LIMITATION OF LEARNING RANKINGS FROM PARTIAL INFORMATION

By Srikanth Jagabathula Devavrat Shah *

Interest is in recovering distribution over the space of permutations over n elements given partial information. Non-trivial sufficient conditions for this question were introduced by Jagabathula and Shah (2008). This note states an unresolved question (see Conjecture 3.1) pertaining necessary conditions.

1. Problem Statement. In this section, we introduce the necessary notations, definitions and provide the formal problem statement. Here we closely follow setup introduced by Jagabathula and Shah [JS08, JS09].

1.1. *Notations.* Let n be the number of elements and S_n be set of all possible $n!$ permutations or rankings of these of n elements. Our interest is in learning non-negative valued functions f defined on S_n, i.e. $f : S_n \to \mathbb{R}_+$, where $\mathbb{R}_+ = \{x \in \mathbb{R} : x \geq 0\}$. The support of f is defined as

$$\operatorname{supp}(f) = \{\sigma \in S_n \colon f(\sigma) \neq 0\}.$$

The cardinality of support, $|\operatorname{supp}(f)|$ is called the *sparsity* of f and denoted by K. We also call it the ℓ_0 norm of f, denoted by $|f|_0$.

In this paper, our goal is to learn f from partial information. In order to formally define the partial information we consider, we need some notations. To this end, consider a partition of n, i.e. an ordered tuple $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_r)$, such that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r \geq 1$, and $n = \lambda_1 + \lambda_2 + \ldots + \lambda_r$. For example, $\lambda = (n-1, 1)$ is a partition of n. Now consider a partition of the n elements, $\{1, \ldots, n\}$, as per the λ partition, i.e. divide n elements into r bins with ith bin having λ_i elements. It is easy to see that n elements can be divided as per the λ partition in D_λ distinct ways, with

$$D_\lambda = \frac{n!}{\prod_{i=1}^{r} \lambda_i!}.$$

Let the distinct partitions be denoted by $t_i, 1 \leq i \leq D_\lambda$[1]. For example, for $\lambda = (n-1, 1)$ there are $D_\lambda = n!/(n-1)! = n$ distinct ways given by

$$t_i \equiv \{1, \ldots, i-1, i+1, \ldots, n\}\{i\}, \ 1 \leq i \leq n.$$

*SJ and DS are with the Laboratory for Information and Decision Systems (LIDS), department of EECS at Massachusetts Institute of Technology.

[1]To keep notation simple, we use t_i instead of t_i^λ that takes explicit dependence on λ into account.

Given a permutation $\sigma \in S_n$, its action on t_i is defined through its action on the n elements of t_i, resulting in a λ partition with the n elements permuted. In the above example with $\lambda = (n-1, 1)$, σ acts on t_i to give $t_{\sigma(i)}$, i.e.

$$\sigma : t_i \equiv \{1, \ldots, i-1, i+1, \ldots, n\}\{i\} \to t_{\sigma(i)} \equiv \{1, \ldots, \sigma(i)-1, \sigma(i)+1, \ldots, n\}\{\sigma(i)\}.$$

Now, for a given partition λ and a permutation $\sigma \in S_n$, define a 0/1 valued $D_\lambda \times D_\lambda$ matrix $M^\lambda(\sigma)$ as

$$M^\lambda_{ij}(\sigma) = \begin{cases} 1, & \text{if } \sigma(t_j) = t_i \\ 0, & \text{otherwise.} \end{cases} \qquad \text{for all } 1 \leq i, j \leq D_\lambda$$

This matrix $M^\lambda(\sigma)$ corresponds to a degree D_λ representation of the permutation group.

1.2. *Partial Information.* The partial information is the Fourier transform coefficient of f at the representation M^λ, for each λ. The motivation for considering Fourier coefficients at representations M^λ is two folds: first, Fourier coefficients at representations M^λ have natural interpretations, which makes it easy to collect in practical applications; second, each representation M^λ contains appropriate "lower-order" irreducible representations; thus, for each λ, M^λ conveniently captures the information contained in all the "lower-order" Fourier coefficients "up to" λ. We now define the Fourier coefficient of f at the representation M^λ, which we call λ-partial information.

DEFINITION 1.1 (λ-Partial Information). *Given a function $f : S_n \to \mathbb{R}_+$ and partition λ. The Fourier Transform coefficient at representation M^λ, which we call the λ-partial information, is denoted by $\hat{f}(\lambda)$ and is defined as*

$$\hat{f}(\lambda) = \sum_{\sigma \in S_n} f(\sigma) M^\lambda(\sigma).$$

Recall the example of $\lambda = (n-1, 1)$ with f as a probability distribution on S_n. Then, $\hat{f}(\lambda)$ is an $n \times n$ matrix with the (i, j)th entry being the probability of element j mapped to element i under f. That is, $\hat{f}(\lambda)$ corresponds to the *first order* marginal of f in this case.

Finally, we describe a graphical representation of $\hat{f}(\lambda)$ that will be useful throughout the paper. To this end, for given λ and $\sigma \in S_n$, one may imagine $M^\lambda(\sigma)$ as a perfect matching in a $D_\lambda \times D_\lambda$ complete bipartite graph with $M^\lambda(\sigma)$ as the adjacency matrix of this matching. Note that an edge in this bipartite graph corresponds to mapping a λ tabloid t_i to t_j. We

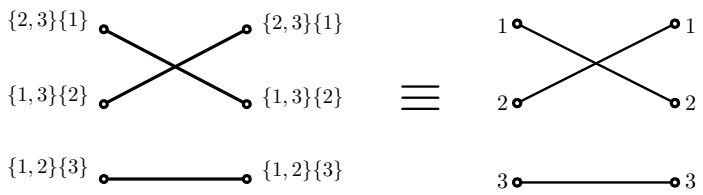

FIG 1. *Representation of the permutation* $(1,2)(3)$ *as the* λ*-bipartite matching for* $n = 3$ *and* $\lambda = (n-1,1)$*. Each edge of the matching represents the mapping of the* λ*-tabloids under the permutation. For* $\lambda = (n-1,1)$*, the representation on the right is an equivalent representation where the nodes of the graph are labeled using only the second partition – 1 to* n*.*

call this the λ-bipartite matching of the permutation. Figure 1 shows the λ-bipartite graph corresponding to the permutation $(1,2)(3)$ for $n = 3$, $\lambda = (n-1,1)$, where we have represented the permutation using the standard cycle notation. In Figure 1 the bipartite matching on the right is an equivalent representation where the nodes of the graph are labeled with only the second partition – from 1 to n. Using this bipartite matching representation of each permutation, the matrix $\hat{f}(\lambda)$ can be thought of as a weighted bipartite graph: the bipartite graph is simply a superposition of the λ-bipartite matchings corresponding to the permutations in the support, and the weight of an edge (t_i, t_j) is given by (i,j)th entry in matrix $\hat{f}(\lambda)$. We call this the λ-weighted bipartite graph of f. Figure 2 illustrates an example where $n = 3$, $K = 3$, $\sigma_1 = (1,2)(3)$, $\sigma_2 = (1)(2)(3)$ and $\sigma_3 = (1)(2,3)$ and $f(\sigma_1) = 0.1, f(\sigma_2) = 0.2$ and $f(\sigma_3) = 0.7$. Further, $f(\sigma) = 0$ for $\sigma \neq \sigma_1, \sigma_2, \sigma_3$. The figure illustrates the λ-weighted bipartite graph for $\lambda = (n-1,1)$.

1.3. *Problem Formulation.* The goal is to recover a function f from its partial information $\hat{f}(\lambda)$, based on partition λ. As noted earlier, one approach is to recover the function as the solution to the following ℓ_0 optimization problem:

$$\begin{array}{lll} \text{minimize} & \|g\|_0 \quad \text{over} & g : S_n \to \mathbb{R}_+ \\ \text{subject to} & \hat{g}(\lambda) = \hat{f}(\lambda). & \end{array} \tag{1.1}$$

We note that the question of recovering f from $\hat{f}(\lambda)$ is very similar to the question studied in the context of compressed sensing, i.e. recover x from $y = Ax$. To see this, with some abuse of notation imagine $\hat{f}(\lambda)$ as the D_λ^2 dimensional vector and f as $n!$ dimensional vector. Then, $\hat{f}(\lambda) = Af$ where each column of A corresponds to $M^\lambda(\sigma)$ for certain permutation σ. The key

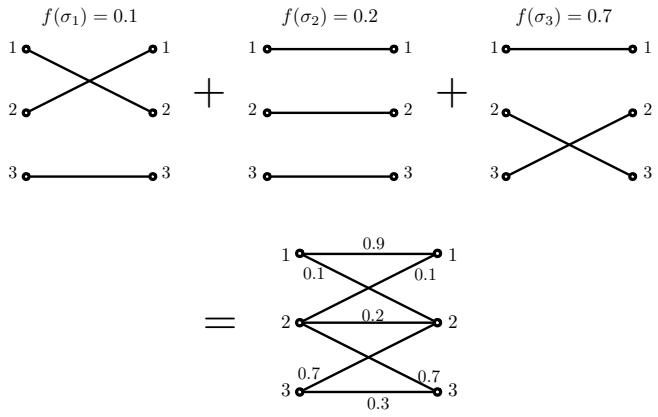

FIG 2. *The λ-weighted bipartite graph of function f for $\lambda = (n-1, 1)$. This is a convenient way to imagine the first-order partial information.*

difference from the compressed sensing literature is that A is given in our setup rather than being a design choice.

Question One. As the first question of interest, we wish to identify precise conditions under which ℓ_0 optimization problem (1.1) recovers the original function f as its unique solution. Unlike the popular literature (cf. compressed sensing), such conditions can not be based on sparsity only. This is illustrated through Example 1.3.1.

EXAMPLE 1.3.1. *For any $n \geq 4$, consider the four permutations $\sigma_1 = (1, 2)$, $\sigma_2 = (3, 4)$, $\sigma_3 = (1, 2)(3, 4)$ and $\sigma_4 = \mathrm{id}$, where id is the identity permutation. In addition, consider the partition $\lambda = (n-1, 1)$. Then, it is easy to see that*

$$M^\lambda(\sigma_1) + M^\lambda(\sigma_2) = M^\lambda(\sigma_3) + M^\lambda(\sigma_4).$$

We now consider three cases where a bound on sparsity is not sufficient to guarantee the existence of a unique solution to (1.1).

1. *Suppose that $f(\sigma_i) = p_i$, where $p_i \in \mathbb{R}_+$ for $1 \leq i \leq 4$, and $f(\sigma) = 0$ for all other $\sigma \in S_n$. Without loss of generality, let $p_1 \leq p_2$. Then,*

$$\begin{aligned}\hat{f}(\lambda) &= p_1 M^\lambda(\sigma_1) + p_2 M^\lambda(\sigma_2) + p_3 M^\lambda(\sigma_3) + p_4 M^\lambda(\sigma_4) \\ &= (p_2 - p_1) M^\lambda(\sigma_2) + (p_3 + p_1) M^\lambda(\sigma_3) + (p_4 + p_1) M^\lambda(\sigma_4).\end{aligned}$$

Thus, function g with $g(\sigma_2) = p_2 - p_1$, $g(\sigma_3) = p_3 + p_1$, $g(\sigma_4) = p_4 + p_1$ and $g(\sigma) = 0$ for all other $\sigma \in S_n$ is such that $\hat{g}(\lambda) = \hat{f}(\lambda)$ but

$\|g\|_0 = 3 < 4 = \|f\|_0$. *That is,* f *can not be recovered as the solution of* ℓ_0 *optimization problem* (1.1) *even when support of* f *is only* 4.

2. *Suppose that* $f(\sigma_1) = f(\sigma_2) = p$ *and* $f(\sigma) = 0$ *for all other* $\sigma \in S_n$. *Then,* $\hat{f}(\lambda) = pM^\lambda(\sigma_1) + pM^\lambda(\sigma_2) = pM^\lambda(\sigma_3) + pM^\lambda(\sigma_4)$. *Thus,* (1.1) *does not have a unique solution.*
3. *Now suppose that* $f(\sigma_i) = p_i$, *where* $p_i \in \mathbb{R}_+$ *for* $1 \le i \le 3$, *and* $f(\sigma) = 0$ *for all other* $\sigma \in S_n$. *Without loss of generality, let* $p_1 \le p_2$. *Then,*

$$\begin{aligned}\hat{f}(\lambda) &= p_1 M^\lambda(\sigma_1) + p_2 M^\lambda(\sigma_2) + p_3 M^\lambda(\sigma_3) \\ &= (p_2 - p_1) M^\lambda(\sigma_2) + (p_3 + p_1) M^\lambda(\sigma_3) + p_1 M^\lambda(\sigma_4).\end{aligned}$$

Here, note that $\left\{M^\lambda(\sigma_1), M^\lambda(\sigma_2), M^\lambda(\sigma_3)\right\}$ *is linearly independent, yet the solution to* (1.1) *is not unique.*

Question Two. The resolution of the first question will provide a way to recover f by means of solving the ℓ_0 optimization problem in (1.1). However, in general, it is computationally a hard problem. Therefore, interest is in computationally efficient algorithm to recover f.

Question Three. Once we identify the conditions for exact recovery of f, the next natural question to ask is "how restrictive are the conditions we imposed on f for exact recovery?" In other words, as mentioned above, we know that the sufficient conditions don't translate to a simple sparsity bound on functions, however, can we find a sparsity bound such that "most," if not all, functions that satisfy the sparsity bound can be recovered? We make the notion of "most" functions precise by proposing a natural random generative model for functions with a given sparsity. Then, for given a partition λ, we want to obtain $K(\lambda)$ so that if $K < K(\lambda)$ then recovery of f generated according to the generative model from $\hat{f}(\lambda)$ is possible with high probability.

This question is essentially an inquiry of whether the situation demonstrated by Example 1.3.1 contrived and special, or not. That is, if such examples happen with vanishingly low probability for a randomly chosen function. To this end, we describe a natural random function generation model.

DEFINITION 1.2 (Random Model). *Given* $K \in \mathbb{Z}_+$ *and an interval* $\mathscr{C} = [a, b]$, $0 < a < b$, *a random function* f *with sparsity* K *and values in* $\mathscr{C}$ *is generated as follows: choose* K *permutations from* S_n *independently and*

uniformly at random [2]*, say* $\sigma_1, \ldots, \sigma_K$*; select* K *values from* $\mathscr{C}$ *uniformly at random, say* $p_1, \ldots, p_K$*; then function* f *is defined as*

$$f(\sigma) = \begin{cases} p_i & \text{if} \quad \sigma = \sigma_i, \ 1 \leq i \leq K \\ 0 & \text{otherwise.} \end{cases}$$

We will denote this model as $R(K, \mathscr{C})$*.*

Question Four. Can we characterize the limitation on the ability of *any* algorithm to recover f from $\hat{f}(\lambda)$ (under random model)?

2. Known Results. Here we present answers to questions *One* to *Three* obtained by Jagabathula and Shah [JS08, JS09]. The question *Four* remains unresolved and primary purpose of this note is to bring this to attention of the reader. Rest of the section describes results from Jagabathula and Shah [JS08, JS09].

We start with recalling some notations. Let $\lambda = (\lambda_1, \ldots, \lambda_r)$ be the given partition of n. We wish to recover function $f : S_n \to \mathbb{R}_+$ from available information $\hat{f}(\lambda)$. Let the sparsity of f be K,

$$\text{supp}\,(f) = \{\sigma_1, \ldots, \sigma_K\}, \quad \text{and} \quad f(\sigma_k) = p_k, 1 \leq k \leq K.$$

Answers One & Two. In order to answer the first question, we need to find sufficiency conditions for recovering f through ℓ_0 optimization (1.1), while the second question requires a simple algorithm to recover the function. As mentioned earlier, a necessary condition for the recovery of the function through ℓ_0 optimization is the uniqueness of the sparsest solution. However, as shown in case 3 of Example 1.3.1, a simple sparsity bound is not sufficient to guarantee uniqueness of the sparsest solution. In order to understand this example further, it is convenient to consider the λ-bipartite matchings of the permutations and think of recovery as the decomposition of the λ-weighted bipartite graph into its corresponding λ-bipartite matchings. Note that while decomposing the λ-weighted bipartite graph, we can "peel-off" the bipartite matching corresponding to σ_3 with either weight p_3 or weight $p_3 + p_1$, resulting in the two sparsest solutions; here, without loss of generality, we have assumed that $p_1 \leq p_2$. Thus, even after having identified the permutation correctly, there is confusion regarding its weight. The reason is that the λ-bipartite matching of σ_3 is completely contained in the superposition of the bipartite matchings of σ_1 and σ_2, as illustrated in

[2]Throughout, we will assume that the random selection is done *with* replacement.

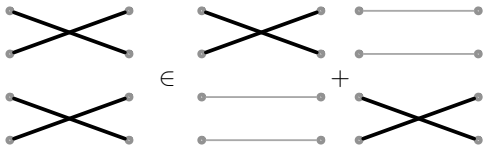

FIG 3. *The bipartite matching of σ_3 is completely "contained" in the superposition of the bipartite matchings of the other two permutations; thus, its weight is swamped by the weights of the other two permutations.*

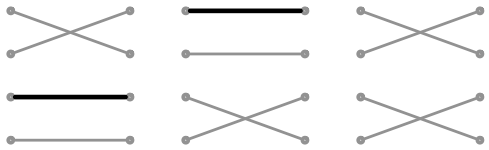

FIG 4. *The dark edges are the witness edges of the corresponding permutations; an edge of a permutation is a witness edge if it is present in no other permutation in the support. A permutation may have more than one witness edge – we have shown only one per permutation. As can be seen, the last permutation has no witness edges.*

Figure 3. Therefore, the weight of σ_3 is completely swamped by the weights of permutations σ_1 and σ_2. In order to avoid this, we require each permutation to have a "witness" or a "signature" in the partial information, which enables us to ascertain the correct weight of each permutation. In particular, we require each permutation in the support have at least one edge in its λ-bipartite matching that is not present in the λ-bipartite matching of any other permutation in the support. We call this condition the *unique witness condition*. Figure 4 illustrates this condition. This condition ensures that the bipartite matching of no permutation in the support is completely contained in the superposition of the bipartite matchings of other permutations in the support. The formal statement of this condition will be given shortly.

The unique witness condition is, however, not sufficient to guarantee uniqueness of sparsest solution. First, note that for exact recovery to be possible, the function values must all be distinct (case 2 of Example 1.3.1 is a simple counter example). However, as will be shown shortly, it is not sufficient for the function values to be all distinct. In fact, we want something stronger: the sum of the function values corresponding to any one subset of permutations should be distinct from the sum of function values over any other subset. The necessity of this condition is illustrated through the following example. Consider *any* four permutations $\sigma_1, \sigma_2, \sigma_3$ and σ_4, possibly satisfying the witness condition. We construct four new permutations by appending two new elements $n+1, n+2$ as follows: $\sigma_1' = \sigma(n+1)(n+2), \sigma_2' =$

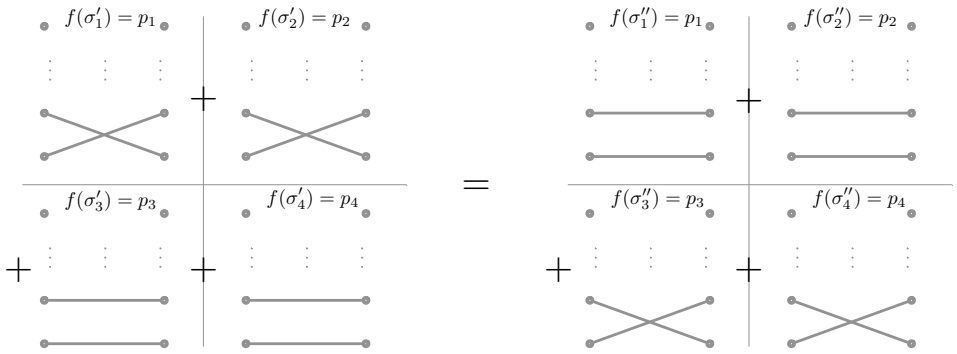

FIG 5. *The function values must be such that they are not only distinct, but their subset sums are also distinct. The figure illustrates an example where the sparsest solution is not unique when $p_1 + p_2 = p_3 + p_4$. The figure only shows the edges corresponding to the last two elements. The rest of the permutation may be chosen arbitrarily. σ_i' and σ_i'' are such that only the indicated edges are different with the unspecified edges remaining the same.*

$\sigma_2(n+1)(n+2), \sigma_3' = \sigma_3(n+1, n+2), \sigma_4' = \sigma_4(n+1, n+2)$, where we assume that all permutations are represented using the cycle notation. Now suppose we form a function f such that $f(\sigma_i) = p_i$ for $i = 1, 2, 3, 4$ and $f(\sigma) = 0$ otherwise; further, suppose that $p_1 + p_2 = p_3 + p_4$. It is easy to see that given the λ-partial information corresponding to $\lambda = (n-1, 1)$, we can also decompose the partial information into the function g with support $\sigma_1'' = \sigma(n+1, n+2), \sigma_2'' = \sigma_2(n+1, n+2), \sigma_3'' = \sigma_3(n+1)(n+2), \sigma_4'' = \sigma_4(n+1)(n+2)$. This is shown in Figure 5. Thus, we require the function values to be such that they are not only distinct, but their subset sums are also distinct. As we show shortly, the above two conditions are, in fact, sufficient for exact recovery. More formally, we impose the following conditions on f.

CONDITION 1 (Sufficiency Conditions). Let f satisfy the following:

- *Unique Witness*: for any $\sigma \in \text{supp}(f)$, there exists $1 \leq i_\sigma, j_\sigma \leq D_\lambda$ such that $M^\lambda_{i_\sigma j_\sigma}(\sigma) = 1$, but $M^\lambda_{i_\sigma j_\sigma}(\sigma') = 0$, for all $\sigma' (\neq \sigma) \in \text{supp}(f)$.
- *Linear Independence*: for any collection of integers $c_1, \ldots, c_K$ taking values in $\{-K, \ldots, K\}$, $\sum_{k=1}^K c_k p_k \neq 0$, unless $c_1 = \ldots = c_K = 0$.

A result from [JS08, JS09] is recalled that establishes Condition 1 as sufficient for recovery of f as the unique solution of ℓ_0 optimization problem. Further, it allows for a simple, iterative recovery algorithm. Thus, Theorem 2.1 provides answers to questions *One* and *Two* of Section 1.3.

THEOREM 2.1. *Under Condition 1, the function f is the unique solution of the ℓ_0 optimization problem (1.1). Further, a simple, iterative algorithm* ALGO*, described in [JS08, JS09], recovers f.*

Linear Programs Don't Work. Theorem 2.1 states that under Condition 1, the ℓ_0 optimization recovers f and ALGO is a simple iterative algorithm to recover it. In the context of compressive sensing literature (cf. [CRT06b, CRT06a, Don06, BGI+08]), it has been shown that convex relaxation of ℓ_0 optimization, the Linear Programing relaxation, have the same solution in similar scenarios. Therefore, it is natural to wonder whether such a relaxation would work in this setup. To this end, consider the following Linear Programing relaxation of (1.1) stated as the following ℓ_1 minimization problem:

$$\begin{array}{lll} \text{minimize} & \|g\|_1 \quad \text{over} & g : S_n \to \mathbb{R}_+ \\ \text{subject to} & \hat{g}(\lambda) = \hat{f}(\lambda). & \end{array} \tag{2.1}$$

THEOREM 2.2. *Consider a function f randomly generated as per Definition 1.2 with sparsity $K \geq 2$. Then, with probability $1 - o(1)$ the solution to ℓ_1 minimization (2.1) is not unique.*

Answer Three. Here interest is in the conditions for high probability recoverability of a random function f in terms of its sparsity. That is, we wish to identify lower bound on the high probability recoverability threshold $K(\lambda)$. In what follows, this is stated starting with specific cases followed by general result to better explain the dependency of $K(\lambda)$ on D_λ. Again these results are from [JS08, JS09].

Case 1: $\lambda = (n-1, 1)$. Here $D_\lambda = n$ and $\hat{f}(\lambda)$ provides the *first order* marginal information. As stated next, for this case the achievable recoverability threshold $K(\lambda)$ scales[3] as $n \log n$.

THEOREM 2.3. *A function f generated randomly as per Definition 1.2 can be recovered by* ALGO *with probability $1 - o(1)$ as long as $K \leq (1 - \varepsilon) n \log n$ for any fixed $\varepsilon > 0$.*

Case 2: $\lambda = (n - m, m)$ with $1 < m = O(1)$. Here $D_\lambda = \Theta(n^m)$ and $\hat{f}(\lambda)$ provides the *mth order* marginal information. As stated next, for this case we find that $K(\lambda)$ scales at least as $n^m \log n$.

[3]Throughout this paper, by log we mean the natural logarithm, i.e. $\log_e$, unless otherwise stated.

THEOREM 2.4. *A function f generated randomly as per Definition 1.2 can be recovered by* ALGO *based on $\hat{f}(\lambda)$ for $\lambda = (n-m, m), m = O(1)$, with probability $1 - o(1)$ as long as $K \leq \frac{(1-\varepsilon)}{m!} n^m \log n$ for any fixed $\varepsilon > 0$.*

In general, for any λ with $\lambda_1 = n - m$ and $m = O(1)$, arguments of Theorem 2.4 can be adapted to show that $K(\lambda)$ scales as $n^m \log n$. Theorems 2.3 and 2.4 suggest that the recoverability threshold scales $D_\lambda \log D_\lambda$ for $\lambda = (\lambda_1, \ldots, \lambda_r)$ with $\lambda_1 = n - m$ for $m = O(1)$. Next, we consider the case of more general λ.

Case 3: $\lambda = (\lambda_1, \ldots, \lambda_r)$ with $\lambda_1 = n - O\left(n^{\frac{2}{9}-\delta}\right)$ for any $\delta > 0$. As stated next, for this case, the recoverability threshold $K(\lambda)$ scales at least as $D_\lambda \log\log D_\lambda$.

THEOREM 2.5. *A function f generated randomly as per Definition 1.2 can be recovered from $\hat{f}(\lambda)$ by* ALGO *for $\lambda = (\lambda_1, \ldots, \lambda_r)$ with $\lambda_1 = n - n^{\frac{2}{9}-\delta}$ for any $\delta > 0$, with probability $1 - o(1)$ as long as $K \leq (1-\varepsilon) D_\lambda \log\log D_\lambda$ for any fixed $\varepsilon > 0$.*

Case 4: Any $\lambda = (\lambda_1, \ldots, \lambda_r)$. The results stated thus far suggest that the threshold is essentially D_λ, ignoring the logarithm term. For general λ, we establish a bound on $K(\lambda)$ as stated in Theorem 2.6 below. Before stating the result, we introduce some notation. For given λ, define $\alpha = (\alpha_1, \ldots, \alpha_r)$ with $\alpha_i = \lambda_i / n,\ 1 \leq i \leq r$. Let

$$H(\alpha) = -\sum_{i=1}^{r} \alpha_i \log \alpha_i, \quad \text{and} \quad H'(\alpha) = -\sum_{i=2}^{r} \alpha_i \log \alpha_i.$$

THEOREM 2.6. *Given $\lambda = (\lambda_1, \ldots, \lambda_r)$, a function f generated randomly as per Definition 1.2 can be recovered from $\hat{f}(\lambda)$ by* ALGO *with probability $1 - o(1)$ as long as*

$$K \;\leq\; C\, D_\lambda^{\gamma(\alpha)}, \tag{2.2}$$

where

$$\gamma(\alpha) = \frac{M}{M+1}\left[1 - O(1)\frac{H(\alpha) - H'(\alpha)}{H(\alpha)}\right], \quad \textit{with} \quad M = \left\lfloor \frac{1}{1-\alpha_1} \right\rfloor$$

and $0 < C < \infty$ is a constant.

At a first glance, the above result seems very different from the crisp formulas of Theorems 2.3-2.5. Therefore, let us consider a few special cases. First, observe that as $\alpha_1 \uparrow 1$, $M/(M+1) \to 1$. Further, as stated in Lemma 2.1, $H'(\alpha)/H(\alpha) \to 1$. Thus, we find that the bound on sparsity essentially scales as D_λ. Note that the cases 1, 2 and 3 fall squarely under this scenario since $\alpha_1 = \lambda_1/n = 1 - o(1)$. Thus, this general result contains the results of Theorems 2.3-2.5 (ignoring the logarithm terms).

Next, consider the other extreme of $\alpha_1 \downarrow 0$. Then, $M \to 1$ and again by Lemma 2.1, $H'(\alpha)/H(\alpha) \to 1$. Therefore, the bound on sparsity scales as $\sqrt{D_\lambda}$. This ought to be the case because for $\lambda = (1, \ldots, 1)$ we have $\alpha_1 = 1/n \to 1$, $D_\lambda = n!$, and unique signature property holds only up to $o(\sqrt{D_\lambda}) = o(\sqrt{n!})$ due to the standard Birthday paradox.

In summary, Theorem 2.6 appears reasonably tight for the general form of partial information λ. We now state the Lemma 2.1 used above.

LEMMA 2.1. *Consider any $\alpha = (\alpha_1, \ldots, \alpha_r)$ with $1 \geq \alpha_1 \geq \cdots \geq \alpha_r \geq 0$ and $\sum_{i=1}^r \alpha_r = 1$. Then,*

$$\lim_{\alpha_1 \uparrow 1} \frac{H'(\alpha)}{H(\alpha)} = 1,$$

$$\lim_{\alpha_1 \downarrow 0} \frac{H'(\alpha)}{H(\alpha)} = 1.$$

3. Question *Four*: A conjecture. Question *Four* asks for fundamental limitation on the ability to recover f, in terms of support (or sparsity) of f, from $\hat{f}(\lambda)$ by any algorithm under the random model. To start with, the λ-partial information is D_λ^2 dimensional. Therefore, clearly a bound is D_λ^2. However, it is quite weak. For example consider scenario when $\lambda = (1, \ldots, 1)$. In this case, $D_\lambda = n!$. In this case, indeed only D_λ out of D_λ^2 entries are useful. Therefore, the answer to question *Four* should be smaller than D_λ^2 in general. Indeed, an evidence of this qualitative statement is provided in [JS09] (which is recalled in Section 3.1). We state the following conjecture.

CONJECTURE 3.1. *Given $\lambda = (\lambda_1, \ldots, \lambda_r)$, a function f generated randomly as per Definition 1.2 can not recovered from $\hat{f}(\lambda)$ by any algorithm with probability $1 - o(1)$ as long as*

$$K = \Omega\Big(D_\lambda^{1+\varepsilon}\Big), \tag{3.1}$$

for any $\varepsilon > 0$.

3.1. *A qualitative evidence.* Here we present a qualitative evidence (from [JS09]) supporting the intuition that the fundamental limitation must be significantly smaller than D_λ^2. As stated below, this utilizes appropriate information theoretic setup.

To this end, as in random model $R(K, \mathscr{C})$, consider a function f generated with given K and λ. For technical reasons (or limitations), we will assume that the values p_is are chosen from a discrete set. Specifically, let each p_i be chosen from integers $\{1, \ldots, T\}$ instead of compact set $\mathscr{C}$. We will denote this random model by $R(K, T)$.

Consider any algorithm that attempts to recover f from $\hat{f}(\lambda)$ under $R(K, T)$. Let h be the estimation of the algorithm. Define probability of error of the algorithm as

$$p_{\text{err}} = \Pr(h \neq f).$$

THEOREM 3.1. *With respect to random model $R(K, T)$, the probability of error is uniformly bounded away from 0 for all n large enough and any λ, if*

$$K \geq \frac{3D_\lambda^2}{n \log n} \left[\log \left(\frac{D_\lambda^2}{n \log n} \vee T \right) \right]. \tag{3.2}$$

REFERENCES

[BGI+08] R. Berinde, AC Gilbert, P. Indyk, H. Karloff, and MJ Strauss, *Combining geometry and combinatorics: A unified approach to sparse signal recovery*, Preprint (2008).

[CRT06a] EJ Candes, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on Information Theory **52** (2006), no. 2, 489–509.

[CRT06b] E.J. Candes, J.K. Romberg, and T. Tao, *Stable signal recovery from incomplete and inaccurate measurements*, Communications on Pure and Applied Mathematics **59** (2006), no. 8.

[Don06] DL Donoho, *Compressed sensing*, IEEE Transactions on Information Theory **52** (2006), no. 4, 1289–1306.

[JS08] S. Jagabathula and D. Shah, *Inferring rankings under constrained sensing*, NIPS, 2008, pp. 7–1.

[JS09] ———, *Inferring rankings using constrained sensing*, Submitted, available at arxiv.org (2009).

SRIKANTH JAGABATHULA
E-MAIL: jskanth@mit.edu

DEVAVRAT SHAH
E-MAIL: devavrat@mit.edu