ has low probability of determining whether $\mathcal{B} = \emptyset$. If $\mathcal{B} = \emptyset$, then $f(S) = \min\{|S|, d\}$, so the minimum value of (6.2) is $d = n^{1/3}$, because any $S \in \mathcal{C}$ has $|S| \geq d$. On the other hand, if $\mathcal{B} \neq \emptyset$, then, as mentioned above, $A_i$ is a minimizer whenever $u_i \in \mathcal{B}$. The remainder of the argument that establishes the second statement of the theorem is analogous to Theorem 15, replacing Theorem 5 with Theorem 19. ∎

**6.3. Submodular vertex cover.** Let $G = (V, E)$ be a graph with $n = |V|$. A set $C \subseteq V$ is a vertex cover if every edge has at least one endpoint in $C$. Let $\mathcal{C} \subset 2^V$ be the collection of vertex covers in the graph. The *submodular vertex cover* problem [32, 46] is

$$\min\{\, f(S) \,:\, S \in \mathcal{C} \,\}, \tag{6.5}$$

where $f : 2^V \to \mathbb{R}$ is a nonnegative, submodular function. An algorithm for this problem is said to have *approximation ratio* $\alpha$ if, for any function $f$, it returns a set $S$ for which $f(S) \leq \alpha \cdot \min\{\, f(S) \,:\, S \in \mathcal{C} \,\}$.

THEOREM 20 (Goel et al. [32], Iwata and Nagano [46]). *There is an algorithm which performs* $\text{poly}(n)$ *queries to* $f$ *and has approximation ratio* 2.

Goel et al. only state that their algorithm is applicable for monotone, submodular functions, but the monotonicity restriction seems to be unnecessary.

THEOREM 21 (Goel et al. [32]). *For any constant* $\epsilon > 0$*, any algorithm for the submodular vertex cover problem with approximation ratio* $2 - \epsilon$ *must perform exponentially many queries to* $f$.

Modifying their result to incorporate our matroid construction in section 4.3, we obtain the following theorem.

THEOREM 22. *Let* $G = (U \cup V, E)$ *be a bipartite graph. Assume that* $f : 2^{U \cup V} \to \mathbb{R}$ *is a nonnegative, monotone, submodular function. Let* $\epsilon \in (0, 1/3)$ *be a constant. For any algorithm that performs any number of queries to* $f$ *and outputs a data structure of size* $\text{poly}(n)$*, that data structure cannot represent the minimizers of* (6.5) *to within an approximation factor better than* $4/3 - \epsilon$*. Moreover, any algorithm that performs* $\text{poly}(n)$ *queries to* $f$ *cannot compute the minimum value of* (6.4) *to within a* $4/3 - \epsilon$ *factor.*

*Proof.* Let $G$ be a graph with vertex set $U \cup V$ such that $|U| = |V| = |E| = n/2$, and where the edges in $E$ form a matching between $U$ and $V$. The minimal vertex covers are those that contain exactly one endpoint of each edge in $E$. Set $k = 2^{\epsilon^2 n/40} + 1$. Let $\mathcal{A} = \{A_1, \ldots, A_k\}$ be a family of sets where

- each $A_i \subseteq U \cup V$,
- $A_k$ is an arbitrary set with $|A_k| = n/2$, and
- $A_1, \ldots, A_{k-1}$ are independently and uniformly chosen minimal vertex covers.

For any $i \neq j$, $\mathbf{E}[\,|A_i \cap A_j|\,] = n/4$ and a Chernoff bound shows that

$$\Pr[\,|A_i \cap A_j| > (1+\epsilon)n/4\,] \;\leq\; \exp(-\epsilon^2 n/12).$$

A union bound shows that, with high probability, $|A_i \cap A_j| \leq (1+\epsilon)n/4$ for all $i \neq j$. We now condition on that event.

We now apply Lemma 8 (in Appendix E.2) with each $b_i = b = (3+\epsilon)n/8$ and $d = n/2$. We have

$$\min_{i,j\in[k]} (b_i + b_j - |A_i \cap A_j|) \;\geq\; 2b - (1+\epsilon)n/4 \;=\; 2(3+\epsilon)n/8 - (1+\epsilon)n/4 \;=\; n/2,$$

and therefore the hypotheses of Lemma 8 are satisfied. It follows that for any set $\mathcal{B} \subseteq \mathcal{A}$ the set

$$\mathcal{I}_{\mathcal{B}} \;=\; \{\, I \,:\, |I| \leq d \;\;\wedge\;\; |I \cap A_j| \leq b \quad \forall A_j \in \mathcal{B} \,\}$$

is the family of independent sets of a matroid. Let $f = \mathrm{rank}_{\mathbf{M}_{\mathcal{B}}}$ be the rank function of this matroid. Observe that $f(S) \geq b$ whenever $|S| \geq b$.

Suppose $\mathcal{ALG}$ performs any number of queries to $f$. As in Theorem 15, the set $\mathcal{B}$ has probability $o(1)$ of being representable in $\mathrm{poly}(n)$ bits, in which case there is a minimal vertex cover $A_i$ whose value is not correctly represented with probability $1/2$. The multiplicative error in the value of $A_i$ is

$$\frac{d}{b} \;=\; \frac{n/2}{(3+\epsilon)n/8} \;>\; \frac{4}{3} - \epsilon.$$

This proves the first part of the theorem.

Next we will argue that any algorithm $\mathcal{ALG}$ performing $m = \mathrm{poly}(n)$ queries to $f$ has low probability of determining whether $\mathcal{B} = \emptyset$. If $\mathcal{B} = \emptyset$, then $f(S) = \min\{|S|, d\}$, so the minimum value of (6.4) is $d$. On the other hand, suppose that $\mathcal{B} \neq \emptyset$. As every $C \in \mathcal{C}$ has $|C| \geq n/2$, we must have $f(C) \geq b$, and hence the minimum value of (6.4) is $b$. This will show that the multiplicative error is at least $d/b$, thereby proving the second part of the theorem. The remainder of the proof is analogous to Theorem 15. ∎

**7. Implications to algorithmic game theory and economics.** An important consequence of our matroid construction in section 3.1 is that matroid rank functions do not have a "sketch," i.e., a concise, approximate representation. As matroid rank functions can be shown to satisfy the "gross substitutes" property [75], our work implies that gross substitutes functions do not have a concise, approximate representation. This provides a surprising answer to an open question in economics [8, 9, 10]. In this section we define gross substitutes functions, briefly describe their importance in economics, and formally state the implications of our results for these functions.

Gross substitutes functions play an important role in algorithmic game theory and economics, particularly through their use as valuation functions in combinatorial auctions [18, 36, 80]. Intuitively, in a gross substitutes valuation, increasing the price of certain items cannot reduce the demand for items whose price has not changed. Formally, we have the following.

Definition 4. *For price vector $\vec{p} \in \mathbb{R}^n$, the demand correspondence $\mathcal{D}_f(\vec{p})$ of valuation $f$ is the collection of preferred sets at prices $\vec{p}$, i.e.,*

$$\mathcal{D}_f(\vec{p}) \;=\; \operatorname*{argmax}_{S\subseteq\{1,\ldots,n\}} \left\{ f(S) - \textstyle\sum_{j\in S} p_j \right\}.$$

*A function $f$ is gross substitutes if for any price vector $\vec{q} \geq \vec{p}$ (i.e., for which $q_i \geq p_i$ for all $i \in [n]$), and any $A \in \mathcal{D}_f(\vec{p})$ there exists $A' \in \mathcal{D}_f(\vec{q})$ with $A' \supseteq \{i \in A : p_i = q_i\}$.*

In other words, the gross substitutes property requires that all items $i$ in some preferred set $A$ at the old prices $\vec{p}$ and for which the old and new prices are equal ($p_i = q_i$) are simultaneously contained in some preferred set $A'$ at the new prices $\vec{q}$.

Gross substitutes valuations (introduced by Kelso and Crawford [59]) enjoy several appealing structural properties whose implications have been extensively studied by many researchers [8]. For example, given bidders with gross substitutes valuations, simple item-price ascending auctions can be used for determining the socially efficient allocation. As another example, the gross substitute condition is actually necessary for important economic conclusions. For example, Gul and Stacchetti [36] and Milgrom [72] showed that given any valuation that is not gross substitutes, one can specify very simple valuations for the other agents to create an economy in which no Walrasian equilibrium exists.

One important unsolved question concerns the complexity of describing gross substitutes valuations. Several researchers have asked whether there exists a "succinct" representation for such valuations [8], [9, section 6.2.1], [10, section 2.2]. In other words, can a bidder disclose the exact details of his valuation without conveying an exceptionally large amount of information? An implication of our work is that the answer to this question is "no," in a very strong sense. Our work implies that gross substitutes functions cannot be represented succinctly, even approximately, and even with a large approximation factor. Formally, we have the following.

DEFINITION 5. *We say that $g : 2^{[n]} \to \mathbb{R}_+$ is an $\alpha$-sketch for $f : 2^{[n]} \to \mathbb{R}_+$ if $g$ can be represented in* $\mathrm{poly}(n)$ *space and for every set $S$ we have that $f(S)/\alpha \leq g(S) \leq f(S)$.*

As matroid rank functions are known to satisfy the gross substitute property [75], our work implies that gross substitutes do not have a concise, approximate representation. Specifically, we have the following.

THEOREM 23. *Gross substitutes functions do not admit $o(n^{1/3}/\log n)$ sketches.*

**8. Conclusions.** In this work we have used a learning theory perspective to uncover new structural properties of submodular functions. We have presented the first algorithms and lower bounds for learning submodular functions in a distributional learning setting. We also presented numerous implications of our work in algorithmic game theory, economics, matroid theory, and combinatorial optimization.

Regarding learnability, we presented polynomial upper and lower bounds on the approximation factor achievable when using only a polynomial number of examples drawn i.i.d. from an arbitrary distribution. We also presented a simple algorithm achieving a constant-factor approximation under product distributions. These results show that, with respect to product distributions, submodular functions behave in a fairly simple manner, whereas with respect to general distributions, submodular functions behave in a much more complex manner.

We constructed a new family of matroids with interesting technical properties in order to prove our lower bound on PMAC-learnability. The existence of these matroids also resolves an open question in economics: an immediate corollary of our construction is that gross substitutes functions have no succinct, approximate representation. We also used these matroids to show that the optimal solutions of various submodular optimization problems can have a very complicated structure.

The PMAC model provides a new approach for analyzing the learnability of real-valued functions. This paper has analyzed submodular functions in the PMAC model. We believe that it will be interesting to study PMAC-learnability of other classes

of real-valued functions. Indeed, as discussed below, subsequent work has already studied subadditive and XOS functions in the PMAC model.

One technical question left open by this work is determining the precise approximation factor achievable for PMAC-learning submodular functions—there is a gap between the $O(n^{1/2})$ upper bound in Theorem 10 and the $\tilde{\Omega}(n^{1/3})$ lower bound in Theorem 8. We suspect that the lower bound can be improved to $\tilde{\Omega}(n^{1/2})$. If such an improved lower bound is possible, the matroids or submodular functions used in its proof are likely to be very interesting.

**8.1. Subsequent work.** Following our work, several authors have provided further results for learning submodular functions in a distributional learning setting.

Balcan et al. [5] and Badanidiyuru et al. [3] have provided further learnability results in the PMAC model for various classes of set functions commonly used in algorithmic game theory and economics. Building on our algorithmic technique, Balcan et al. [5] give a computationally efficient algorithm for PMAC-learning subadditive functions to within an $\tilde{O}(\sqrt{n})$ factor. They also provide new target-dependent learnability result for XOS (or fractionally subadditive) functions. Their algorithms use the algorithmic technique that we develop in section 4.4, together with new structural results for these classes of functions. Badanidiyuru et al. [3] consider the problem of *sketching* subadditive and submodular functions. They show that the existence of such a sketch implies that PMAC-learning to within a factor $\alpha$ is possible if computational efficiency is ignored. As a consequence they obtain (computationally inefficient) algorithms for PMAC-learning to within an $\tilde{O}(\sqrt{n})$ factor for subadditive functions and to within a $1+\epsilon$ factor for both coverage functions and OXS functions.

Regarding inapproximability, Badanidiyuru et al. and Balcan et al. show that XOS (i.e., fractionally subadditive) functions do not have sketches that approximate to within a factor $o(\sqrt{n})$. Consequently, every algorithm for PMAC-learning XOS functions must have approximation factor $\tilde{\Omega}(\sqrt{n})$. The construction used to prove this result is significantly simpler than our construction in section 4.3, because XOS functions are a more expressive class than submodular functions.

Motivated by problems in privacy preserving data analysis, Gupta et al. [37] considered how to perform statistical queries to a data set in order to learn the answers to all statistical queries from a certain class. They showed that this problem can be efficiently solved when the queries are described by a submodular function. One of the technical pieces in their work is an algorithm to learn submodular functions under a product distribution. A main building block of their technique is the algorithm we provide in section 4.2 for learning under a product distribution, and their analysis is inspired by ours. Their formal guarantee is incomparable to ours: it is stronger in that they allow non-Lipschitz and nonmonotone functions, but it is weaker in that they require access to the submodular function via a value oracle, and they guarantee only additive error (assuming the function is appropriately normalized). Moreover, their running time is $n^{\mathrm{poly}\,(1/\epsilon)}$, whereas ours is $\mathrm{poly}\,(n, 1/\epsilon)$. Feldman and Vondrák further refined these results in a series of papers [28, 27].

Cheraghchi et al. [17] study the noise stability of submodular functions. As a consequence they obtained an algorithm for learning a submodular function under product distributions. Their algorithm also works for nonsubmodular and non-Lipschitz functions and only requires access to the submodular function via statistical queries, though the running time is $n^{\mathrm{poly}\,(1/\epsilon)}$. Their algorithm is agnostic (meaning that they do not assume the target function is submodular), and their performance guarantee proves that the $L_1$-loss of their hypothesis is at most $\epsilon$ more than the best

error achieved by any submodular function (assuming the function is appropriately normalized).

Raskhodnikova and Yaroslavtsev [84] consider learnability of integer-valued, submodular functions and prove that any submodular function $f : \{0,1\}^n \to \{0,1,\ldots,k\}$ can be represented as a pseudo-Boolean $2k$-DNF formula. They use this to provide an algorithm for learning such functions using membership queries under the uniform distribution; the algorithm runs in time polynomial in $\text{poly}(n, k^{O(k \log k/\epsilon)}, 1/\epsilon)$.

Iyer, Jegelka, and Bilmes [48] show that for submodular functions with good curvature, our general PMAC learnability results can be further improved in a target-dependent manner.

Balcan, Vitercik, and White [6] further explored a variant of our PMAC learning model where the goal is to learn an underlying combinatorial function up to pairwise comparisons, from pairwise comparisons. They present several general algorithms that can be used to learn over a variety of combinatorial function classes, including those consisting of submodular, XOS, subadditive, coverage, and Fourier sparse functions.

## Appendix A. Standard facts.

### A.1. Submodular functions.

THEOREM 24. *Given a finite universe $U$, let $S_1, S_2, \ldots, S_n$ be subsets of $U$. Define $f : 2^{[n]} \to \mathbb{R}_+$ by*

$$f(A) = |\cup_{i \in A} S_i| \quad \text{for} \quad A \subseteq [n].$$

*Then $f$ is monotone and submodular. More generally, for any nonnegative weight function $w : U \to \mathbb{R}_+$, the function $f$ defined by*

$$f(A) = w\left(\cup_{i \in A} S_i\right) \quad \text{for} \quad A \subseteq [n]$$

*is monotone and submodular.*

LEMMA 7. *The minimizers of any submodular function are closed under union and intersection.*

*Proof.* Assume that $J_1$ and $J_2$ are minimizers for $f$. By submodularity we have

$$f(J_1) + f(J_2) \geq f(J_1 \cap J_2) + f(J_1 \cup J_2).$$

We also have

$$f(J_1 \cap J_2) + f(J_1 \cup J_2) \geq f(J_1) + f(J_2),$$

so $f(J_1) = f(J_2) = f(J_1 \cap J_2) = f(J_1 \cup J_2)$, as desired. ∎

**A.2. Sample complexity results.** We state here several known sample complexity bounds that were used for proving the results in section 4.4. See, e.g., [19, 2].

THEOREM 25. *Let $C$ be a set of functions from $\mathcal{X}$ to $\{-1, 1\}$ with finite VC-dimension $D \geq 1$. Let $D$ be an arbitrary but fixed probability distribution over $\mathcal{X}$ and let $c^*$ be an arbitrary target function. For any $\epsilon$, $\delta > 0$, if we draw a sample $\mathcal{S}$ from $D$ of size*

$$m(\epsilon, \delta, D) = \frac{1}{\epsilon}\left(4D \log\left(\frac{1}{\epsilon}\right) + 2\log\left(\frac{2}{\delta}\right)\right),$$

*then with probability $1 - \delta$, all hypotheses with error $\geq \epsilon$ are inconsistent with the data; i.e., uniformly for all $h \in C$ with $\text{err}(h) \geq \epsilon$, we have $\widehat{\text{err}}(h) > 0$. Here $\text{err}(h) = \Pr_{x \sim D}[h(x) \neq c^*(x)]$ is the true error of $h$ and $\widehat{\text{err}}(h) = \Pr_{x \sim \mathcal{S}}[h(x) \neq c^*(x)]$ is the empirical error of $h$.*

THEOREM 26. *Suppose that $C$ is a set of functions from $\mathcal{X}$ to $\{-1,1\}$ with finite VC-dimension $D \geq 1$. For any distribution $D$ over $\mathcal{X}$, any target function (not necessarily in $C$), and any $\epsilon$, $\delta > 0$, if we draw a sample from $D$ of size*

$$m(\epsilon, \delta, D) = \frac{64}{\epsilon^2}\left(2D \ln\left(\frac{12}{\epsilon}\right) + \ln\left(\frac{4}{\delta}\right)\right),$$

*then with probability at least $1-\delta$, we have $|\mathrm{err}(h) - \widehat{\mathrm{err}}(h)| \leq \epsilon$ for all $h \in C$.*

## Appendix B. Proofs for concentration of submodular functions.

### B.1. Proof of Theorem 6.

THEOREM 6. *Let $f : 2^{[n]} \to \mathbb{R}_+$ be a nonnegative, monotone, submodular, 1-Lipschitz function. Let the random variable $X \subseteq [n]$ have a product distribution. For any $b, t \geq 0$,*

$$\Pr\left[\, f(X) \leq b - t\sqrt{b} \,\right] \cdot \Pr\left[\, f(X) \geq b \,\right] \;\leq\; \exp(-t^2/4).$$

*Proof.* We begin by observing that the theorem is much easier to prove in the special case that $f$ is integer-valued. Together with our other hypotheses on $f$, this implies that $f$ must actually be a matroid rank function. Whenever $f(S)$ is large, this fact can "certified" by any maximal independent subset of $S$. The theorem then follows easily from a version of Talagrand's inequality which leverages this certification property; see, e.g., [1, section 7.7] or [74, section 10.1].

We now prove the theorem in its full generality. We may assume that $t \leq \sqrt{b}$; otherwise the theorem is trivial, since $f(X)$ is nonnegative. Talagrand's inequality states that for any $\mathcal{A} \subseteq \{0,1\}^n$ and $y \in \{0,1\}^n$ drawn from a product distribution,

$$\Pr\left[\, y \in \mathcal{A} \,\right] \cdot \Pr\left[\, \rho(\mathcal{A}, y) > t \,\right] \;\leq\; \exp(-t^2/4), \tag{B.1}$$

where $\rho$ is a distance function defined by

$$\rho(\mathcal{A}, y) \;=\; \sup_{\substack{\alpha \in \mathbb{R}^n \\ \|\alpha\|_2 = 1}} \min_{z \in \mathcal{A}} \sum_{i \,:\, y_i \neq z_i} \alpha_i.$$

We apply this inequality to the set $\mathcal{A} \subseteq 2^V$ defined by $\mathcal{A} = \{X : f(X) < b - t\sqrt{b}\}$.

CLAIM 7. *For every $Y \subseteq V$, $f(Y) \geq b$ implies $\rho(\mathcal{A}, Y) > t$.*

*Proof.* Suppose to the contrary that $\rho(\mathcal{A}, Y) \leq t$. By relabeling, we can write $Y$ as $Y = \{1, \ldots, k\}$. For $i \in \{0, \ldots, k\}$, let $E_i = \{1, \ldots, i\}$. Define

$$\alpha_i \;=\; \begin{cases} f(E_i) - f(E_{i-1}) & (\text{if } i \in Y), \\ 0 & (\text{otherwise}). \end{cases}$$

Since $f$ is monotone and 1-Lipschitz, we have $0 \leq \alpha_i \leq 1$. Thus $\|\alpha\|_2 \leq \sqrt{\sum_i \alpha_i} \leq \sqrt{f(Y)}$, by nonnegativity of $f$.

The definition of $\rho$ and our supposition $\rho(\mathcal{A}, Y) \leq t$ imply that there exists $Z \in \mathcal{A}$ with

$$\sum_{i \in (Y \setminus Z) \cup (Z \setminus Y)} \alpha_i \;\leq\; \rho(\mathcal{A}, Y) \cdot \|\alpha\|_2 \;\leq\; t\sqrt{f(Y)}. \tag{B.2}$$

We may assume that $Z \subset Y$, since $Z \cap Y$ also satisfies the desired conditions. This follows since monotonicity of $f$ implies that $\alpha \geq 0$ and that $\mathcal{A}$ is downward-closed.

We will obtain a contradiction by showing that $f(Y) - f(Z) \leq t\sqrt{f(Y)}$. First let us order $Y \setminus Z$ as $(\phi(1), \ldots, \phi(m))$, where $\phi(i) < \phi(j)$ iff $i < j$. Next, define $F_i = Z \cup \{\phi(1), \ldots, \phi(i)\} \subseteq Y$. Note that $E_j \subseteq F_{\phi^{-1}(j)}$; this follows from our choice of $\phi$, since $Z \subseteq F_{\phi^{-1}(j)}$ but we might have $Z \not\subseteq E_j$. Therefore

$$\begin{aligned}
&f(Y) - f(Z) \\
&\quad = \sum_{i=1}^{m} \big(f(F_i) - f(F_{i-1})\big) \\
&\quad = \sum_{j \in Y \setminus Z} \big(f(F_{\phi^{-1}(j)}) - f(F_{\phi^{-1}(j)-1})\big) \\
&\quad \leq \sum_{j \in Y \setminus Z} \big(f(E_j) - f(E_{j-1})\big) \qquad \text{(since } E_j \subseteq F_{\phi^{-1}(j)} \text{ and } f \text{ is submodular)} \\
&\quad = \sum_{j \in Y \setminus Z} \alpha_j \\
&\quad \leq t\sqrt{f(Y)} \qquad \text{(by (B.2)).}
\end{aligned}$$

So $f(Z) \geq f(Y) - t\sqrt{f(Y)} \geq b - t\sqrt{b}$, since $f(Y) \geq b$ and $t \leq \sqrt{b}$. This contradicts $Z \in \mathcal{A}$. ∎

This claim implies $\Pr[\, f(Y) \geq b \,] \leq \Pr[\, \rho(\mathcal{A}, Y) > t \,]$, so the theorem follows from (B.1). ∎

### B.2. Proof of Corollary 1.

COROLLARY 1. *Let $f : 2^{[n]} \to \mathbb{R}_+$ be a nonnegative, monotone, submodular, 1-Lipschitz function. Let the random variable $X \subseteq [n]$ have a product distribution. For any $0 \leq \alpha \leq 1$,*

$$\Pr[\, |f(X) - \mathbf{E}[\, f(X)\,]\,| > \alpha\, \mathbf{E}[\, f(X)\,]\,] \;\leq\; 4\exp\big(-\alpha^2\, \mathbf{E}[\, f(X)\,]/422\big).$$

*Proof.* Let $Y = f(X)$ and let $M$ be a median of $Y$. The idea of the proof is simple: Theorem 6 shows tight concentration of $Y$ around $M$. Since $Y$ is so tightly concentrated, we must have $\mathbf{E}[\, Y\,] \approx M$. This allows us to show tight concentration around $\mathbf{E}[\, Y\,]$. The remainder of the proof is simply a matter of detailed calculations. Similar arguments can be found in [49, section 2.5] and [74, section 20.2].

CLAIM 8.

$$\Pr[\, |Y - M| \geq \lambda\,] \;\leq\; \begin{cases} 4e^{-\lambda^2/8M} & (0 \leq \lambda \leq M), \\ 2e^{-\lambda/8} & (\lambda \geq M). \end{cases}$$

*Also,*

$$\Pr[\, |Y - M| \geq \lambda\,] \;\leq\; 4e^{-\lambda^2/24M} \qquad (0 \leq \lambda \leq 5M).$$

*Proof.* First, apply Theorem 6 with $b = M$ and $t = \lambda/\sqrt{M}$. Since $\Pr[\, Y \geq b\,] \leq 1/2$, we get

$$\Pr[\, Y \leq M - \lambda\,] \;\leq\; 2\exp(-t^2/4M). \tag{B.3}$$

Next, apply Theorem 6 with $b = M + \lambda$ and $t = \lambda/\sqrt{M+\lambda}$. Since

$$\Pr\Big[Y \le b - t\sqrt{b}\Big] \;=\; \Pr[Y \le M] \;\le\; \frac{1}{2},$$

we get

$$\Pr[Y \ge M+\lambda] \;\le\; 2\exp\big(-t^2/4(M+\lambda)\big). \tag{B.4}$$

Combining (B.3) and (B.4) proves the claim. ∎

Claim 9. $|\mathbf{E}[Y] - M| \le 15\sqrt{\mathbf{E}[Y]} + 16$. *Consequently, if* $\mathbf{E}[Y] \ge 256$, *then* $|\mathbf{E}[Y] - M| \le 16\sqrt{\mathbf{E}[Y]}$.

*Proof.* This is a standard calculation; see, e.g., [49, section 2.5]. Using Claim 8,

$$\begin{aligned}
|\mathbf{E}[Y] - M| &\le \mathbf{E}[\,|Y - M|\,] \\
&= \int_0^\infty \Pr[\,|Y-M| \ge \lambda\,]\,d\lambda \\
&= \int_0^M 4e^{-\lambda^2/8M}\,d\lambda \;+\; \int_M^\infty 2e^{-\lambda/8}\,d\lambda \\
&\le 4\sqrt{2\pi M} + 16e^{-M/8}.
\end{aligned}$$

Since $Y \ge 0$ we have $0 \le M \le 2\,\mathbf{E}[Y]$ (by Markov's inequality), so

$$|\mathbf{E}[Y] - M| \;\le\; 15\sqrt{\mathbf{E}[Y]} + 16.$$

This quantity is at most $16\sqrt{\mathbf{E}[Y]}$ if $\mathbf{E}[Y] \ge 256$. ∎

*Case* 1: $\mathbf{E}[Y] \ge 584/\alpha^2$. Then

$$\begin{aligned}
&\Pr\Big[\,|Y - \mathbf{E}[Y]| \ge (\sqrt{2}t + 16)\sqrt{\mathbf{E}[Y]}\,\Big] \\
&\quad\le \Pr\Big[\,|Y-M| \ge (\sqrt{2}t+16)\sqrt{\mathbf{E}[Y]} - |\mathbf{E}[Y]-M|\,\Big] \\
&\quad\le \Pr\Big[\,|Y-M| \ge t\sqrt{2\,\mathbf{E}[Y]}\,\Big] \qquad \text{(by Claim 9)} \\
&\quad\le \Pr\Big[\,|Y-M| \ge t\sqrt{M}\,\Big] \qquad \text{(since } \mathbf{E}[Y] \ge M/2\text{)} \\
&\quad\le 4\exp(-t^2/24) \qquad \text{(if } t \le 5\sqrt{M}\text{)} \qquad \text{(B.5)}
\end{aligned}$$

by Claim 8. Set $t = (\alpha\sqrt{\mathbf{E}[Y]} - 16)/\sqrt{2}$. One may check that

$$\frac{t^2}{24} \;=\; \frac{(\alpha\sqrt{\mathbf{E}[Y]} - 16)^2}{2\cdot 24} \;\ge\; \frac{\alpha^2\,\mathbf{E}[Y]}{422}, \tag{B.6}$$

since we assume $\mathbf{E}[Y] \ge 584/\alpha^2$. Furthermore,

$$\begin{aligned}
\sqrt{\mathbf{E}[Y]} &\le \sqrt{M} + \sqrt{|\mathbf{E}[Y] - M|} \\
&\le \sqrt{M} + 4\,\mathbf{E}[Y]^{1/4} \qquad \text{(by Claim 9)} \\
&\le \sqrt{M} + 0.82\sqrt{\mathbf{E}[Y]},
\end{aligned}$$

since $\mathbf{E}[Y]^{1/4} \ge 584^{1/4} > 4/0.82$. Rearranging, $0.18 \cdot \sqrt{\mathbf{E}[Y]} \le \sqrt{M}$. Therefore we have

$$t \;<\; \sqrt{\mathbf{E}[Y]}/\sqrt{2} \;<\; 4\cdot 0.18\cdot\sqrt{\mathbf{E}[Y]} \;\le\; 4\sqrt{M},$$

so we may apply (B.5) with this value of $t$.

$$\begin{aligned}\Pr\left[\,|Y-\mathbf{E}\left[\,Y\,\right]|\geq\alpha\,\mathbf{E}\left[\,Y\,\right]\,\right] &= \Pr\left[\,|Y-\mathbf{E}\left[\,Y\,\right]|\geq(\sqrt{2}t+16)\sqrt{\mathbf{E}\left[\,Y\,\right]}\,\right]\\ &\leq 4\exp(-t^2/24)\\ &\leq 4\exp(-\alpha^2\,\mathbf{E}\left[\,Y\,\right]/422),\end{aligned}$$

by (B.6).

*Case* 2: $\mathbf{E}\left[\,Y\,\right]<584/\alpha^2$. Then $\alpha^2\,\mathbf{E}\left[\,Y\,\right]/422<\ln(4)$, so

$$4\exp(-\alpha^2\,\mathbf{E}\left[\,Y\,\right]/422)\;>\;1,$$

and the claimed inequality is trivial. ∎

## Appendix C. Additional proofs for learning submodular functions.

### C.1. Learning Boolean submodular functions.

THEOREM 27. *The class of monotone, Boolean-valued, submodular functions is efficiently PMAC-learnable with approximation factor* 1.

*Proof.* Let $f:2^{[n]}\to\{0,1\}$ be an arbitrary monotone, Boolean, submodular function. We claim that $f$ is either constant or a monotone disjunction. If $f(\emptyset)=1$, then this is trivial, so assume $f(\emptyset)=0$.

Since submodularity is equivalent to the property of decreasing marginal values, and since $f(\emptyset)=0$, we get

$$f(T\cup\{x\})-f(T)\leq f(\{x\})\qquad\forall T\subseteq[n],x\in[n]\setminus T.$$

If $f(\{x\})=0$, then this together with monotonicity implies that $f(T\cup\{x\})=f(T)$ for all $T$. On the other hand, if $f(\{x\})=1$, then monotonicity implies that $f(T)=1$ for all $T$ such that $x\in T$. Thus we have argued that $f$ is a disjunction:

$$f(S)\;=\;\begin{cases}1 & (\text{if } S\cap X\neq\emptyset),\\ 0 & (\text{otherwise}),\end{cases}$$

where $X=\{\,x:f(\{x\})=1\,\}$. This proves the claim.

It is well known that the class of disjunctions is easy to learn in the supervised learning setting [58, 93]. ∎

Nonmonotone, Boolean, submodular functions need not be disjunctions. For example, consider the function $f$ where $f(S)=0$ if $S\in\{\emptyset,[n]\}$ and $f(S)=1$ otherwise; it is submodular, but not a disjunction. However, it turns out that any submodular Boolean function is a 2-DNF. This was already known [23], and it can be proven by case analysis as in Proposition 27. It is well known that 2-DNFs are efficiently PAC-learnable. We summarize this discussion as follows.

THEOREM 28. *The class of Boolean-valued, submodular functions is efficiently PMAC-learnable with approximation factor* 1.

### C.2. Learning under product distributions.

LEMMA 2. *Let $f:2^{[n]}\to\mathbb{R}$ be a nonnegative, monotone, submodular,* 1*-Lipschitz function. Suppose that $S_1,\ldots,S_l$ are drawn from a product distribution $D$ over $2^{[n]}$. Let $\mu$ the empirical average $\mu=\sum_{i=1}^{\ell}f^*(S_i)/\ell$, which is our estimate for $\mathbf{E}_{S\sim D}\left[\,f^*(S)\,\right]$. Let $\epsilon,\delta\leq 1/5$. We have the following:*

(1) *If* $\mathbf{E}[f^*(S)] > H\log(1/\epsilon)$ *and* $\ell \geq 16\log(1/\delta)$, *then*

$$\Pr[\mu \geq M\log(1/\epsilon)] \;\geq\; 1-\delta/4.$$

(2) *If* $\mathbf{E}[f^*(S)] > L\log(1/\epsilon)$ *and* $\ell \geq 16\log(1/\delta)$, *then*

$$\Pr\left[\tfrac{5}{6}\mathbf{E}[f^*(S)] \leq \mu \leq \tfrac{4}{3}\mathbf{E}[f^*(S)]\right] \;\geq\; 1-\delta/4.$$

(3) *If* $\mathbf{E}[f^*(S)] \leq H\log(1/\epsilon)$, *then*

$$\Pr[f^*(S) < K\log(1/\epsilon)] \;\geq\; 1-\epsilon.$$

(4) *If* $\mathbf{E}[f^*(S)] < L\log(1/\epsilon)$ *and* $\ell \geq 16\log(1/\delta)$, *then*

$$\Pr[\mu < M\log(1/\epsilon)] \;\geq\; 1-\delta/4.$$

*Proof.* (1) Let $\hat{f} : 2^{[n]\times[\ell]} \to \mathbb{R}$ be defined by

$$\hat{f}(S_1,\ldots,S_\ell) \;=\; \sum_{i=1}^{\ell} f^*(S_i).$$

It is easy to check that $\hat{f}$ is also nonnegative, monotone, submodular, and 1-Lipschitz. We will apply Corollary 1 to $\hat{f}$ with $\alpha = 1/10$. Then

$$\begin{aligned}
&\Pr[\mu < M\log(1/\epsilon)]\\
&\quad= \Pr\Big[\textstyle\sum_{i=1}^{\ell} f^*(S_i) < M\ell\log(1/\epsilon)\Big]\\
&\quad\leq \Pr\Big[\Big|\hat{f}(X) - \mathbf{E}\Big[f(\hat{X})\Big]\Big| > \mathbf{E}\Big[\hat{f}(X)\Big]/10\Big] \quad (\text{since } M \leq 0.9\cdot H)\\
&\quad\leq 4\exp\big(-\mathbf{E}\Big[\hat{f}(X)\Big]/42200\big)\\
&\quad\leq 4\exp\big(-\ell/4\big) \quad (\text{since } H > 10550)\\
&\quad\leq 4\delta^3 \;\leq\; \delta/4. \qquad \text{(C.1)}
\end{aligned}$$

(2) Let $\hat{f}$ and $X$ be as above. Then

$$\begin{aligned}
&\Pr\left[\tfrac{5}{6}\mathbf{E}[f^*(S)] \leq \mu \leq \tfrac{4}{3}\mathbf{E}[f^*(S)]\right]\\
&\quad\leq \Pr\left[\left|\mu - \mathbf{E}[f^*(S)]\right| > \mathbf{E}[f^*(S)]/10\right]\\
&\quad= \Pr\Big[\Big|\hat{f}(X) - \mathbf{E}\Big[\hat{f}(X)\Big]\Big| > \mathbf{E}\Big[\hat{f}(X)\Big]/10\Big]\\
&\quad\leq \delta/4,
\end{aligned}$$

as in (C.1), since $L \geq 10550$.

(3) Set $b = K\log(1/\epsilon)$ and $t = 4\sqrt{\log(1/\epsilon)}$. Since $K - 4\sqrt{K} \geq 2H$ we have $b - t\sqrt{b} \geq 2H\log(1/\epsilon) \geq 2\mathbf{E}[f^*(S)]$, and so $\Pr\big[f^*(S) \leq b - t\sqrt{b}\big] \geq 1/2$ by Markov's inequality. By Theorem 6, we have $\Pr[f^*(S) \geq b] \leq 2\exp(-t^2/4) \leq \epsilon$ since $\epsilon \leq 1/2$.

(4) Set $b = M\log(1/\epsilon)\ell$ and $t = 4\sqrt{\log(1/\delta)}$. Then

$$\begin{aligned}
b - t\sqrt{b} \;&=\; M\log(1/\epsilon)\ell - 4\sqrt{\log(1/\delta)}\sqrt{M\log(1/\epsilon)\ell}\\
&>\; (M - \sqrt{M})\log(1/\epsilon)\ell
\end{aligned}$$

since $4\sqrt{\log(1/\delta)\ell} \leq \ell$ and $\epsilon \leq 1/5$. Then, by Markov's inequality,

$$\begin{aligned}\Pr\left[\sum_{i=1}^{\ell} f^*(S_i) \leq b - t\sqrt{b}\right] &\geq 1 - \frac{\mathbf{E}\left[\sum_{i=1}^{\ell} f^*(S_i)\right]}{(M-\sqrt{M})\log(1/\epsilon)\ell} \\ &\geq 1 - \frac{L\log(1/\epsilon)\ell}{(M-\sqrt{M})\log(1/\epsilon)\ell} \\ &\geq 1/20\end{aligned}$$

since $L/(M-\sqrt{M}) \leq 0.95$. Applying Theorem 6 to the submodular function $\hat{f}(S_1, \ldots, S_\ell) = \sum_{i=1}^{\ell} f^*(S_i)$, we have $\Pr[\sum_{i=1}^{\ell} f^*(S_i) \geq b] \leq 20\exp(-t^2/4) \leq 20 \cdot \delta^4 \leq \delta/4$. ∎

### C.3. Learning lower bounds.

THEOREM 9. *Let $\mathcal{ALG}$ be an arbitrary learning algorithm that uses only a polynomial number of training examples, which can be either drawn i.i.d. from the underlying distribution or value queries. There exists a distribution $D$ and a submodular target function $f^*$ such that, with probability at least $1/4$ (over the draw of the training samples), the hypothesis function output by $\mathcal{ALG}$ does not approximate $f^*$ within an $o(n^{1/3}/\log n)$ factor on at least a $1/4$ fraction of the examples under $D$. This holds even for the subclass of matroid rank functions.*

*Proof.* First, consider a fully deterministic learning algorithm $\mathcal{ALG}$, i.e., an algorithm that doesn't even sample from $D$, though it knows $D$ and can use it in deterministically choosing queries. Say this algorithm makes $q < n^c$ queries (which could be chosen adaptively). Each query has at most $n$ possible answers, since the minimum rank of any set is zero and the maximum rank is at most $n$. So the total number of possible sequences of answers is at most $n^q$.

Now, since the algorithm is deterministic, the hypothesis it outputs at the end is uniquely determined by this sequence of answers. To be specific, its choice of the second query is uniquely determined by the answer given to the first query, its choice of the third query is uniquely determined by the answers given to the first two queries, and by induction, its choice of the $i$th query $q_i$ is uniquely determined by the answers given to all queries $q_1, \ldots, q_{i-1}$ so far. Its final hypothesis is uniquely determined by all $q$ answers. This then implies that $\mathcal{ALG}$ can output at most $n^q$ different hypotheses.

We will apply Theorem 1 with $k = 2^t$, where $t = c\log(n) + \log(\ln n) + 14$ (so $k = n^c \cdot \ln(n) \cdot 2^{14} > 10000 \cdot q \cdot \ln(n)$). Let $\mathcal{A}$ and $\mathcal{M}$ be the families constructed by Theorem 1. Let the underlying distribution $D$ on $2^{[n]}$ be the uniform distribution on $\mathcal{A}$. (Note that $D$ is not a product distribution.) Choose a matroid $\mathbf{M}_{\mathcal{B}} \in \mathcal{M}$ uniformly at random and let the target function be $f^* = \mathrm{rank}_{\mathbf{M}_{\mathcal{B}}}$. Let us fix a hypotheses $h$ that $\mathcal{ALG}$ might output. By Hoeffding bounds, we have

$$\Pr_{f^*,\mathcal{S}}\left[\Pr_{A\sim D}\left[f^*(A) \notin \left[h(A), \tfrac{n^{1/3}}{16t}h(A)\right] \leq 0.49\right]\right] \leq e^{-2(.01)^2 k} = e^{-2q\cdot\ln(n)} = n^{-2q},$$

i.e., with probability at least $1 - n^{-2q}$, $h$ has high approximation error on over 49% of the examples.

By a union bound over all over all the $n^q$ hypotheses $h$ that $\mathcal{ALG}$ might output, we obtain that with probability at least $1/4$ (over the draw of the training samples) the hypothesis function output by $\mathcal{ALG}$ does not approximate $f^*$ within a $o(n^{1/3}/\log n)$ factor on at least $1/4$ fraction of the examples under $D$.

The above argument is a fixed randomized strategy for the adversary that works against any deterministic $\mathcal{ALG}$ making at most $n^c$ queries. By Yao's minimax principle, this means that, for any randomized algorithm making at most $n^c$ queries, there exists $\mathbf{M}_{\mathcal{B}}$ which the algorithm does not learn well, even with arbitrary value queries. ∎

COROLLARY 2. *Suppose one-way functions exist. For any constant $\epsilon > 0$, no algorithm can PMAC-learn the class of nonnegative, monotone, submodular functions with approximation factor $O(n^{1/3-\epsilon})$, even if the functions are given by polynomial-time algorithms computing their value on the support of the distribution.*

*Proof of* Corollary 2. The argument follows Kearns and Valiant [57]. We will apply Theorem 1 with $k = 2^t$, where $t = n^\epsilon$. There exists a family of pseudorandom Boolean functions $H_t = \{h_y : y \in \{0,1\}^t\}$, where each function is of the form $h_y : \{0,1\}^t \to \{0,1\}$. Choose an arbitrary bijection between $\{0,1\}^t$ and $\mathcal{A}$. Then each $h_y \in H_t$ corresponds to some subfamily $\mathcal{B} \subseteq \mathcal{A}$ and hence to a matroid rank function $\text{rank}_{\mathbf{M}_{\mathcal{B}}}$. Suppose there is a PMAC-learning algorithm for this family of functions which achieves an approximation ratio better than $n^{1/3}/16t$ on a set of measure $1/2 + 1/\operatorname{poly}(n)$. Then this algorithm must be predicting the function $h_y$ on a set of size $1/2 + 1/\operatorname{poly}(n) = 1/2 + 1/\operatorname{poly}(t)$. This is impossible, since the family $H_t$ is pseudorandom. ∎

## Appendix D. Expander construction.

THEOREM 5. *Let $G = (U \cup V, E)$ be a random multigraph where $|U| = k$, $|V| = n$, and every $u \in U$ has exactly $d$ incident edges, each of which has an endpoint chosen uniformly and independently from all nodes in $V$. Suppose that $k \geq 4$, $d \geq \log(k)/\epsilon$, and $n \geq 16Ld/\epsilon$. Then, with probability at least $1 - 2/k$,*

$$|\Gamma(J)| \;\geq\; (1-\epsilon)\cdot d \cdot |J| \qquad \forall J \subseteq U,\, |J| \leq L. \tag{D.1}$$

*If it is desired that $|\Gamma(\{u\})| = d$ for all $u \in U$, then this can be achieved by replacing any parallel edges incident on $u$ by new edges with distinct endpoints. This cannot decrease $|\Gamma(J)|$ for any $J$, and so* (D.1) *remains satisfied.*

The proof is an variant of the argument in Vadhan's survey [91, Theorem 4.4].

*Proof.* Fix $j \leq L$ and consider any set $J \subseteq U$ of size $|J| = j$. The sampling process decides the neighbors $\Gamma(J)$ by picking a sequence of $jd$ neighbors $v_1, \ldots, v_{jd} \in V$. An element $v_i$ of that sequence is called a *repeat* if $v_i \in \{v_1, \ldots, v_{i-1}\}$. Conditioned on $v_1, \ldots, v_{i-1}$, the probability that $v_i$ is a repeat is at most $jd/n$. The set $J$ violates (D.1) only if there exist more than $\epsilon jd$ repeats. The probability of this is at most

$$\binom{jd}{\epsilon jd}\left(\frac{jd}{n}\right)^{\epsilon jd} \;\leq\; \left(\frac{e}{\epsilon}\right)^{\epsilon jd}\left(\frac{jd}{n}\right)^{\epsilon jd} \;\leq\; (1/4)^{\epsilon jd}.$$

The last inequality follows from $j \leq L$ and our hypothesis $n \geq 16Ld/\epsilon$. So the probability that there exists a $J \subseteq U$ with $j = |J|$ that violates (D.1) is at most

$$\binom{k}{j}(1/4)^{-\epsilon jd} \;\leq\; k^j 2^{-2\epsilon jd} \;=\; 2^{-j(2\epsilon d - \log k)} \;\leq\; k^{-j},$$

since $d \geq \log(k)/\epsilon$. Therefore the probability that any $J$ with $|J| \leq L$ violates (D.1) is at most

$$\sum_{j \geq 1} k^{-j} \;\leq\; 2/k.$$

∎

**Appendix E. Special cases of the matroid construction.** The matroid constructions of Theorems 2 and 3 have several interesting special cases.

**E.1. Partition matroids.** Suppose we are given disjoint sets $A_1, \ldots, A_k$ and values $b_1, \ldots, b_k$. We claim that the matroid $\mathcal{I}$ defined in Theorem 2 is a partition matroid. To see this, note that $g(J) = \sum_{j \in J} b_j$, since the $A_j$'s are disjoint, so $g$ is a modular function. Similarly, $|I \cap A(J)|$ is a modular function of $J$. Thus, whenever $|J| > 1$, the constraint $|I \cap A(J)| \leq g(J)$ is redundant—it is implied by the constraints $|I \cap A_j| \leq b_j$ for $j \in J$. So we have

$$\mathcal{I} \;=\; \{\, I : |I \cap A(J)| \leq g(J) \;\; \forall J \subseteq [k] \,\} \;\;=\;\; \{\, I : |I \cap A_j| \leq b_j \;\; \forall j \in [k] \,\},$$

which is the desired partition matroid.

**E.2. Pairwise intersections.** Suppose we are given sets $A_1, \ldots, A_k$ and values $b_1, \ldots, b_k$. We now describe the special case of the matroid construction which only considers the pairwise intersections of the $A_i$'s.

LEMMA 8. *Let $d$ be a nonnegative integer such that $d \leq \min_{i,j \in [k]}(b_i + b_j - |A_i \cap A_j|)$. Then*

$$\mathcal{I} \;=\; \{\, I : |I| \leq d \;\wedge\; |I \cap A_j| \leq b_j \;\forall j \in [k] \,\}$$

*is the family of independent sets of a matroid.*

*Proof.* Note that for any pair $J = \{i, j\}$, we have $g(J) = b_i + b_j - |A_i \cap A_j|$. Then

$$d \;\leq\; \min_{i,j \in [k]} (b_i + b_j - |A_i \cap A_j|) \;\;=\;\; \min_{J \subseteq [k],\, |J| = 2} g(J),$$

so $g$ is $(d, 2)$-large. The lemma follows from Theorem 3. ∎

**E.3. Paving matroids.** A *paving matroid* is defined to be a matroid $\mathbf{M} = (V, \mathcal{I})$ of rank $m$ such that every circuit has cardinality either $m$ or $m + 1$. We will show that every paving matroid can be derived from our matroid construction (Theorem 3). First of all, we require a structural lemma about paving matroids.

LEMMA 9. *Let $\mathbf{M} = (V, \mathcal{I})$ be a paving matroid of rank $m$. There exists a family $\mathcal{A} = \{A_1, \ldots, A_k\} \subset 2^V$ such that*

$$\mathcal{I} \;=\; \{\, I : |I| \leq m \;\wedge\; |I \cap A_i| \leq m - 1 \;\; \forall i \,\}, \tag{E.1a}$$

$$|A_i \cap A_j| \;\leq\; m - 2 \quad \forall i \neq j. \tag{E.1b}$$

Related results can be found in Theorem 5.3.5, Problem 5.3.7, and Exercise 5.3.8 of Frank's book [30].

*Proof.* It is easy to see that there exists $\mathcal{A}$ satisfying (E.1a), since we may simply take $\mathcal{A}$ to be the family of circuits which have size $m$. So let us choose a family $\mathcal{A}$ that satisfies (E.1a) and minimizes $|\mathcal{A}|$. We will show that this family must satisfy (E.1b). Suppose otherwise, i.e., there exist $i \neq j$ such that $|A_i \cap A_j| \geq m - 1$.

*Case* 1: $r(A_i \cup A_j) \leq m - 1$. Then $\mathcal{A} \setminus \{A_i, A_j\} \cup \{A_i \cup A_j\}$ also satisfies (E.1a), contradicting minimality of $|\mathcal{A}|$.

*Case* 2: $r(A_i \cup A_j) = m$. Observe that $r(A_i \cap A_j) \geq m - 1$ since $|A_i \cap A_j| \geq m - 1$ and every set of size $m - 1$ is independent. So we have

$$r(A_i \cup A_j) + r(A_i \cap A_j) \;\geq\; m + (m - 1) \;>\; (m - 1) + (m - 1) \;\geq\; r(A_i) + r(A_j).$$

This contradicts submodularity of the rank function. ∎

For any paving matroid, Lemma 9 implies that its independent sets can be written in the form

$$\mathcal{I} \;=\; \{\, I \,:\, |I| \le m \;\;\wedge\;\; |I \cap A_i| \le m-1 \;\; \forall i \,\},$$

where $|A_i \cap A_j| \le m-2$ for each $i \neq j$. This is a special case of Theorem (3) since we may apply Lemma 8 with each $b_i = m-1$ and $d = m$, since

$$\min_{i,j \in [k]} (b_i + b_j - |A_i \cap A_j|) \;\ge\; 2(m-1) - (m-2) \;=\; m.$$

**Acknowledgments.** We thank Jan Vondrák for simplifying our original proof of Theorem 2, Atri Rudra for explaining how our original proof of Theorem 1 was connected to expander graphs, and Florin Constantin for discussions about gross substitutes. We also thank Avrim Blum, Shahar Dobzinski, Steve Hanneke, Satoru Iwata, Lap Chi Lau, Noam Nisan, Alex Samorodnitsky, Mohit Singh, Santosh Vempala, and Van Vu for helpful discussions.

## REFERENCES

[1] N. Alon and J. Spencer, *The Probabilistic Method*, Wiley, New York, 2000.
[2] M. Anthony and P. Bartlett, *Neural Network Learning: Theoretical Foundations*, Cambridge University Press, Cambridge, UK, 1999.
[3] A. Badanidiyuru, S. Dobzinski, H. Fu, R. Kleinberg, N. Nisan, and T. Roughgarden, *Sketching valuation functions*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2012.
[4] M. F. Balcan, A. Blum, and Y. Mansour, *Item pricing for revenue maxmimization*, in Proceedings of the ACM Conference on Electronic Commerce, 2009.
[5] M.-F. Balcan, F. Constantin, S. Iwata, and L. Wang, *Learning valuation functions*, in Proceedings of the 25th Conference on Learning Theory, 2012.
[6] M. F. Balcan, E. Vitercik, and C. White, *Learning combinatorial functions from pairwise comparisons*, in Proceedings of the 29th Conference on Learning Theory, 2016.
[7] E. Baum and K. Lang, *Query learning can work poorly when a human oracle is used*, in Proceedings of the IEEE International Joint Conference on Neural Networks, 1993.
[8] M. Bing, D. Lehmann, and P. Milgrom, *Presentation and structure of substitutes valuations*, in Proceedings of the ACM Conference on Electronic Commerce, 2004.
[9] L. Blumrosen, *Information and Communication in Mechanism Design*, Ph.D. thesis, Hebrew University, Jerusalem, 2006.
[10] L. Blumrosen and N. Nisan, *On the computational power of iterative auctions*, in Proceedings of the ACM Conference on Electronic Commerce, 2005, pp. 29–43.
[11] S. Boucheron, G. Lugosi, and P. Massart, *On concentration of self-bounding functions*, Electron. J. Probab., 14 (2009), pp. 1884–1899.
[12] H. Buhrman, P. B. Miltersen, J. Radhakrishnan, and S. Venkatesh, *Are bitvectors optimal?*, SIAM J. Comput., 31 (2002), pp. 1723–1744.
[13] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, *Maximizing a submodular set function subject to a matroid constraint*, SIAM J. Comput., 40 (2011), pp. 1740–1766.
[14] C. Chekuri and J. Vondrák, *Randomized Pipage Rounding for Matroid Polytopes and Applications*, arXiv:0909.4348v1, 2009.
[15] C. Chekuri, J. Vondrák, and R. Zenklusen, *Dependent randomized rounding via exchange properties of combinatorial structures*, in Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, 2010, pp. 575–584.
[16] C. Chekuri, J. Vondrak, and R. Zenklusen, *Submodular function maximization via the multilinear relaxation and contention resolution schemes*, in Proceedings of the ACM Symposium on Theory of Computing, 2011.
[17] M. Cheraghchi, A. R. Klivans, P. Kothari, and H. K. Lee, *Submodular functions are noise stable*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2012.
[18] P. Cramton, Y. Shoham, and R. Steinberg, eds., *Combinatorial Auctions*, MIT Press, Cambridge, MA, 2006.
[19] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer, New York, 1996.

[20] S. Dobzinski, N. Nisan, and M. Schapira, *Truthful randomized mechanisms for combinatorial auctions*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing, 2006, pp. 644–652.

[21] S. Dobzinski and J. Vondrák, *Communication complexity of combinatorial auctions with submodular valuations*, in Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms, 2013, pp. 1205–1215.

[22] J. Edmonds, *Submodular Functions, Matroids, and Certain Polyhedra*, in Combinatorial Structures and Their Applications, R. Guy, H. Hanani, N. Sauer, and J. Schönheim, eds., Gordon and Breach, New York, 1970, pp. 69–87.

[23] O. Ekin, P. L. Hammer, and U. N. Peled, *Horn functions and submodular boolean functions*, Theoret. Comput. Sci., 175 (1997), pp. 257–270.

[24] U. Feige, V. Mirrokni, and J. Vondrák, *Maximizing non-monotone submodular functions*, in Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, 2007.

[25] M. Feldman, J. Naor, and R. Schwartz, *Nonmonotone submodular maximization via a structural continuous greedy algorithm*, in Proceedings of the 38th International Colloquium on Automata, Languages and Programming, 2011.

[26] M. Feldman, J. Naor, and R. Schwartz, *A unified continuous greedy algorithm for submodular maximization*, in Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, 2011.

[27] V. Feldman and J. Vondrák, *Tight bounds on low-degree spectral concentration of submodular and XOS functions*, in Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science, 2015.

[28] V. Feldman and J. Vondrák, *Optimal bounds on approximation of submodular and XOS functions by juntas*, SIAM J. Comput., 45 (2016), pp. 1129–1170.

[29] Y. Filmus and J. Ward, *A tight combinatorial algorithm for submodular maximization subject to a matroid constraint*, in Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science, 2012, pp. 659–668.

[30] A. Frank, *Connections in Combinatorial Optimization*, Oxford University Press, New York, 2011.

[31] S. Fujishige, *Submodular Functions and Optimization*, Elsevier, Amsterdam, 2005.

[32] G. Goel, C. Karande, P. Tripathi, and L. Wang, *Approximability of combinatorial problems with multi-agent submodular cost functions*, in Proceedings of the 50th Annual Symposium on Foundations of Computer Science, 2009.

[33] M. Goemans, N. Harvey, S. Iwata, and V. Mirrokni, *Approximating submodular functions everywhere*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2009.

[34] M. Goemans, R. Kleinberg, N. Harvey, and V. Mirrokni, *On Learning Submodular Functions*, manuscript, 2008.

[35] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer, New York, 1993.

[36] F. Gul and E. Stacchetti, *Walrasian equilibrium with gross substitutes*, J. Econom. Theory, 87 (1999), pp. 95–124.

[37] A. Gupta, M. Hardt, A. Roth, and J. Ullman, *Privately releasing conjunctions and the statistical query barrier*, in Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, 2011.

[38] V. Guruswami and P. Raghavendra, *Hardness of learning halfspaces with noise*, in Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, 2006.

[39] V. Guruswami, C. Umans, and S. P. Vadhan, *Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes*, J. ACM, 56 (2009).

[40] M. T. Hajiaghayi, J. H. Kim, T. Leighton, and H. Räcke, *Oblivious routing in directed graphs with random demands*, in Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 193–201.

[41] W. Hanson and R. K. Martin, *Optimal bundle pricing*, Management Sci., 36 (1990), pp. 155–174.

[42] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York, 2009.

[43] D. Haussler, *Decision theoretic generalizations of the PAC model for neural net and other learning applications*, Inform. and Comput., 100 (1992), pp. 78–150.

[44] S. Hoory, N. Linial, and A. Wigderson, *Expander graphs and their applications*, Bull. AMS, 43 (2006), pp. 439–561.

[45] S. Iwata, L. Fleischer, and S. Fujishige, *A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions*, J. ACM, 48 (2001), pp. 761–777.

[46] S. Iwata and K. Nagano, *Submodular function minimization under covering constraints*, in Proceedings of the 50th Annual Symposium on Foundations of Computer Science, 2009.
[47] S. Iwata and J. Orlin, *A simple combinatorial algorithm for submodular function minimization*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2009.
[48] R. K. Iyer, S. Jegelka, and J. A. Bilmes, *Curvature and optimal algorithms for learning and minimizing submodular functions*, in 27th Annual Conference on Neural Information Processing Systems, 2013.
[49] S. Janson, T. Łuczak, and A. Ruciński, *Random Graphs*, Wiley-Interscience, New York, 2000.
[50] S. Jegelka and J. Bilmes, *Notes on graph cuts with submodular edge weights*, in Proceedings of the Workshop on Discrete Optimization in Machine Learning: Submodularity, Sparsity & Polyhedra, 2009.
[51] P. M. Jensen and B. Korte, *Complexity of matroid property algorithms*, SIAM J. Comput, 11 (1982), pp. 184–190.
[52] F. John, *Extremum Problems with inequalities as subsidiary conditions*, in Studies and Essays, Interscience, New York, 1948, pp. 187–204.
[53] A. Kalai, A. Klivans, Y. Mansour, and R. Servedio, *Agnostically learning halfspaces*, SIAM J. Comput., 37 (2008), pp. 1777–1805.
[54] A. T. Kalai, A. Samorodnitsky, and S. Teng, *Learning and smoothed analysis*, in Proceedings of the 50th Annual Symposium on Foundations of Computer Science, 2009.
[55] G. Kalai, *Learnability and Rationality of Choice*, Technical report, Institute of Mathematics and Center for Rationality, Hebrew University of Jerusalem, Jerusalem, Israel, 2001.
[56] G. Kalai, *Learnability and rationality of choice*, J. Econom. Theory, 113 (2003), pp. 104–117.
[57] M. Kearns and L. Valiant, *Cryptographic limitations on learning boolean formulae and finite automata*, J. ACM, 41 (1994), pp. 67–95.
[58] M. Kearns and U. Vazirani, *An Introduction to Computational Learning Theory*, MIT Press, Cambridge, MA, 1994.
[59] A. Kelso and V. Crawford, *Job matching, coalition formation, and gross substitutes*, Econometrica, 50 (1982), pp. 1483–1504.
[60] A. Klivans, R. O'Donnell, and R. Servedio, *Learning intersections and thresholds of halfspaces*, J. Comput. System Sci., 68 (2004), pp. 808–840.
[61] A. Krause and C. Guestrin, *Near-optimal nonmyopic value of information in graphical models*, in Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, 2005.
[62] A. Krause and C. Guestrin, *Beyond Convexity: Submodularity in Machine Learning*, 2008, http://www.select.cs.cmu.edu/tutorials/icml08submodularity.html.
[63] A. Krause and C. Guestrin, *Intelligent Information Gathering and Submodular Function Optimization*, http://www.ijcai.org/past/ijcai-09/tutorials/tutorial-MP4.html (2009).
[64] A. Kulik, H. Shachnai, and T. Tamir, *Maximizing submodular set functions subject to multiple linear constraints*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2009, pp. 545–554.
[65] J. Lee, V. Mirrokni, V. Nagarajan, and M. Sviridenko, *Maximizing nonmonotone submodular functions under matroid and knapsack constraints*, SIAM J. Discrete Math., 23 (2010), pp. 2053–2078.
[66] J. Lee, M. Sviridenko, and J. Vondrak, *Submodular maximization over multiple matroids via generalized exchange properties*, Math. Oper. Res., 35 (2010), pp. 795–806.
[67] B. Lehmann, D. J. Lehmann, and N. Nisan, *Combinatorial auctions with decreasing marginal utilities*, Games Econom. Behav., 55 (2006), pp. 270–296.
[68] Y. Li, P. M. Long, and A. Srinivasan, *Improved bounds on the sample complexity of learning*, J. Comput. System Sci., (2001), pp. 516–527.
[69] N. Linial, Y. Mansour, and N. Nisan, *Constant depth circuits, Fourier transform and learnability*, J. ACM, 40 (1993), pp. 607–620.
[70] L. Lovász, *Submodular functions and convexity*, Mathematical Programmming: The State of the Art, Springer, New York, 1983, pp. 235–257.
[71] S. T. McCormick, *Submodular Function Minimization*, in Handbook on Discrete Optimization, K. Aardal, G. Nemhauser, and R. Weismantel, eds., Elsevier, Amsterdam, 2006, pp. 321–391.
[72] P. R. Milgrom, *Putting auction theory to work: The simultaneous ascending auction*, J. Political Economy, 108 (2000), pp. 245–272.
[73] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*, Cambridge University Press, Cambridge, UK, 2005.
[74] M. Molloy and B. Reed, *Graph Colouring and the Probabilistic Method*, Springer, New York, 2001.

[75] K. Murota, *Discrete Convex Analysis*, Discrete Math. Appl., SIAM, Philadelphia, 2003.
[76] M. Narasimhan and J. Bilmes, *Local search for balanced submodular clusterings*, in Proceedings of the 20th International Joint Conference on Artificial Intelligence, 2007.
[77] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, *An analysis of approximations for maximizing submodular set functions*—I, Math. Programm., 14 (1978), pp. 265–294.
[78] Proceedings of the *NIPS Workshop on Discrete Optimization in Machine Learning (DISCML): Submodularity, Sparsity & Polyhedra*, http://www.discml.cc (2009).
[79] Proceedings of the *NIPS Workshop on Discrete Optimization in Machine Learning (DISCML): Uncertainty, Generalization and Feedback*, http://las.ethz.ch/discml (2011).
[80] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, eds., *Algorithmic Game Theory*, Cambridge University Press, Cambridge, UK, 2007.
[81] S. O. Gharan and J. Vondrák, *Submodular maximization by simulated annealing*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2011, pp. 1098–1117.
[82] J. G. Oxley, *Matroid Theory*, Oxford University Press, New York, 1992.
[83] J.-C. Picard and M. Queyranne, *On the structure of all minimum cuts in a network and applications*, Math. Program. Study, 13 (1980), pp. 8–16.
[84] S. Raskhodnikova and G. Yaroslavtsev, *Learning Pseudo-Boolean k-DNF and Submodular Functions*, arXiv:1208.2294, 2012.
[85] A. Schrijver, *A combinatorial algorithm minimizing submodular functions in strongly polynomial time*, J. Combin. Theory Ser. B, 80 (2000), pp. 346–355.
[86] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Springer, New York, 2004.
[87] R. Servedio, *On learning monotone DNF under product distributions*, Inform. and Comput., 193 (2004), pp. 1710–1722.
[88] M. Sipser and D. A. Spielman, *Expander codes*, IEEE Trans. Inform. Theory, 42 (1996), pp. 1710–1722.
[89] Z. Svitkina and L. Fleischer, *Submodular approximation: Sampling-based algorithms and lower bounds*, in Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science, 2008.
[90] M. Talagrand, *Concentration of measure and isoperimetric inequalites in product spaces*, Publ. Math. Inst. Hautes Etudes Sci., 81 (1995), pp. 73–205.
[91] S. P. Vadhan, *Pseudorandomness*, Found. Trends Theor. Comput. Sci., 7, (2012), pp. 1–336.
[92] L. G. Valiant, *A theory of the learnable*, Commun. ACM, 27 (1984), pp. 1134–1142.
[93] V. N. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998.
[94] S. A. Vinterbo, *A stab at approximating minimum subadditive join*, in Proceedings of the 10th International Workshop on Algorithms and Data Structures, 2007.
[95] J. Vondrák, *Optimal approximation for the submodular welfare problem in the value oracle model*, in Proceedings of the 40th Annual ACM Symposium on Theory of Computing, 2008.
[96] J. Vondrák, *Symmetry and approximability of submodular maximization problems*, in Proceedings of the IEEE Symposium on Foundations of Computer Science, 2009, pp. 651–670.
[97] J. Vondrák, *A Note on Concentration of Submodular Functions*, arXiv:1005.2791, 2010.
[98] L. A. Wolsey, *An analysis of the greedy algorithm for the submodular set covering problem*, Combinatorica, 2 (1982), pp. 385–393.