EXACT FOURIER-SPARSE RECOVERY OVER HYPERCUBE: SIMPLER AND STRONGER

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the problem of exact recovery of Fourier-sparse functions over the Boolean hypercube $\mathbf{F}_2^n$, a fundamental problem at the intersection of learning theory, signal processing, and inverse problems. Given oracle access to such a function whose Fourier spectrum is supported on at most k coefficients, the goal is to identify all nonzero Fourier coefficients exactly using as few queries as possible.

An $\Omega(nk)$ query lower bound is known for this problem, which also implies an $\Omega(nk)$ lower bound on the running time. Recent works have made significant progress, developing algorithms whose query and time complexities are within polylogarithmic factors of these lower bounds. However, these algorithms often rely on relatively sophisticated machinery, making the underlying recovery process less transparent.

In this work, we present a randomized algorithm that achieves polylogarithmic improvements over the best known results, while being substantially simpler in its design. Our algorithm uses $O(nk)$ queries and $O(nk \log k)$ time, making it sample-optimal up to constant factors, with running time within an $O(\log k)$ factor of the corresponding lower bound.

Beyond these quantitative improvements, we believe our approach enables a substantially simpler and more transparent recovery framework, which may prove useful in broader sparse-recovery settings.

1 INTRODUCTION

This paper studies the problem of exact recovery of Fourier-sparse functions over the Boolean cube. Given oracle access to a function $f : \mathbb{F}_2^n \to \mathbb{R}$ whose Fourier spectrum is supported on at most k coefficients, the goal is to identify all nonzero Fourier coefficients exactly using as few queries as possible. Our main result (Theorem 1) is a randomized algorithm that solves this problem using $O(nk)$ queries and $O(nk \log k)$ time. In the regime $k = \mathrm{poly}(n)$, this improves on the best known algorithm (Amrollahi et al., 2019) by a $\Theta(\log k)$ factor in queries and a $\Theta(\log^3 k)$ factor in running time. Beyond these quantitative improvements, our main contribution is a substantially simpler recovery algorithm and analysis. By exploiting structural properties of the Fourier spectrum induced by random subspace restrictions, we obtain a recovery procedure that is both simple and transparent. We hope that our perspective makes the underlying ideas more accessible and potentially useful in other sparse-recovery settings.

We begin by briefly reviewing the broader context of sparse recovery and the relevance of Fourier-sparse functions over the Hypercube in large-scale data analysis. Readers, already familiar with sparse recovery and related results in Fourier setting, may skip this discussion and proceed directly to the main theorem of this paper (Theorem 1).

Sparse Recovery: Inverse problems seek to recover hidden structure from observations, and many learning problems can be naturally formulated in this framework (De Vito et al., 2005). Such problems, however, are often challenging in their full generality. One of the most influential paradigms for making them tractable is *sparse recovery*, which shows that sparse signals can often be recovered efficiently from substantially fewer measurements (Candès & Tao, 2005; Candès et al., 2006; Donoho, 2006). Sparse

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

recovery has since become an important topic across theoretical computer science, applied mathematics, signal processing, and machine learning.

Fourier Sparse Recovery: Fourier analysis provides an important setting for sparse recovery. The classical problem considers functions $f : \mathbb{Z}_N \to \mathbb{C}$ with sparse Fourier spectra. A major line of work develops recovery algorithms from random measurements, including approaches based on the Restricted Isometry Property (RIP) of subsampled Fourier and Hadamard matrices (Bourgain, 2014; Cheraghchi et al., 2013; Haviv & Regev, 2016; Candès & Tao, 2006; Blumensath & Davies, 2008; Needell & Tropp, 2009; Needell & Vershynin, 2009). Although these methods achieve near-optimal sample complexity, their running time is generally at least linear in the ambient dimension N. Moreover, these approaches typically operate in the random-measurement model, where the algorithm only sees random linear measurements or random samples of the function.

A different line of work studies Fourier sparse recovery for functions $f : \mathbb{Z}_N \to \mathbb{C}$ in the *query-access model*, where the algorithm can evaluate f at chosen points (Mansour, 1992; Akavia et al., 2003; Iwen, 2010; Hassanieh et al., 2012b; Pawar & Ramchandran, 2014; Indyk & Kapralov, 2014; Kapralov, 2017). The central idea is to use these function evaluations to implicitly hash Fourier coefficients into buckets and recover coefficients from sufficiently isolated buckets, following the general paradigm of Count-Sketch (Charikar et al., 2002). This stronger access model offers significant computational advantages over earlier methods based solely on random measurements (Bourgain, 2014; Needell & Vershynin, 2010), enabling recovery algorithms with near-optimal bounds in both sample complexity and running time (Hassanieh et al., 2012a).

Fourier Sparse Recovery on the Boolean Cube: We study the analogous problem for real-valued functions over the Boolean hypercube, namely functions $f : \mathbb{F}_2^n \to \mathbb{R}$. Fourier sparsity arises naturally in several structured classes of functions on the Boolean hypercube. For instance, a decision tree of depth d has Fourier degree at most d and Fourier support of size at most 2^d. Similarly, the cut function of a graph with m edges has Fourier degree at most 2 and Fourier sparsity $O(m)$, while for a hypergraph, the Fourier degree of its cut function is bounded by the maximum hyperedge size. These connections have motivated the use of Fourier methods for learning, graph sketching, and the recovery of structured set functions from limited observations (Stobbe & Krause, 2012; Karger, 1999; Ahn et al., 2012; Kocaoglu et al., 2014; Kogan & Krauthgamer, 2015).

Having established this background, we now turn to our main result.

Theorem 1. *There exists a non-adaptive randomized algorithm (Algorithm 2) that, given oracle access to a function $f : \mathbb{F}_2^n \to \mathbb{R}$ with Fourier sparsity at most k, recovers all nonzero Fourier coefficients exactly using $O(nk)$ queries and $O(nk \log k)$ time, with probability at least $2/3$.*

For a general real-valued f, the output consists of k frequencies, each represented by n bits, together with their corresponding Fourier coefficients. Thus, an $\Omega(nk)$ lower bound is natural (see Haviv & Regev (2015) for a formal argument), and our $O(nk)$ query complexity matches this bound up to constant factors. While sample-optimal guarantees of this form were known for sparse recovery over $\mathbb{Z}_n$ (Hassanieh et al., 2012a), to the best of our knowledge, such guarantees were not previously known for Fourier-sparse functions over $\mathbb{F}_2^n$. Our algorithm therefore gives the first $O(nk)$-query guarantee in this setting, while running in $O(nk \log k)$ time. This running time is within a logarithmic factor of the corresponding $\Omega(nk)$ lower bound, which follows trivially from the $\Omega(nk)$ query lower bound.

Comparison with Kushilevitz–Mansour. The problem of learning sparse Fourier representations was first studied by Kushilevitz and Mansour (Kushilevitz & Mansour, 1993), building on the work of Goldreich and Levin (Goldreich & Levin, 1989). Their algorithm does not recover a k-Fourier-sparse function exactly; instead, it produces an approximation in the ℓ_2 sense. For Boolean-valued functions, exact recovery can be obtained by combining their approach with the granularity theorem for Fourier-sparse Boolean functions (Gopalan et al., 2011). This, however, requires $O(nk^6)$ queries and does not extend to general real-valued functions.

Comparison with Amrollahi et al. Subsequently, the problem was studied in several works (Feldman et al., 2006; Negahban & Shah, 2012; Stobbe & Krause, 2012; Amrollahi et al., 2019). Among these, to the best of our knowledge, the strongest results are due to Amrollahi et al. (2019), who study the exact recovery of k-Fourier-sparse functions whose Fourier support is confined to frequencies of degree at most d. Their algorithm uses $O(kd \log n)$ queries and $O(kn \log^2 k \log n \log d)$ time. We are interested in the general setting, where the support may contain frequencies of arbitrary degree, i.e. $d = \Theta(n)$. In this regime the two sets of bounds compare as follows.

Work	Queries	Time
Amrollahi et al. (2019)	$O(kn \log n)$	$O(kn \log^2 k \log^2 n)$
This work	$O(kn)$	$O(kn \log k)$

In the standard regime $k = \mathrm{poly}(n)$ (so that $\log k = \Theta(\log n)$), our algorithm uses a $\Theta(\log k)$ factor fewer queries and is faster by a $\Theta(\log^3 k)$ factor. Beyond this quantitative gain, the two approaches also differ structurally, and it is worth understanding how.

Algorithms of this kind, which have access to the function through queries, including that of Amrollahi et al. (2019), belong to the broad family of sparse Hadamard transforms (Hassanieh et al., 2012a; Kapralov, 2016; Indyk et al., 2014) and share a common two-stage template. First, a *hashing* stage partitions the frequencies into buckets so that enough of them become isolated. Second, an *estimate-and-recurse* stage identifies singleton buckets and recovers their frequencies and coefficients, typically via residual-energy estimation. The main difficulty is to construct a hashing scheme with good collision properties while controlling the spectral leakage the scheme introduces.

Our algorithm follows this template but instantiates it in a simpler way. Rather than explicitly constructing a hash family in the Fourier domain, we restrict the input function to a uniformly random affine subspace. A basic structural property of the Fourier transform over $\mathbb{F}_2^n$ guarantees that such a restriction automatically induces the required partition of the spectrum. The hashing is thus *implicit* in the algebraic structure of the transform, rather than implemented through a separately designed scheme. This is a useful feature of the Fourier transform over the Boolean cube that is not directly available for many other orthogonal transforms, including some that satisfy analogous isometry properties. Our approach takes advantage of this structure in a way that leads to a particularly simple recovery procedure. Building on this, we further simplify the recovery procedure by avoiding several computationally expensive steps required in earlier approaches.

Organization

The rest of the paper is organized as follows. Section 2 introduces the necessary background on Fourier analysis over the Boolean cube. Section 3 presents a warm-up algorithm that isolates our core idea, implicit hashing via random affine subspace restriction, in its simplest form, and proves its guarantee (Theorem 2). Section 4 presents the main algorithm (Algorithm 2) and proves our main theorem (Theorem 1). The appendix provides some details from the proof of Theorem 1 that could not be accommodated due to space constraints, and discusses several open direction arising from our results.

2 Background

We recall the basic framework of Fourier analysis over the Boolean cube (Ceccherini-Silberstein et al., 2018; Stein & Shakarchi, 2003). A function $f : \mathbb{F}_2^n \to \mathbb{R}$ admits the expansion

$$f(x) = \sum_{\alpha \in \mathbb{F}_2^n} \hat{f}(\alpha)\, \chi_\alpha(x), \text{ where } \chi_\alpha(x) := (-1)^{\langle \alpha, x \rangle}.$$

The Fourier coefficients are given by

$$\hat{f}(\alpha) = \mathop{\mathbb{E}}_{x \in \mathbb{F}_2^n} [f(x) \chi_\alpha(x)],$$

and measure the correlation of f with each parity function. The collection $\{\hat{f}(\alpha)\}$ is called the Fourier spectrum of f. We say that f is k-Fourier sparse if $|\mathrm{supp}(\hat{f})| \le k$, where $\mathrm{supp}(\hat{f}) = \{\alpha \in \mathbb{F}_2^n : \hat{f}(\alpha) \neq 0\}$.

We also recall two linear algebra concepts useful for studying Fourier spectra of functions defined over subspaces of $\mathbb{F}_2^n$: orthogonal subspaces and direct sums. For a subspace $A \subseteq \mathbb{F}_2^n$, its orthogonal complement is

$$A^\perp := \{\gamma \in \mathbb{F}_2^n : \langle \gamma, \gamma' \rangle = 0 \text{ for all } \gamma' \in A\},$$

and it satisfies $\dim(A) + \dim(A^\perp) = n$. The codimension of A is $\operatorname{codim}(A) = n - \dim(A)$. Two subspaces A and B are independent if $a + b \neq 0$ for any nonzero $a \in A$ and $b \in B$, and if every element of $\mathbb{F}_2^n$ can be expressed uniquely as $a + b$, we write $A \oplus B = \mathbb{F}_2^n$ and call it a direct sum. In this case, $\dim(A) + \dim(B) = n$. Note that A and $A^\perp$ are not necessarily independent. If A and B are independent, the affine subspaces $b + A$ for distinct $b \in B$ are disjoint. Cosets of A can be naturally indexed by evaluating a basis of $A^\perp$ on elements $x \in \mathbb{F}_2^n$.

3 A WARM-UP ALGORITHM

The main ingredient is understanding how the Fourier spectrum of f behaves when f is restricted to a random affine subspace. We now begin by formalizing this notion. Let $H \leq \mathbb{F}_2^n$ be a linear subspace of dimension d. For any coset $A = a + H$, we define the restriction of f to A by

$$f_A(x) := f(a + x), \qquad x \in H,$$

viewing f_A as a function on H.

Algorithm 1: Simple Sparse Fourier Recovery

Input: Query access to $f : \mathbb{F}_2^n \to \mathbb{R}$, sparsity k

Output: $\{(\alpha, \hat{f}(\alpha))\}_{\alpha \in \operatorname{supp}(\hat{f})}$

Sample a random subspace $H \leq \mathbb{F}_2^n$ with $\dim(H) = \lceil 2\log(4k) \rceil$;

Compute $\widehat{f_a}(\gamma)$ for all $\gamma \in W$ and $a \in \{0, e_1, \ldots, e_n\}$;

for $\gamma \in W$ with $\widehat{f_0}(\gamma) \neq 0$ **do**

  $\alpha_i \leftarrow \dfrac{1 - \widehat{f_{e_i}}(\gamma)/\widehat{f_0}(\gamma)}{2}$ for $i = 1, \ldots, n$;

  Output $\big((\alpha_1, \ldots, \alpha_n),\ \widehat{f_0}(\gamma)\big)$;

end

Fourier coefficients of the restriction. When we restrict f to a coset of H, the resulting function lives on H, and its characters are indexed not by all of $\mathbb{F}_2^n$ but by the characters of H. Two frequencies $\alpha, \beta \in \mathbb{F}_2^n$ induce the same character on H exactly when $\alpha - \beta \in H^\perp$; that is, the character of α on H depends only on the coset $\alpha + H^\perp$. To index these cosets by concrete vectors, we fix, once and for all, a subspace $W \leq \mathbb{F}_2^n$ complementary to $H^\perp$, so that

$$W \oplus H^\perp = \mathbb{F}_2^n.$$

The subspace W has dimension $\dim(H)$ and serves a single purpose: it supplies a canonical representative for each coset of $H^\perp$, and hence a concrete label for each bucket into which the Fourier spectrum is hashed. Concretely, we write $\alpha|_H$ for the unique element $w \in W$ with $\alpha - w \in H^\perp$, i.e. the representative in W of the equivalence class of α modulo $H^\perp$. This defines the surjective linear map

$$\pi_H : \mathbb{F}_2^n \longrightarrow W, \qquad \alpha \mapsto \alpha\big|_H, \qquad \ker \pi_H = H^\perp,$$

so that π_H is precisely the quotient map $\mathbb{F}_2^n \to \mathbb{F}_2^n / H^\perp$ read off in the coordinates supplied by W. Throughout, "the bucket of α" means the value $\alpha|_H \in W$.

Note that, Fourier coefficients of f_A are given by

$$\widehat{f_A}(\gamma) = \sum_{\substack{\alpha \in \operatorname{supp}(\hat{f}) \\ \alpha|_H = \gamma}} \hat{f}(\alpha)\, \chi_\alpha(a), \qquad \gamma \in W.$$

In other words, $\widehat{f_A}(\gamma)$ aggregates all Fourier coefficients $\hat{f}(\alpha)$ of f whose character α restricts to γ on H, weighted by the phase $\chi_\alpha(a)$ determined by the coset representative a. To see why, recall that,

for every $x \in H$, we may write the Fourier expansion of f as

$$f(a+x) = \sum_{\alpha \in \mathbb{F}_2^n} \hat{f}(\alpha)\chi_\alpha(a+x).$$

Hence,

$$f_A(x) = \sum_{\alpha \in \mathbb{F}_2^n} \hat{f}(\alpha)\chi_\alpha(a+x).$$

Using the multiplicativity of characters, $\chi_\alpha(a+x) = \chi_\alpha(a)\chi_\alpha(x)$, we obtain

$$f_A(x) = \sum_{\alpha \in \mathbb{F}_2^n} \hat{f}(\alpha)\chi_\alpha(a)\chi_\alpha(x).$$

Now observe that, when restricted to H, the character χ_α depends only on the restriction $\alpha|_H$. In particular, if $\alpha|_H = \gamma$, then $\chi_\alpha(x) = \chi_\gamma(x)$ for all $x \in H$. Grouping together all terms having the same restriction to H, we obtain

$$f_A(x) = \sum_{\gamma \in W} \left(\sum_{\substack{\alpha \in \mathbb{F}_2^n \\ \alpha|_H = \gamma}} \hat{f}(\alpha)\chi_\alpha(a) \right) \chi_\gamma(x).$$

Since this is precisely the Fourier expansion of f_A over the domain H, it follows that for every $\gamma \in W$,

$$\widehat{f_A}(\gamma) = \sum_{\substack{\alpha \in \mathrm{supp}(\hat{f}) \\ \alpha|_H = \gamma}} \hat{f}(\alpha)\chi_\alpha(a).$$

Fourier Separation by Random Subspace. We now show that a uniformly random subspace of dimension $2\log k + O(1)$ acts as a hashing mechanism that, with high probability, separates all k nonzero Fourier coefficients of a k-Fourier-sparse function.

Lemma 1. *Let $f : \mathbb{F}_2^n \to \mathbb{R}$ with $|\mathrm{supp}(\hat{f})| \le k$. Let H be a uniformly random subspace of $\mathbb{F}_2^n$ of dimension $d \ge 2\log 4k$. Then, except with probability $1/16$, the map $\alpha \mapsto \alpha|_H$ is injective on $\mathrm{supp}(\hat{f})$. In that case, for every $\gamma \in W$ at most one $\alpha \in \mathrm{supp}(\hat{f})$ satisfies $\alpha|_H = \gamma$, and for every coset $A = a + H$ we have $\widehat{f_A}(\gamma) = \hat{f}(\alpha)\,\chi_\alpha(a)$ for that α (and $\widehat{f_A}(\gamma) = 0$ if there is none).*

Proof. We say a *collision* occurs if there exist distinct $\alpha, \beta \in \mathrm{supp}(\hat{f})$ with $\alpha|_H = \beta|_H$, equivalently $(\alpha+\beta) \in H^\perp$. For any fixed such pair,

$$\Pr_H\big[(\alpha+\beta) \in H^\perp\big] = \frac{2^{n-d}-1}{2^n-1} \le 2^{-d},$$

since $H^\perp$ has dimension $n-d$ and $\alpha+\beta \neq 0$.

There are at most $\binom{k}{2} < k^2$ such pairs in $\mathrm{supp}(\hat{f})$, so by a union bound,

$$\Pr_H[\text{any collision occurs}] \le k^2 \cdot 2^{-d} \le \frac{1}{16},$$

for $d \ge 2\log 4k$. Conditioning on no collision, for each γ there is at most one $\alpha \in \mathrm{supp}(\hat{f})$ with $\alpha|_H = \gamma$, and the formula for $\widehat{f_A}(\gamma)$ reduces from the general sum to the single term $\hat{f}(\alpha)\,\chi_\alpha(a)$. □

With these insights in hand, we now present a simple algorithm (Algorithm 1) that exactly recovers all Fourier coefficients of a k-sparse function. The theoretical guarantees of the algorithm are summarized in the following theorem.

Theorem 2. *Let $f : \mathbb{F}_2^n \to \mathbb{R}$ be k-Fourier sparse. There exists a randomized algorithm that exactly recovers $\mathrm{supp}(\hat{f})$ along with their values $\hat{f}(\alpha)$, except with probability $1/16$, using $\tilde{O}(nk^2)$ queries and $\tilde{O}(nk^2)$ time.*

Proof. In Algorithm 1, we sample a random subspace $H \leq \mathbb{F}_2^n$ of dimension $d = \lceil 2\log(4k) \rceil$. By Lemma 1, except with probability $1/16$, the restriction map $\alpha \mapsto \alpha|_H$ is injective on $\mathrm{supp}(\hat{f})$. Consequently, each $\gamma \in W$ with $\widehat{f_0}(\gamma) \neq 0$ corresponds to exactly one $\alpha \in \mathrm{supp}(\hat{f})$, and there are at most k such γ. For each such γ, $\widehat{f_a}(\gamma) = \hat{f}(\alpha)\,\chi_\alpha(a)$.

Let $e_1, \ldots, e_n$ denote the standard unit vectors in $\mathbb{F}_2^n$, and let $e_0 = \mathbf{0}$. For each $i \in \{0, 1, \ldots, n\}$, define the coset $A_i = e_i + H$.

Note that

$$\widehat{f_{e_i}}(\gamma)/\widehat{f_0}(\gamma) = \chi_\alpha(e_i) = (-1)^{\langle \alpha, e_i \rangle} = (-1)^{\alpha_i}.$$

Thus, defining

$$b_i := \begin{cases} 0 & \text{if } \widehat{f_{e_i}}(\gamma)/\widehat{f_0}(\gamma) = +1, \\ 1 & \text{if } \widehat{f_{e_i}}(\gamma)/\widehat{f_0}(\gamma) = -1, \end{cases}$$

we recover $\alpha_i = b_i$ for each $i \in [n]$. Therefore, $\alpha = (b_1, \ldots, b_n)$ is recovered coordinate-wise, without requiring any linear algebra. Finally, the Fourier coefficient is obtained as $\hat{f}(\alpha) = \widehat{f_0}(\gamma)$. It is easy to see that the total number of queries is $(n+1) \cdot 2^d = \tilde{O}(nk^2)$, and the overall running time is $\tilde{O}(nk^2)$. □

4 The Main Algorithm

In this section, we present an improved version of the sparse Fourier recovery algorithm (Algorithm 2). The algorithm follows the same basic idea as Algorithm 1, but uses a much smaller random subspace: its size is only $O(k)$, rather than $O(k^2)$. This reduction makes each round substantially cheaper, but it also means that more Fourier coefficients may collide in the same bucket. Consequently, a single round may not recover all coefficients.

To overcome this, we apply the recovery procedure iteratively. In each round, we recover the coefficients that can be identified unambiguously and add them to a dictionary. We then subtract these recovered contributions from the function and repeat the procedure on the remaining residual. The key observation is that the residual becomes rapidly sparser after each round, so later rounds become progressively cheaper.

The intuition is simple. Consider a Fourier coefficient that falls into a bucket by itself. Since there are no other coefficients interfering with it, the coefficient can be recovered exactly from the values of that bucket under suitable shifts. On the other hand, if several coefficients fall into the same bucket, the bucket may not be decoded correctly. Thus, the only coefficients that can cause trouble are those involved in collisions. The random choice of the subspace makes such collisions unlikely. In particular, when the residual contains relatively few coefficients compared with the number of available buckets, most coefficients are isolated. Hence, one round removes almost all of the current residual, leaving only a small fraction behind. The same argument can then be applied again to this smaller residual. There is an important additional benefit. As the rounds progress, we reduce the size of the subspace used for hashing. The number of buckets decreases geometrically, but the residual sparsity decreases even faster. Thus, the residual becomes increasingly sparse relative to the number of buckets, making collisions progressively less likely. This geometric decrease is what allows the total query complexity to remain $O(nk)$, even though several rounds are performed.

We first formalize the hashing and isolation properties that underlie this argument. We then show that a single round reduces the residual to a small multiple of the number of coefficients involved in collisions. Finally, we combine these two facts across all rounds to obtain the overall success probability and complexity bounds. The formal analysis appears below.

4.1 Analysis of Algorithm 2

The analysis has three steps, which mirror the intuition above. First, in each round most coefficients of the residual are isolated (Lemma 2). Second, a single round always makes progress: the new residual is at most $\frac{3}{2}$ times as large as the number of coefficients that collided (Lemma 3). Third, combining the two with the schedule of subspace sizes shows that the residual reaches zero.

Setup. All logarithms are to base 2, and in running-time bounds $\log k$ stands for $\max\{1, \log k\}$. We may assume $100k < 2^n$. Otherwise we query f on all of $\mathbb{F}_2^n$, using $2^n \le 100k$ queries, and compute $\hat{f}$ by a fast Walsh–Hadamard transform in time $O(n2^n) = O(nk)$. Under this assumption, every subspace used by the algorithm has dimension at most $\lceil \log(100k) \rceil \le n$.

Let D_r denote the dictionary at the start of round r, with $D_0 = \emptyset$. We regard D_r as a function on $\mathbb{F}_2^n$ that vanishes outside its keys. The residual at the start of round r is then

$$f^{(r)} = f - \sum_{\alpha \in D_r} D_r(\alpha)\, \chi_\alpha, \qquad \text{so that} \qquad \widehat{f^{(r)}} = \hat{f} - D_r.$$

Let $m^{(r)} = |\operatorname{supp}(\widehat{f^{(r)}})|$ be the number of coefficients left in the residual. Then $m^{(0)} \le k$, and the algorithm returns $\hat{f}$ exactly when $m^{(R)} = 0$. Throughout, we write

$$d^{(r)} = k/2^r \qquad \text{and} \qquad t^{(r)} = k/4^r.$$

Here $d^{(r)}$ controls the size of the subspace in round r, and $t^{(r)}$ is the *target*: the number of coefficients we want the residual to have at most at the start of round r. We say the residual is *on schedule* in round r if $m^{(r)} \le t^{(r)}$. Our goal is to show that, with probability at least $2/3$, the residual is on schedule in every round.

Step 1: most coefficients are isolated. In round r, the algorithm samples a uniformly random subspace $H \le \mathbb{F}_2^n$ of dimension $\lceil \log(100\, d^{(r)}) \rceil$. This partitions the Fourier domain into $|W| = 2^{\dim H}$ buckets, one for each coset of $H^\perp$, and

$$100\, d^{(r)} \;\le\; |W| \;<\; 200\, d^{(r)}. \tag{1}$$

We say that $\alpha \in \operatorname{supp}(\widehat{f^{(r)}})$ is *isolated* if no other element of $\operatorname{supp}(\widehat{f^{(r)}})$ lies in its bucket.

Lemma 2. *For every $\alpha^* \in \operatorname{supp}(\widehat{f^{(r)}})$,*

$$\Pr_H\big[\alpha^* \textit{ is not isolated}\big] \;\le\; \frac{m^{(r)}}{100\, d^{(r)}}.$$

Proof. The subspace H is sampled independently of D_r, so $f^{(r)}$ is fixed when H is drawn. Two distinct frequencies α and α^* share a bucket exactly when $\alpha - \alpha^* \in H^\perp$. Since H is uniform, so is $H^\perp$: it is a uniformly random subspace of dimension $n - \dim H$. As $\alpha - \alpha^* \neq 0$, the calculation in the proof of Lemma 1 gives

$$\Pr_H\big[\alpha - \alpha^* \in H^\perp\big] \;\le\; 2^{-\dim H} \;\le\; \frac{1}{100\, d^{(r)}}.$$

There are $m^{(r)} - 1 < m^{(r)}$ choices of $\alpha \in \operatorname{supp}(\widehat{f^{(r)}}) \setminus \{\alpha^*\}$, and a union bound gives the claim. □

Let $X^{(r)}$ denote the number of coefficients of the residual that are *not* isolated. Summing Lemma 2 over all $m^{(r)}$ coefficients gives

$$\mathbb{E}\big[X^{(r)} \mid D_r\big] \;\le\; \frac{\big(m^{(r)}\big)^2}{100\, d^{(r)}}. \tag{2}$$

So collisions are rare as long as the residual is small compared to the number of buckets.

Step 2: each round makes progress. To see what one round does, we need the bucket values of the residual. Applying the restriction formula of Section 3 to $f^{(r)}$ gives, for every shift $c \in \mathbb{F}_2^n$ and every $\gamma \in W$,

$$\widehat{f^{(r)}_{c+H}}(\gamma) \;=\; \sum_{\alpha \in (\gamma + H^\perp) \cap \operatorname{supp}(\widehat{f^{(r)}})} \widehat{f^{(r)}}(\alpha)\, \chi_\alpha(c). \tag{3}$$

In words, the value of bucket γ at shift c is the sum of the residual coefficients in that bucket, each multiplied by its sign at c.

The next lemma is the heart of the analysis. A round removes every isolated coefficient outright. Each bucket with a collision produces at most one output, so it adds at most one wrong coefficient. Since such a bucket contains at least two coefficients, the damage is at most half a wrong coefficient per collided coefficient. This is where the factor $\frac{3}{2}$ comes from.

Algorithm 2: Efficient Sparse Fourier Recovery

Input: Query access to $f : \mathbb{F}_2^n \to \mathbb{R}$, sparsity k
Output: All pairs $(\alpha, \hat{f}(\alpha))$ for $\alpha \in \mathrm{supp}(\hat{f})$
$D \leftarrow \emptyset$ $R \leftarrow \lfloor \log_4 k \rfloor + 1$;
for $r = 0, 1, \ldots, R-1$ **do**
  $f^{(r)} \leftarrow f - \sum_{\alpha \in D} D(\alpha)\, \chi_\alpha$ $d^{(r)} \leftarrow k/2^r$;
  Sample $H \leq \mathbb{F}_2^n$ uniformly with $\dim(H) = \lceil \log(100\, d^{(r)}) \rceil$;
  Compute $\widehat{f^{(r)}_{\mathbf{c}+H}}(\gamma)$ for all $\gamma \in W$ and $\mathbf{c} \in \{0, \mathbf{e}_1, \ldots, \mathbf{e}_n\}$;
  $U \leftarrow \emptyset$;
  for $\gamma \in W$ with $\widehat{f^{(r)}_H}(\gamma) \neq 0$ **do**
    $\alpha_i \leftarrow \mathbf{1}\big[\widehat{f^{(r)}_{\mathbf{e}_i+H}}(\gamma) \cdot \widehat{f^{(r)}_H}(\gamma) < 0\big]$ for all $i \in [n]$;
    Add $\big((\alpha_1, \ldots, \alpha_n), \widehat{f^{(r)}_H}(\gamma)\big)$ to U;
  end
  for $(\alpha, v) \in U$ **do**
    $D(\alpha) \leftarrow D(\alpha) + v$, and delete α from D if $D(\alpha) = 0$;
  end
end
return D;

Lemma 3. *For every round r and every choice of H,*

$$m^{(r+1)} \;\leq\; \tfrac{3}{2}\, X^{(r)}.$$

Proof. We classify the buckets $\gamma \in W$ by how many coefficients of the residual they contain, and use equation 3 in each case.

Empty buckets. Here $\widehat{f^{(r)}_H}(\gamma) = 0$, so the bucket is skipped and produces no output.

Singleton buckets. Suppose the bucket contains exactly one element β. Then $\widehat{f^{(r)}_{c+H}}(\gamma) = \widehat{f^{(r)}}(\beta)\, \chi_\beta(c)$ for every shift c. In particular, $\widehat{f^{(r)}_H}(\gamma) = \widehat{f^{(r)}}(\beta) \neq 0$, so the bucket is decoded. For every $i \in [n]$,

$$\widehat{f^{(r)}_{e_i+H}}(\gamma)\; \widehat{f^{(r)}_H}(\gamma) \;=\; (-1)^{\beta_i}\, \widehat{f^{(r)}}(\beta)^2,$$

which is negative exactly when $\beta_i = 1$. So the decoded frequency is β, and the bucket outputs exactly $\big(\beta, \widehat{f^{(r)}}(\beta)\big)$.

Buckets with collisions. Suppose the bucket contains at least two elements. Then it produces at most one output, whose frequency and value may be arbitrary.

Let U_1 and U_2 be the sets of outputs of singleton buckets and of buckets with collisions, respectively. Distinct singleton buckets contain distinct elements, so they output distinct frequencies. Since every output is added to the dictionary,

$$\widehat{f^{(r+1)}} \;=\; \hat{f} - D_{r+1} \;=\; \widehat{f^{(r)}} \;-\; \sum_{(\alpha,v)\in U_1} v\, \mathbf{1}_\alpha \;-\; \sum_{(\alpha,v)\in U_2} v\, \mathbf{1}_\alpha,$$

where $\mathbf{1}_\alpha$ is the indicator function of $\{\alpha\}$. Deleting keys whose value becomes 0 does not change D_{r+1} as a function, so it does not affect this identity.

We bound the support of the right-hand side in two steps. First, subtracting the outputs in U_1 sets $\widehat{f^{(r)}}$ to zero at every isolated element and leaves it unchanged elsewhere, so the result is supported on the $X^{(r)}$ non-isolated elements. Second, subtracting the outputs in U_2 changes at most $|U_2|$ further coordinates. This holds even if an output in U_2 lands on a frequency already handled in U_1, or coincides with another output in U_2. Every bucket with a collision contains at least two non-isolated elements, and distinct buckets are disjoint, so $|U_2| \le X^{(r)}/2$. Therefore

$$m^{(r+1)} \;\le\; X^{(r)} + \tfrac{1}{2}X^{(r)} \;=\; \tfrac{3}{2}\,X^{(r)}. \qquad \square$$

Note that Lemma 3 involves no probability at all: it holds for every choice of H. All the randomness of the analysis enters through equation 2.

Step 3: the residual reaches zero. Lemma 3 reduces everything to one question: how many coefficients collide in each round? By equation 2, few do, provided the residual is small compared to the number of buckets. The schedule makes sure this remains true in every round.

The key is that the two quantities shrink at different rates. The target $t^{(r)} = k/4^r$ shrinks by a factor of 4 per round, while the number of buckets, about $100\,d^{(r)} = 100\,k/2^r$, shrinks only by a factor of 2. So the number of buckets per remaining coefficient *doubles* every round. Concretely, if the residual is on schedule in round r, then equation 2 gives

$$\mathbb{E}\big[X^{(r)} \mid D_r\big] \;\le\; \frac{\big(t^{(r)}\big)^2}{100\,d^{(r)}} \;=\; \frac{t^{(r)}}{100\cdot 2^r}.$$

In other words, the expected number of collided coefficients is a tiny fraction of the target, and this fraction halves every round. By Lemma 3, the residual stays on schedule in round $r+1$ as long as $X^{(r)} \le t^{(r)}/6$, since then $m^{(r+1)} \le \frac{3}{2}X^{(r)} \le t^{(r)}/4 = t^{(r+1)}$. Markov's inequality therefore bounds the probability of falling off schedule in round r by $6/(100\cdot 2^r) = 0.06/2^r$.

These failure probabilities form a geometric series, so the residual stays on schedule in every round except with probability less than 0.12. After $R = \lfloor \log_4 k \rfloor + 1$ rounds, the target $t^{(R)} = k/4^R$ is below 1. A residual on schedule then has no coefficients left, so the dictionary is exactly $\hat{f}$.

The same geometric decay also controls the cost. Round r queries f on $n+1$ cosets of a subspace of size less than $200\,d^{(r)}$. Since $\sum_r d^{(r)} \le 2k$, the total number of queries is less than $400\,k(n+1) = O(nk)$: the first round dominates, and all later rounds together cost no more than the first. The running time is dominated by one Walsh–Hadamard transform per coset, which gives $O(nk\log k)$ in total.

The appendix (Section A) provides the formal proof of the success probability and the complete query and running-time analysis, which together establish Theorem 1. It also provides some empirical insight into the performance of Algorithm 2 beyond the asymptotic analysis (Subsection B.1), and discusses possible extensions and open directions arising from our approach (Subsection B.2).

AI USE STATEMENT

We used generative AI tools solely for editorial assistance — improving grammar, spelling, and word choice in the writing of this paper. All theoretical results, including every theorem, lemma, and proof, were derived and verified by the authors. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This is a theoretical paper on exact Fourier-sparse recovery. Our work does not involve human subjects, data collection, or the release of models or datasets, and we identify no ethical concerns specific to it.

REPRODUCIBILITY STATEMENT

This is a theoretical paper, and all of our theoretical claims are supported by formal proofs. In Subsection B.1, we additionally discuss some concrete aspects of the algorithm, including its practical implementation and explicit constants. These aspects can be independently verified using the accompanying anonymized reference implementation, available at https://anonymous.4open.science/r/simple-sparse-recovery-64C2. Detailed instructions for reproducing these results are provided there.

REFERENCES

Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 459–467, 2012.

Adi Akavia, Shafi Goldwasser, and Shmuel Safra. Proving hard-core predicates using list decoding. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 146–157, 2003.

Andisheh Amrollahi, Amir Zandieh, Michael Kapralov, and Andreas Krause. Efficiently learning Fourier sparse set functions. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, 2019.

Thomas Blumensath and Mike E. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14(5–6):629–654, 2008.

Jean Bourgain. An improved estimate in the restricted isometry problem. In *Geometric Aspects of Functional Analysis*, pp. 65–70. Springer, 2014.

Emmanuel J. Candès and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

Emmanuel J. Candès and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

Emmanuel J. Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

Tullio Ceccherini-Silberstein, Fabio Scarabotti, and Filippo Tolli. *Discrete Harmonic Analysis: Representations, Number Theory, Expanders, and the Fourier Transform*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2018.

Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*, pp. 693–703. Springer, 2002.

Mahdi Cheraghchi, Venkatesan Guruswami, and Ameya Velingker. Restricted isometry of Fourier matrices and list decodability of random linear codes. *SIAM Journal on Computing*, 42(5): 1888–1914, 2013.

Ernesto De Vito, Lorenzo Rosasco, Andrea Caponnetto, Umberto De Giovannini, and Francesca Odone. Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6: 883–904, 2005.

David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. New results for learning noisy parities and halfspaces. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 563–574, 2006.

Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing (STOC)*, pp. 25–32, 1989.

Parikshit Gopalan, Ryan O'Donnell, Rocco A. Servedio, Amir Shpilka, and Karl Wimmer. Testing Fourier dimensionality and sparsity. *SIAM Journal on Computing*, 40(4):1075–1100, 2011.

Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price. Nearly optimal sparse Fourier transform. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing (STOC)*, pp. 563–578, 2012a.

Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price. Simple and practical algorithm for sparse Fourier transform. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1183–1194, 2012b.

Ishay Haviv and Oded Regev. The List-Decoding Size of Fourier-Sparse Boolean Functions. In *Proceedings of the 30th Conference on Computational Complexity (CCC 2015)*, pp. 58–71, 2015.

Ishay Haviv and Oded Regev. The restricted isometry property of subsampled Fourier matrices. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 288–297, 2016.

Piotr Indyk and Michael Kapralov. Sample-optimal Fourier sampling in any constant dimension. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 514–523, 2014.

Piotr Indyk, Michael Kapralov, and Eric Price. (Nearly) Sample-Optimal Sparse Fourier Transform. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 480–499, 2014.

Mark A. Iwen. Combinatorial sublinear-time Fourier algorithms. *Foundations of Computational Mathematics*, 10(3):303–338, 2010.

Michael Kapralov. Sparse Fourier transform in any constant dimension with nearly-optimal sample complexity in sublinear time. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 264–277, 2016.

Michael Kapralov. Sample efficient estimation and recovery in sparse FFT via isolation on average. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 651–662, 2017.

David R. Karger. Random sampling in cut, flow, and network design problems. *Mathematics of Operations Research*, 24(2):383–413, 1999.

Murat Kocaoglu, Karthikeyan Shanmugam, Alexandros G. Dimakis, and Adam R. Klivans. Sparse Polynomial Learning and Graph Sketching. In *Advances in Neural Information Processing Systems 27*, pp. 3122–3130, 2014.

Dmitry Kogan and Robert Krauthgamer. Sketching cuts in graphs and hypergraphs. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science (ITCS)*, pp. 367–376, 2015.

Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the Fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

Yi Li and Vasileios Nakos. Deterministic sparse Fourier transform with an ℓ_∞ guarantee. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP)*, pp. 77:1–77:14, 2020.

Yishay Mansour. Randomized interpolation and approximation of sparse polynomials. In *Proceedings of the 19th International Colloquium on Automata, Languages and Programming (ICALP)*, pp. 261–272, 1992.

Deanna Needell and Joel A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

Deanna Needell and Roman Vershynin. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Foundations of Computational Mathematics*, 9(3): 317–334, 2009.

Deanna Needell and Roman Vershynin. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):310–316, 2010.

Sahand Negahban and Devavrat Shah. Learning sparse Boolean polynomials. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 2032–2036, 2012.

Sameer Pawar and Kannan Ramchandran. A robust R-FFAST framework for computing a k-sparse n-length DFT in $o(k \log n)$ sample complexity using sparse-graph codes. In *2014 IEEE International Symposium on Information Theory (ISIT)*, pp. 1852–1856, 2014.

Elias M. Stein and Rami Shakarchi. *Fourier Analysis: An Introduction*. Princeton University Press, 2003.

Peter Stobbe and Andreas Krause. Learning Fourier Sparse Set Functions. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, PMLR*, volume 22, pp. 1125–1133, 2012.

Appendix

A Missing Parts of Proof of Theorem 1

Success Probability. Let E_r be the event that $m^{(r)} \le t^{(r)} = k/4^r$. Since $m^{(0)} \le k$, the event E_0 always holds. Fix a round r and condition on any value of D_r for which E_r holds. By Lemma 3, if E_{r+1} fails then $X^{(r)} > \frac{2}{3}\, t^{(r+1)} = t^{(r)}/6$. By Markov's inequality and equation 2,

$$\Pr\big[\neg E_{r+1} \,\big|\, D_r\big] \;\le\; \frac{6\,\mathbb{E}[X^{(r)} \mid D_r]}{t^{(r)}} \;\le\; \frac{6\,\big(t^{(r)}\big)^2}{100\, d^{(r)}\, t^{(r)}} \;=\; \frac{6}{100}\cdot\frac{k/4^r}{k/2^r} \;=\; \frac{0.06}{2^r}.$$

Hence $\Pr[E_r \wedge \neg E_{r+1}] \le 0.06/2^r$. If E_R fails, then, since E_0 holds, there is a first round $r < R$ with E_r and not E_{r+1}. Therefore

$$\Pr[\neg E_R] \;\le\; \sum_{r=0}^{R-1} \frac{0.06}{2^r} \;<\; 0.12 \;<\; \frac{1}{3}.$$

On E_R we have $m^{(R)} \le k/4^R < 1$, because $4^R = 4^{\lfloor \log_4 k \rfloor + 1} > k$. Hence $m^{(R)} = 0$, that is, $\hat{f} = D_R$, and Algorithm 2 returns all nonzero Fourier coefficients of f exactly, with probability at least $2/3$.

Query Complexity. In round r, the algorithm evaluates $\widehat{f^{(r)}_{c+H}}(\cdot)$ for the $n+1$ shifts $c \in \{0, e_1, \ldots, e_n\}$. Each evaluation queries f on the coset $c+H$, that is, on $|H| = |W| < 200\, d^{(r)}$ points by equation 1. Since $\sum_{r\geq 0} d^{(r)} = \sum_{r\geq 0} k/2^r \leq 2k$, the total number of queries is

$$Q(n,k) \;\leq\; (n+1)\cdot 200 \sum_r d^{(r)} \;\leq\; 400\, k\,(n+1) \;=\; O(nk).$$

Time Complexity. For the running time, fix a basis $h_1, \ldots, h_{\dim H}$ of H; the map $\alpha \mapsto (\langle \alpha, h_1\rangle, \ldots, \langle \alpha, h_{\dim H}\rangle)$ identifies the buckets with $\mathbb{F}_2^{\dim H}$, and the bucket of any α is computed in time $O(\log k)$. We work in the word-RAM model with words of $\Theta(n)$ bits and exact arithmetic on the values returned by the oracle. In round r:

1. Each of the $n+1$ evaluations enumerates $c+H$ in Gray-code order in $O(1)$ time per point and applies a Walsh–Hadamard transform over H, at cost $O(|H|\log|H|) = O(d^{(r)}\log k)$ after the queries to f. Over all rounds this is $O\big(n\log k \sum_r d^{(r)}\big) = O(nk\log k)$.
2. To pass from f to the residual $f^{(r)}$, each $\alpha \in D_r$ contributes $D_r(\alpha)\chi_\alpha(c)$ to its bucket for each of the $n+1$ shifts c. The bucket costs $O(\log k)$ and, since $\chi_\alpha(e_i) = (-1)^{\alpha_i}$, each contribution costs $O(1)$, so this step costs $O(|D_r|\,(n+\log k))$ per round.
3. Decoding reads n signs for each of at most $|W|$ buckets, at cost $O(n\,|W|)$, and the at most $|W|$ dictionary updates cost $O(|W|\log k)$ using a balanced search tree. Over all rounds this is $O(nk + k\log k)$ by equation 1.

Each round adds at most $|W| < 200\, d^{(r)}$ keys, so $|D_r| < 400k$ throughout, and step 2 costs $O(k(n+\log k))$ per round. Summing over the $R = O(\log k)$ rounds, the total running time is

$$\mathrm{Time}(n,k) = O(nk\log k).$$

All subspaces depend only on the internal randomness of the algorithm, so all queries can be fixed in advance and the algorithm is non-adaptive.

B Discussion

B.1 Explicit Complexity Analysis of Algorithm 2

Let c denote the number of buckets per residual coefficient, so that round r uses a subspace of dimension $\lceil \log_2(ck/2^r)\rceil$; our analysis takes $c = 100$. Both costs grow linearly in c: the algorithm makes fewer than $4ck(n+1)$ queries, and its running time is dominated by fewer than $4ck(n+1)(\log_2(2ck)-1)$ additions in the Walsh–Hadamard transforms. The failure probability, in turn, is at most $12/c$, so any $c > 36$ gives a success probability of at least $2/3$.

Our reference implementation[1] suggests that the constant can be substantially smaller in practice. With $c = 3$ buckets per coefficient, Algorithm 2 succeeded in $9/10$ fraction of the trials in our experiments. This gap between the constant used in our analysis and the values observed empirically suggests that there may be room for a sharper analysis of the bucket constant, which we leave as an interesting direction for future work.

B.2 Extensions

Extending to Approximate Sparsity. Our analysis assumes a noiseless, exactly k-sparse function. Even so, the framework seems like a natural place to start when studying more robust recovery settings. Each bit of a frequency is read from the sign of a product of two bucket values, and this decoding is stable under small perturbations relative to those bucket values. For larger perturbations, one option is to take a majority vote over $O(\log n + \log k)$ random shifts, which may add robustness at only a logarithmic cost. Handling approximately sparse functions would also require controlling the Fourier tail within each bucket. Here the ℓ_2/ℓ_∞ tail estimates from the filter-based literature

[1] https://anonymous.4open.science/r/simple-sparse-recovery-64C2/README.md

(Amrollahi et al., 2019) may be a useful tool. Because our random restriction hashes in the input domain rather than through an explicit spectral filter, a natural question is whether analogous tail bounds can be established in our setting, and whether they lead to simpler analyses.

Beyond $\mathbb{F}_2^n$. It is also worth asking whether the ideas extend beyond functions on $\mathbb{F}_2^n$ to functions of the form $[N] \to \mathbb{C}$, with $N = 2^n$ as a natural special case. Our central observation, that random restrictions can implicitly partition the Fourier spectrum, may have analogues in other settings. A further step would be to handle higher-dimensional domains such as $[n]^d$, which would bring the framework closer to the sparse Fourier transform setting of Kapralov (2017); Li & Nakos (2020). Finally, one could ask whether similar techniques carry over to sparse recovery over general finite abelian groups or other structured transform domains.