Approximating Sumset Size

Anindya De
University of Pennsylvania

Shivam Nadimpalli
Columbia University

Rocco A. Servedio
Columbia University

Abstract

Given a subset A of the n-dimensional Boolean hypercube $\mathbb{F}_2^n$, the *sumset* $A+A$ is the set $\{a+a' : a, a' \in A\}$ where addition is in $\mathbb{F}_2^n$. Sumsets play an important role in additive combinatorics, where they feature in many central results of the field.

The main result of this paper is a sublinear-time algorithm for the problem of *sumset size estimation*. In more detail, our algorithm is given oracle access to (the indicator function of) an arbitrary $A \subseteq \mathbb{F}_2^n$ and an accuracy parameter $\varepsilon > 0$, and with high probability it outputs a value $0 \leq v \leq 1$ that is $\pm\varepsilon$-close to $\mathrm{Vol}(A' + A')$ for some perturbation $A' \subseteq A$ of A satisfying $\mathrm{Vol}(A \setminus A') \leq \varepsilon$. It is easy to see that without the relaxation of dealing with A' rather than A, any algorithm for estimating $\mathrm{Vol}(A + A)$ to any nontrivial accuracy must make $2^{\Omega(n)}$ queries. In contrast, we give an algorithm whose query complexity depends only on ε and is completely independent of the ambient dimension n.

1 Introduction

Recent decades have witnessed a paradigm shift in the notion of what constitutes an "efficient algorithm" in algorithms and complexity theory. Motivated both by practical applications and theoretical considerations, the traditional gold standard of linear time as the ultimate benchmark for algorithmic efficiency has given way to the notion of *sublinear-time* and *sublinear-query* algorithms, as introduced by Blum and Kannan [BK89] and Blum, Luby and Rubinfeld [BLR93]. The study of sublinear algorithms is flourishing, with deep connections to many other areas including PCPs, hardness of approximation, and streaming algorithms (see e.g. the surveys [Rub06, Gol17, Fis01, Ron01]).

The current paper is at the confluence of two different lines of research in the area of sublinear algorithms:

1. The first strand of work deals with sublinear algorithms to *approximately compute (numerical-valued) functions on various combinatorial objects*. Example problems of this sort include (i) estimating the weight of a minimum spanning tree [CRT05]; (ii) approximating the minimum vertex cover size in a graph [PR07]; and (iii) approximating the number of k-cliques in an undirected graph [ERS18]. We note that for the first two of these results, the number of local queries that are made to the input combinatorial object is completely independent of its size.

2. The second strand of work is on *property testing of Boolean-valued functions*. Given a class of Boolean-valued functions $\mathcal{C}$, a testing algorithm for $\mathcal{C}$ is a query-efficient procedure which, given oracle access to an arbitrary Boolean-valued function f, distinguishes between the two cases that (i) f belongs to class $\mathcal{C}$, versus (ii) f is ε-far from every function in $\mathcal{C}$. Flagship results in this area include algorithms for linearity testing [BLR93], testing of low-degree polynomials [RS96, JPRZ04], junta testing [FKR$^+$04, Bla09], and monotonicity testing [GGL$^+$00, KMS18]. Here too, for the first three of these properties, the query complexity of the testing algorithms depend only on the accuracy parameter ε and are completely independent of the ambient dimension n of the function f.

In recent years, a nascent line of work has emerged at the intersection of these two strands, where the high-level goal is to approximately compute various numerical parameters of Boolean-valued functions. As an example, building on the work of Kothari et al. [KNOW14], Neeman [Nee14] gave an algorithm to approximate the "surface area" of a Boolean-valued function on $\mathbb{R}^n$, which is a fundamental measure of its complexity [KOS08]. The [Nee14] algorithm has a query complexity of $\mathrm{poly}(S)$ if the target surface area is S, which is completely independent of the ambient dimension n. Fitting the same motif is the work of Ron et al. [RRS$^+$12] who studied

the problem of approximating the "total influence" (or equivalently, "average sensitivity") of a Boolean function. They showed that the optimal query complexity to approximate the influence $\mathbf{Inf}[f]$ of an arbitrary n-variable Boolean function f to constant relative error is $\Theta(n/\mathbf{Inf}[f])$, and that this can be strengthened to essentially $\sqrt{n}/\mathbf{Inf}[f]$ for monotone functions. More recently, in closely related work Rubinfeld and Vasiliyan [RV19] have given a constant-query algorithm to approximate the "noise sensitivity" of a Boolean function.

We note that each of the above three numerical parameters — surface area, total influence, and noise sensitivity — is essentially a measure of the "smoothness" of the Boolean function in question. In contrast, in this work we are interested in the *sumset size*, which has a rather different flavor and, as discussed below, is intimately connected to the subspace structure of the function.

Sumsets. Let $A \subseteq \mathbb{F}_2^n$ be an arbitrary subset (which may of course be viewed as a Boolean function by considering its $\{0,1\}$-valued characteristic function). One of the most fundamental operations on such a set A is to consider the *sumset* $A + A$, defined as

$$A + A := \{x + y : x, y \in A\}.$$

Here '+' is the group operation in $\mathbb{F}_2^n$. Note that for A an affine subspace we have that $|A + A| = |A|$, and the converse (the only sets A for which $|A + A| = |A|$ are affine subspaces) is also easily seen to hold. In fact, something significantly stronger is true: The celebrated Freiman–Ruzsa theorem [Fre73, Ruz99, San12] states that if $|A + A| \leq K \cdot |A|$, then A is contained inside an affine subspace H such that $|H| \leq O_K(1) \cdot |A|$. Thus, the value of $|A + A|$ *vis-a-vis* $|A|$ can be seen as a measure the "subspace structure" of A.

1.1 The Question We Consider For $A \subseteq \mathbb{F}_2^n$, we define $\mathrm{Vol}(A) := |A|/2^n \in [0,1]$ to be the *normalized size* or *volume* of A. This paper is motivated by the following basic algorithmic problem about sumsets:

> **Sumset size estimation (naive formulation):** Given black-box oracle access to a set $A \subseteq \mathbb{F}_2^n$ (via its characteristic function $A : \mathbb{F}_2^n \to \{0,1\}$), can we estimate the $\mathrm{Vol}(A + A)$ while making only "few" oracle calls to A?

At first glance this seems to be a difficult problem, since to confirm that a given point z does not belong to $A + A$ we must verify that at least one of $x, y \notin A$ for each of the 2^n pairs (x, y) satisfying $x + y = z$. Indeed, for the above naive problem formulation, any algorithm must make $2^{\Omega(n)}$ queries even to distinguish between the two extreme cases that $\mathrm{Vol}(A + A) = 0$ (i.e. $A = \emptyset$) versus $\mathrm{Vol}(A + A) = 1 - \exp(-\Theta(n))$. To see this, suppose that A is a uniform random subset of $2^{0.51n}$ many elements from $\mathbb{F}_2^n$. It is clear that any algorithm will need $\Omega(2^{0.49n})$ queries to distinguish such an A from the empty set, and an easy calculation shows that such a random A will with extremely high probability have $\mathrm{Vol}(A + A) = 1 - \exp(-\Theta(n))$.

This simple example already shows that some care must be taken to formulate the "right" version of the sumset size estimation problem. This situation is analogous to the surface area testing problem that was studied in [KNOW14, Nee14]: In that setting, given oracle access to any set A, by adding a measure zero set R to A (which is undetectable by an algorithm with oracle access to A) it is possible to "blow up" the surface area of $A \cup R$ to an arbitrarily large value. Thus the goal in [KNOW14, Nee14] is to find a value S such that $\mathrm{surf}(A) \leq S \leq \mathrm{surf}(B)$ for a set B that is "close to A." Note that for surface area, it may be possible to dramatically increase the surface area of a set A either by adding a small subset of new points or removing a small subset of existing points from A. In contrast, for sumset size it is clear that removing points from A can never cause the sumset size to increase, and moreover adding a small (random) collection $R \subseteq \mathbb{F}_2^n$ of $2^{0.51n}$ points to A can always cause $\mathrm{Vol}((A \cup R) + (A \cup R))$ to become extremely close to 1. Hence for our sumset size estimation problem we only allow *subsets* of A as the permissible "close to A" sets.

We thus arrive at the following formulation of our problem:

> **Sumset size estimation:** Given black-box oracle access to a set $A \subseteq \mathbb{F}_2^n$ and an accuracy parameter $\varepsilon > 0$, compute $\mathrm{Vol}(A' + A')$ to additive accuracy $\pm\varepsilon$ for some subset $A' \subseteq A$ which has $\mathrm{Vol}(A \setminus A') \leq \varepsilon$.

1.2 Motivation Given the importance of sumsets in additive combinatorics, we feel that it is natural to investigate algorithmic questions dealing with basic properties of sumsets; estimating the size of a sumset is a natural algorithmic question of this sort. We further remark that while there is no direct technical connection to the present work, the path which led us to the sumset size estimation problem originated in an effort to

develop a query-efficient algorithm for *convexity testing* (i.e. testing whether a subset $S \subseteq \mathbb{R}^n$ is convex versus far from convex, where the standard Normal distribution $\mathcal{N}(0,1)^n$ provides the underlying distance measure on $\mathbb{R}^n$). In particular, the recent characterization by Shenfeld and van Handel of equality cases for the Ehrhard–Borell inequality (see Theorem 1.2 of [SvH18]) implies that a closed symmetric set $S \subseteq \mathbb{R}^n$ is convex if and only if the Gaussian volume of S equals the Gaussian volume of $\frac{S+S}{2}$. We believe that a robust version of this theorem might be useful for convexity testing; this naturally motivates a Gaussian space version of the sumset size estimation question, where now the Minskowski sum of sets in $\mathbb{R}^n$ plays the role of sumsets over $\mathbb{F}_2^n$. We hope that the ideas and ingredients in the current work may eventually be of use for the Gaussian space Minkowski sum size estimation problem, and perhaps ultimately for convexity testing.

1.3 Our Main Result Our main result is an algorithm for the subset size estimation problem which makes only *constantly* many queries, independent of the ambient dimension n. We state our main result informally below:

INFORMAL THEOREM 1. *Given oracle access to any set $A \subseteq \mathbb{F}_2^n$ and an error parameter $\varepsilon > 0$, there is an algorithm making $O_\varepsilon(1)$ queries to A with the following guarantee: with high probability, the algorithm outputs a value $0 \le v \le 1$ such that $\mathrm{Vol}(A'+A') - \varepsilon \le v \le \mathrm{Vol}(A'+A') + \varepsilon$ for some set $A' \subseteq A$ such that $\mathrm{Vol}(A \setminus A') \le \varepsilon$.*

In fact, as we describe in more detail later, our algorithm does more than just approximate the volume of $A'+A'$: it outputs a high-accuracy approximate oracle for the set $A'+A'$, given which it is trivially easy to approximate $\mathrm{Vol}(A'+A')$ by random sampling. (As we will see, our algorithm also outputs an exact oracle for the set A'.) Later we will give a formal definition of what it means to "output an oracle" for a set B; informally, it means we give a description of an oracle algorithm (which uses a black-box oracle to A) which, on any input x, (i) determines whether $x \in B$, and (ii) makes few invocations to the oracle for A. We further note that the running time of our algorithm is linear in n (note that even writing down an n-bit string as a query input to A takes linear time).

1.4 Technical Overview

1.4.1 A Conceptual Overview of the Algorithm In this subsection we give a technical overview of our algorithm. At a high level, our approach is based on the *structure versus randomness* paradigm that has proven to be very influential in additive combinatorics [TV06] and property testing. Our algorithm relies on two main ingredients, which we describe below.

To explain the key ingredients we need the notion of quasirandomness from additive combinatorics. For a set $A \subseteq \mathbb{F}_2^n$, we say A is *ε-quasirandom* if each non-empty Fourier coefficient $\widehat{A}(\alpha), 0^n \neq \alpha \in \mathbb{F}_2^n$, satisfies $|\widehat{A}(\alpha)| \le \varepsilon$, where we are viewing A as a characteristic function over the domain $\mathbb{F}_2^n$. The definition of the Fourier transform extends to the more general setting in which A is a characteristic function whose domain is some coset $x+H$ (of size 2^{n-k}) of $\mathbb{F}_2^n$. This is done by identifying H with $\mathbb{F}_2^k$ via a homomorphism; we give details later in Definition 2.

The first ingredient is the following: Let H be a linear subspace of $\mathbb{F}_2^n$, and let $B_x \subseteq x+H$, $B_y \subseteq y+H$ be subsets of cosets $x+H$ and $y+H$ respectively. Suppose that both $|B_x|/|x+H|$ and $|B_y|/|y+H|$ are at least τ, and that both B_x and B_y are ε-quasirandom (viewed as characteristic functions whose domains are the cosets $x+H$ and $y+H$ respectively). Our first ingredient is the simple but useful observation that if $\tau \gg \sqrt{\varepsilon}$, then the set B_x+B_y (which is easily seen to be a subset of the coset $x+y+H$) must be almost the entire coset $x+y+H$ (see Lemma 3.1).

The second ingredient is Green's well-known "regularity lemma" for Boolean functions [Gre05]. To explain this, for any set $A \subseteq \mathbb{F}_2^n$, subspace H of $\mathbb{F}_2^n$, and coset H', let $A_{H'} := A \cap H'$ be the intersection of A with the coset H'. Roughly speaking, Green's regularity lemma shows that for any $A \subseteq \mathbb{F}_2^n$, there is a subspace H of codimension at most $O_{\gamma,\varepsilon}(1)$ such that the following holds: With probability $1-\gamma$ over a uniform random choice of cosets $\{H_i\}$, the set A_{H_i} is ε-quasirandom (viewed as a subset of the coset H_i). Moreover, the proof of the regularity lemma gives an iterative procedure to identify H; very roughly speaking, until the procedure terminates, at each stage it identifies a vector $\alpha \in \mathbb{F}_2^n$ such that $|\widehat{A}(\alpha)|$ is large, and sets H to be the span of the vectors identified so far.

With these two ingredients in place, we are ready to explain (at least at a qualitative level; we defer discussion of how to achieve the desired $O(1)$ query complexity to the next subsection) the algorithm for simulating an oracle

to $A'+A'$. First, we run the algorithmic version of Green's regularity lemma; having done so, we have a subspace H and we know that for most cosets H', the set $A_{H'}$ is ε-quasirandom. Let k be the codimension of H and let $\mathcal{B}'$ be a set of 2^k many coset representatives for the 2^k cosets of H. Let $\mathcal{B} \subseteq \mathcal{B}'$ be the subset consisting of those coset representatives $y \in \mathcal{B}'$ for which the set A_{y+H} (i) is ε-quasirandom and (ii) has density at least τ when viewed as a subset of $y+H$ (where τ is some carefully chosen parameter that we do not specify here). We note that given any coset $y+H$, condition (ii) can be checked using simple random sampling. Condition (i) is equivalent to checking that the set A_{y+H} has no Fourier coefficient larger than ε. This can be done using the celebrated Goldreich-Levin algorithm [GL89].[1] Thus, at this point our algorithm has determined the set $\mathcal{B} \subseteq \mathcal{B}'$.

The set $A' \subset A$ is defined to be

$$A' := \bigcup_{y \in \mathcal{B}} A_{y+H},$$

i.e. A' is obtained from A by removing A_{y+H} for each $y \in \mathcal{B}' \setminus \mathcal{B}$, or equivalently, "zeroing out" A on every coset $y+H$ where A_{y+H} either is not ε-quasirandom or has density smaller than τ. (Since the algorithm knows H and $\mathcal{B}$, it is clear from this definition of A' that, as mentioned after the informal theorem statement given earlier, the algorithm can simulate an exact oracle for the set A'.) Turning to $A'+A'$, we have that

$$\begin{aligned} A'+A' &= \bigcup_{y,z \in \mathcal{B}} \big(A \cap (y+H)\big) + \big(A \cap (z+H)\big), \\ &\approx \bigcup_{y,z \in \mathcal{B}} A \cap (y+z+H), \end{aligned} \tag{1.1}$$

where the last line follows from Lemma 3.1 (that we informally stated as the first ingredient mentioned above). As above, since the algorithm knows H and $\mathcal{B}$, it is clear from that the algorithm can simulate an approximate oracle for $A'+A'$.

1.4.2 Achieving Constant Query Complexity The above description essentially gives the high level description of our algorithm, at least at a conceptual level. However, there is a significant caveat, which arises when we consider the query complexity of the algorithm. Our goal is to achieve query complexity $O_\varepsilon(1)$, but explicitly obtaining a description of the subspace H necessarily requires a number of queries that scales at least linearly in n; indeed, even explicitly describing a single vector in H requires $\Theta(n)$ bits of information (and thus this many queries). Similarly, obtaining an explicit description of even a single vector $y \in \mathcal{B}'$ would be prohibitively expensive using only constantly many queries. To circumvent these obstacles and achieve constant (rather than linear or worse) query complexity, we need to develop "implicit" versions of the procedures described above.

As an example, we recall that the standard Goldreich-Levin algorithm, given oracle access to any set $A \subseteq \mathbb{F}_2^n$, outputs a list of parity functions $\chi_{\alpha^{(1)}}, \chi_{\alpha^{(2)}}, \dots$ such that the Fourier coefficient $|\widehat{A}(\alpha^{(i)})|$ is "large" (roughly, at least ε) for each i. However, explicitly outputting the label $\alpha^{(i)}$ of even a single parity would require n bits of information. To avoid this, we slightly modify the standard Goldreich-Levin procedure to show that with $\mathrm{poly}(1/\varepsilon)$ queries, we can output *oracles* to the parity functions $\chi_{\alpha^{(1)}}, \chi_{\alpha^{(2)}}, \dots$. In turn, each such oracle can be computed on any point $x \in \mathbb{F}_2^n$ with just $\mathrm{poly}(1/\varepsilon)$ many queries to the set A; thus, we have *implicit* access to the parity functions $\{\chi_\alpha\}$ rather than explicit descriptions of the parities. In the language of coding theory, this amounts to an analysis showing that the Goldreich-Levin algorithm can be used to achieve constant-query "local list correction" of the Hadamard code. We view this as essentially folklore [Sud21]; it is implicit in a number of previous works [STV01, KS13], but the closest explicit statements we have been able to find in the literature essentially say that Goldreich-Levin is a constant-query local list decoder (rather than local list corrector) for the Hadamard code.

With an "implicit" version of the Goldreich-Levin algorithm in hand, we show how to carefully use this implicit Goldreich-Levin to obtain an "implicit" algorithmic version of Green's regularity lemma. This implicit version is sufficient to carry out the steps mentioned above with overall constant query complexity. We hope that the implicit (query-efficient) versions of these algorithms may be useful in other settings beyond the current work.

[1] To be more accurate, this requires a slight adaptation of the Goldreich-Levin algorithm because the domain here is a coset rather than the more familiar domain $\mathbb{F}_2^n$ for Goldreich-Levin.

1.5 Related Work As noted earlier, our sumset size estimation problem has a similar flavor to the work of [KNOW14, Nee14] on testing surface area, but the technical details are entirely different.

We note that for any invertible affine transformation $\Phi : \mathbb{F}_2^n \to \mathbb{F}_2^n$, we have that $\mathrm{Vol}(A+A) = \mathrm{Vol}(\Phi A + \Phi A)$ (but clearly this need not hold for noninvertible affine transformations). Starting with the influential paper of Kaufman and Sudan [KS08], a number of works have studied the testability of affine-invariant properties, see e.g. [BGS15, BFL13, HL13, Yos14, HHL16, Bha13] These works consider properties that are invariant under *all* affine transformations (not just invertible ones), which makes them inapplicable to our setting. However, we note that there are thematic similarities between the approaches in those works and our approach (in particular, the use of the "structure versus randomness" paradigm).

2 Preliminaries

In this section, we set notation and briefly recall preliminaries from additive combinatorics and Fourier analysis of Boolean functions. Given arbitrary $A, B \subseteq \mathbb{F}_2^n$, we define

$$\mathrm{Vol}(A) := \frac{|A|}{2^n} \qquad \text{and} \qquad \mathrm{Vol}_B(A) := \frac{|A \cap B|}{|B|}.$$

We will sometimes identify a set $A \subseteq \mathbb{F}_2^n$ with its indicator function $A : \mathbb{F}_2^n \to \{0,1\}$, defined as

$$A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

for $x \in \mathbb{F}_2^n$. When $A \subseteq x + H$ for some coset $x + H$, we similarly identify A with its indicator function $A : x + H \to \{0,1\}$. We write $e_i \in \mathbb{F}_2^n$ to denote the vector with a 1 in the i^{th} position and 0 everywhere else. The function $2 \uparrow\uparrow m$ denotes an exponential tower of 2's of height m and the function $\log^*$ denotes its inverse.

2.1 Analysis of Boolean Functions Our notation and terminology follow [O'D14]. We will view the vector space of functions $f : \mathbb{F}_2^n \to \mathbb{R}$ as a real inner product space, with inner product $\langle f, g \rangle := \mathbf{E}_{\boldsymbol{x} \sim \mathbb{F}_2^n}\big[f(\boldsymbol{x})g(\boldsymbol{x})\big]$. It is easy to see that the collection of *parity functions* $\{\chi_\alpha\}_{\alpha \in \mathbb{F}_2^n}$ where $\chi_\alpha(x) := (-1)^{\langle \alpha, x \rangle} = (-1)^{\sum_{i=1}^n \alpha_i x_i}$ forms an orthonormal basis for this vector space. In particular, every function $f : \mathbb{F}_2^n \to \mathbb{R}$ can be uniquely expressed by its *Fourier transform*, given by

$$f(x) = \sum_{\alpha \in \mathbb{F}_2^n} \widehat{f}(\alpha)\chi_\alpha(x). \tag{2.2}$$

The real number $\widehat{f}(\alpha)$ is called the *Fourier coefficient of f on α*, and the collection of all 2^n Fourier coefficients of f is called the *Fourier spectrum* of f. We recall Parseval's and Plancherel's formulas: for all $f, g : \mathbb{F}_2^n \to \mathbb{R}$, we have

$$\langle f, f \rangle = \sum_{\alpha \in \mathbb{F}_2^n} \widehat{f}(\alpha)^2 \qquad \text{and} \qquad \langle f, g \rangle = \sum_{\alpha \in \mathbb{F}_2^n} \widehat{f}(\alpha)\widehat{g}(\alpha). \tag{2.3}$$

It follows that $\mathbf{E}[f] = \widehat{f}(0)$. Given $f, g : \mathbb{F}_2^n \to \mathbb{R}$, their *convolution* is the function $f * g : \mathbb{F}_2^n \to \mathbb{R}$ defined by

$$f * g(x) := \mathop{\mathbf{E}}_{\boldsymbol{y} \sim \mathbb{F}_2^n}\big[f(\boldsymbol{y})g(x + \boldsymbol{y})\big],$$

which satisfies

$$\widehat{f * g}(\alpha) = \widehat{f}(\alpha) \cdot \widehat{g}(\alpha). \tag{2.4}$$

2.2 Subspaces and Functions on Subspaces Throughout this subsection, let $f : \mathbb{F}_2^n \to \mathbb{R}$ and let $H \leq \mathbb{F}_2^n$ be a linear subspace of codimension k (so $|H| = 2^{n-k}$). We can write

$$H = \big\{x : \langle x, \alpha_i \rangle = 0 \ \forall\ i \in \{1, \dots, k\}\big\} \tag{2.5}$$

for some linearly independent collection of vectors $\{\alpha_1, \ldots, \alpha_k\}$.

A coset H', which is an affine subspace or, equivalently, a "translate" $y + H$ for some $y \in \mathbb{F}_2^n$, can be expressed as a set of the form

$$H' = \left\{x : \langle x, \alpha_i \rangle = b_i \ \forall\ i \in \{1, \ldots, k\}\right\}$$

for some $b_i \in \mathbb{F}_2$; we will often identify H' with the vector $b := (b_1, \ldots, b_k)$. Note that if $H' = y + H$, then $b_i = \langle y, \alpha_i \rangle$.

Any coset of H is affinely isomorphic to a copy of $\mathbb{F}_2^{n-k}$, and this lets us define the Fourier transform of a function $f : \mathbb{F}_2^n \to \mathbb{R}$ restricted to a coset H'. More formally, consider the function $f_{H'} : H' \to \mathbb{R}$ defined as $f_{H'}(x) = f(x)$. Its Fourier spectrum is indexed by the 2^{n-k} elements of H; in particular, for each $\beta \in H$ we have

$$\widehat{f_{H'}}(\beta) = \frac{1}{2^{n-k}} \sum_{x \in H'} f(x)\chi_\beta(x). \tag{2.6}$$

We can alternatively restrict a function $f : \mathbb{F}_2^n \to \mathbb{R}$ to a coset H', but treat it as a function on $\mathbb{F}_2^n$ that takes value 0 on all points in $\mathbb{F}_2^n \setminus H'$; this viewpoint will be notationally cleaner to work with going forward so we elaborate on it here. We define the function $f\restriction_{H'} : \mathbb{F}_2^n \to \mathbb{R}$ as

$$f\restriction_{H'}(x) = \begin{cases} f(x) & x \in H' \\ 0 & \text{otherwise} \end{cases}. \tag{2.7}$$

The Fourier coefficients of $f_{H'}$ and $f\restriction_{H'}$ are related by the following simple fact.

FACT 2.1. *Let $f : \mathbb{F}_2^n \to \mathbb{R}$, H be as in Equation (2.5), and let H' be a coset of H. Let $\mathcal{B}' \subseteq \mathbb{F}_2^n, |\mathcal{B}'| = 2^k$ be a collection of 2^k coset representatives for H (so every vector in $\mathbb{F}_2^n$ has a unique representation as $\gamma + \beta$ for some $\gamma \in \mathcal{B}', \beta \in H$). For any $\gamma \in \mathcal{B}', \beta \in H$, we have*

$$\left|\widehat{f\restriction_{H'}}(\gamma + \beta)\right| = \frac{1}{2^k} \cdot \left|\widehat{f_{H'}}(\beta)\right|.$$

Proof. For ease of notation we first consider the case that $\alpha_i = e_i$. Suppose that H' is given by

$$H' = \left\{x : \langle x, e_i \rangle = b_i \ \forall\ i \in \{1, \ldots, k\}\right\}$$

where we write $b := (b_1, \ldots, b_k)$. We may take $\mathcal{B}'$ to be the set of all 2^k vectors in $\mathbb{F}_2^n$ whose last $n-k$ coordinates are all 0, and we note that $H = \operatorname{span}\{e_{k+1}, \ldots, e_n\}$.

For $\gamma \in \mathcal{B}', \beta \in H$, we have

$$\begin{aligned}
\widehat{f\restriction_{H'}}(\gamma + \beta) &= \frac{1}{2^n} \sum_{x \in \mathbb{F}_2^n} f\restriction_{H'}(x)\chi_{\gamma+\beta}(x) \\
&= \frac{1}{2^n} \sum_{x_1 \in \mathbb{F}_2^k} \sum_{x_2 \in \mathbb{F}_2^{n-k}} f\restriction_{H'}(x_1, x_2)\chi_\gamma(x_1)\chi_\beta(x_2)
\end{aligned}$$

where we have abused notation in the last line and viewed $\gamma \in \mathbb{F}_2^k, \beta \in \mathbb{F}_2^{n-k}$. In turn the above is equal to

$$= \frac{1}{2^n} \sum_{x_2 \in \mathbb{F}_2^{n-k}} f\restriction_{H'}(b, x_2)\chi_\gamma(b)\chi_\beta(x_2)$$

as $(x_1, x_2) \notin H'$ (and hence $f\restriction_{H'}(x_1, x_2) = 0$) if $x_1 \neq b$, and so

$$\begin{aligned}
&= \frac{\chi_\gamma(b)}{2^n} \sum_{x_2 \in \mathbb{F}_2^{n-k}} f\restriction_{H'}(b, x_2)\chi_\beta(x_2) \\
&= \frac{\chi_\gamma(b)}{2^k} \cdot \frac{1}{2^{n-k}} \sum_{x_2 \in \mathbb{F}_2^{n-k}} f_{H'}(b, x_2)\chi_\beta(x_2)
\end{aligned}$$

which by Equation (2.6) gives us

$$= \frac{\chi_\gamma(b)}{2^k} \cdot \widehat{f_{H'}}(\beta).$$

The result in the general case follows by applying an invertible linear transformation mapping $\alpha_i \mapsto e_i$ (see Exercise 3.1 of [O'D14]). □

2.3 Parity Decision Trees We will only need the notion of a "nonadaptive" parity decision tree:

DEFINITION 1. (NONADAPTIVE PARITY DECISION TREE) *A* nonadaptive parity decision tree $\mathcal{T}_f$ *is a representation of a function* $f : \mathbb{F}_2^n \to \mathbb{R}$. *It consists of a rooted binary tree of depth* d *with* 2^d *leaves, so every root-to-leaf path has length exactly* d. *Each internal node at depth* i *is is labeled by a vector* $\alpha_i \in \mathbb{F}_2^n$ *corresponding to the parity function* $\chi_{\alpha_i}(\cdot)$, *and the vectors* $\alpha_1, \ldots, \alpha_d \in \mathbb{F}_2^n$ *are linearly independent. (Having all nodes at level* i *be labeled with the same vector* α_i *is the sense in which the tree is "nonadaptive.") The outgoing edges of each internal node are labeled* 0 *and* 1, *and the leaves of* $\mathcal{T}_f$ *are labeled by functions (which are restrictions of* f*). The* size *of* $\mathcal{T}_f$ *is the number of leaf nodes of* $\mathcal{T}_f$.

In more detail, a root-to-leaf path can be written as $\{(\alpha_i \to b_i)\}$ *where we follow the outgoing edge* b_i *from the internal node* α_i, *with* $b_i \in \mathbb{F}_2$. *On an input* x, *the parity decision tree* $\mathcal{T}_f$ *follows the root-to-leaf path* $\{(\alpha_i \to \langle \alpha_i, x \rangle)\}$ *and outputs the value of the function associated to the leaf at* x.

Note that given $f : \mathbb{F}_2^n \to \mathbb{R}$ and a subspace $H \leq \mathbb{F}_2^n$ of codimension k as in Equation (2.5), we can associate a natural parity decision tree $\mathcal{T}_f$ in which each level-i internal node is labeled by α_i and each leaf node (corresponding to some coset H' of H) is labeled by $f\restriction_{H'}$.

2.4 Quasirandomness and Green's Regularity Lemma The following definition of *quasirandomnesss* has been well-studied as a notion of pseudorandomness in additive combinatorics; we refer the interested reader to [CG92] for more details.

DEFINITION 2. (ε-QUASIRANDOMNESS) *We say that* $f : \mathbb{F}_2^n \to \mathbb{R}$ *is* ε-quasirandom *if*

$$\sup_{0^n \neq \alpha} \left| \widehat{f}(\alpha) \right| \leq \varepsilon.$$

DEFINITION 3. (ε-QUASIRANDOM WHEN RESTRICTED TO COSET) *Let* $f : \mathbb{F}_2^n \to \mathbb{R}$, $H \leq \mathbb{F}_2^n$ *as in Equation (2.5), and let* H' *be a coset of* H. *We say that* $f_{H'} : H' \to \mathbb{R}$ *is* ε-quasirandom *if*

$$\sup_{0^n \neq \beta \in H} \left| \widehat{f_{H'}}(\beta) \right| \leq \varepsilon$$

where $\widehat{f_{H'}}(\beta)$ *is as defined in Equation (2.6).*

In Definitions 2 and 3, the function of interest will often be the indicator of a subset $A \subseteq \mathbb{F}_2^n$. We next state Green's regularity lemma for Boolean functions, which is analogous to Szemerédi's celebrated graph regularity lemma [Sze78].

PROPOSITION 2.1. (GREEN'S REGULARITY LEMMA IN $\mathbb{F}_2^n$) *Let* $A \subseteq \mathbb{F}_2^n$ *and* $\varepsilon, \gamma > 0$. *There exists a subspace* $H \leq \mathbb{F}_2^n$ *with cosets* $\{H_i\}$ *such that*

1. *the codimension of* H *is at most* $2 \uparrow\uparrow \frac{1}{\gamma \varepsilon^2}$; *and*
2. *for all but* γ*-fraction of cosets of* H, *the function* $A_{H_i} : H_i \to \{0, 1\}$ *is* ε*-quasirandom.*

In Section 3, we will closely follow the proof of Green's regularity lemma (in the course of providing a constructive albeit highly query-inefficient version of the lemma).

2.5 The Goldreich–Levin Theorem Given *query access* to a function $f : \mathbb{F}_2^n \to \{0,1\}$, the Goldreich–Levin algorithm [GL89] allows us to find all linear (parity) functions that are well-correlated with f (equivalently, it allows us to find all the "significant" Fourier coefficients of f). More formally, we have the following result.

PROPOSITION 2.2. (GOLDREICH–LEVIN ALGORITHM) *Let $A \subseteq \mathbb{F}_2^n$ be arbitrary and let $\theta, \delta > 0$ be fixed. There is an algorithm* GOLDREICH–LEVIN(A, θ, δ) *that, given query access to $A : \mathbb{F}_2^n \to \{0,1\}$, outputs a subset $\mathcal{S} \subseteq \mathbb{F}_2^n$ of size $O\big(1/\theta^2\big)$ such that with probability at least $1 - \delta$, we have*

- *if $\alpha \in \mathcal{S}$, then $\left|\widehat{A}(\alpha)\right| \geq \frac{\theta}{2}$; and*
- *if $\left|\widehat{A}(\alpha)\right| \geq \theta$, then $\alpha \in \mathcal{S}$.*

Furthermore, GOLDREICH–LEVIN(A, θ, δ) *runs in* $\mathrm{poly}\big(n, \frac{1}{\theta}, \log\frac{1}{\delta}\big)$ *time and makes* $\mathrm{poly}\big(n, \frac{1}{\theta}, \log\frac{1}{\delta}\big)$ *queries to A.*

2.6 Oracles and Oracle Machines As stated in the introduction, the outputs of our algorithmic procedures—Algorithms 3.1 and 3.2—will be *oracles* to the indicator functions of specific subsets of $\mathbb{F}_2^n$. We first recall the definition of a probabilistic oracle machine:

DEFINITION 4. *Let $f : \mathbb{F}_2^n \to \{0,1\}$. A randomized algorithm $\mathcal{O}$ with black-box query access to f, denoted $\mathcal{O}^f$, is said to be a* probabilistic oracle machine *for $g : \mathbb{F}_2^n \to \{0,1\}$ if for any input $x \in \mathbb{F}_2^n$, the algorithm $\mathcal{O}^f$ outputs a bit $\mathcal{O}^f(x)$ that satisfies*

$$\mathbf{Pr}[\mathcal{O}^f(x) = g(x)] \geq 2/3,$$

where the probability is taken over the internal coin tosses of $\mathcal{O}^f$. The query complexity *of the machine is the number of oracle calls made by $\mathcal{O}$ to f and the* running time *of the machine is the number of time steps it takes in the worst case (counting each oracle call as a single time step).*

Of course, the $2/3$ in the above definition can be upgraded to $1-\tau$ at a cost of increasing the query complexity by a factor of $O(\log(1/\tau))$. We next define what it means for an algorithm to "output an (approximate) oracle" for a function.

DEFINITION 5. *Let f, g be two functions $f, g : \mathbb{F}_2^n \to \{0,1\}$. An algorithm $\mathcal{A}$ with query access to f, denoted by $\mathcal{A}^f$, is said to* output a (δ, q, T)-oracle $\mathcal{O}_g^f$ for the function g *if it outputs a representation of a probabilistic oracle machine $\mathcal{O}_g^f$ for a function $h : \mathbb{F}_2^n \to \mathbb{R}$ for which the following hold:*

1. *We have* $\mathrm{dist}(h, g) \leq \delta$, *i.e.*
$$\mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \mathbb{F}_2^n}[h(\boldsymbol{x}) \neq g(\boldsymbol{x})] \leq \delta;$$
2. *The query complexity of $\mathcal{O}_g^f$ is at most q and the running time of $\mathcal{O}_g^f$ is at most T.*

If $\delta = 0$, then we say that $\mathcal{O}_g^f$ is an exact oracle *for g.*

3 A Query-Inefficient Version of the Main Result

In this section, we prove a query-inefficient "non-implicit" version of our main result, which has a polynomial query complexity dependence on the ambient dimension n. In particular, we will prove the following theorem.

THEOREM 3.1. (MAIN RESULT, QUERY-INEFFICIENT VERSION) *Let $A \subseteq \mathbb{F}_2^n$ be an arbitrary subset, and let $\varepsilon, \tau > 0$. Given query access to A, there exists an algorithm that makes* $\mathrm{poly}\big(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\big)$ *queries to A and does a* $\mathrm{poly}\big(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\big)$ *time computation and outputs with probability at least $9/10$:*

1. *A $\big(0, 1, O(n)\big)$-oracle $\mathcal{O}_{A'}^A$ to the indicator function of $A' \subseteq A$ where $\mathrm{Vol}\big(A \setminus A'\big) \leq \varepsilon + \tau$; and*
2. *A $\Big(O\big(\varepsilon^2/\tau^4\big), 0, O(n)\Big)$-oracle $\mathcal{O}_{A'+A'}^A$ to the indicator function of the sumset $A' + A'$.*

In Section 4, we will present an "implicit" version of Theorem 3.1 that makes only $O_\varepsilon(1)$ queries, independent of the ambient dimension n, and thereby prove our main result.

We start by recording a corollary of Green's regularity lemma in Section 3.1, which (informally), given an arbitrary set $A \subseteq \mathbb{F}_2^n$, establishes the existence of a "structured" set $A' \subseteq A$ capturing "almost all" of A. Section 3.2 then presents a procedure—ConstructDT—that constructs an *exact* oracle to this structured set A', giving item (1) of the above theorem. In Section 3.3, we present a procedure—Simulate-Sumset—that constructs an *approximate* oracle to the sumset $A' + A'$, giving item (2).

3.1 Partitioning Arbitrary Sets into Dense Quasirandom Cosets Green's regularity lemma in $\mathbb{F}_2^n$ says that given an arbitrary set $A \subseteq \mathbb{F}_2^n$ and an error parameter $\varepsilon > 0$, we can partition $\mathbb{F}_2^n$ into $O_\varepsilon(1)$ (independent of n) many sets such that A is "random-like" on almost all of these sets. Moreover, all these sets have a convenient structure: they are cosets of a common subspace of constant codimension.

We will use the following easy consequence of Green's lemma:

Proposition 3.1. *Given $A \subseteq \mathbb{F}_2^n$ and $\varepsilon, \tau > 0$, there exists a subspace $H \leq \mathbb{F}_2^n$ of codimension at most $2 \uparrow\uparrow \frac{1}{\varepsilon^3}$ and a set $A' \subseteq A$ such that*

1. $\mathrm{Vol}\big(A \setminus A'\big) \leq \varepsilon + \tau$;
2. *For any coset H_i, either $\mathrm{Vol}_{H_i}(A') = 0$ or $\mathrm{Vol}_{H_i}(A') \geq \tau$; and*
3. *A'_{H_i} is ε-quasirandom for all cosets H_i.*

Proof. Let $H \leq \mathbb{F}_2^n$ be the subspace of of codimension at most $2 \uparrow\uparrow \frac{1}{\varepsilon^3}$ guaranteed to exist by Proposition 2.1, and let $\{H_1, \ldots, H_M\}$ be an enumeration of the cosets of H where $M = 2^n \cdot |H|^{-1}$. We know from Proposition 2.1 that for all but ε-fraction of $\{H_i\}$, the function $A_{H_i} : H_i \to \{0,1\}$ is ε-quasirandom.

Define disjoint subsets $A'_1, \ldots, A'_M$, where each $A'_i \subseteq A \cap H_i$, as follows:

1. If A_{H_i} is not ε-quasirandom, then $A'_i = \emptyset$;
2. If $\mathrm{Vol}_{H_i}(A) \leq \tau$, set $A'_i = \emptyset$;
3. Otherwise, set $A'_i = A \cap H_i$.

We now define $A' \subseteq \mathbb{F}_2^n$ as

$$A' := \bigsqcup_{i=1}^{M} A'_i. \tag{3.8}$$

We clearly have $\mathrm{Vol}(A \setminus A') \leq \varepsilon + \tau$ and that A_{H_i} is ε-quasirandom for all $i \in [M]$. (Note that $\emptyset$ is trivially ε-quasirandom.) ∎

Informally, Proposition 3.1 modifies A to obtain a structured set $A' \subseteq A$ that contains "most" of A and has either empty or "large" intersection with all of the cosets guaranteed to exists by Green's regularity lemma. Furthermore, A' is "random-like" on *all*—as opposed to *almost all*—of these cosets.

3.2 A Constructive Regularity Lemma via the Goldreich–Levin Theorem In this section, we make Proposition 3.1 constructive via the Goldreich–Levin algorithm. The procedure ConstructDT presented in Algorithm 3.1 closely follows the structure of Green's original proof of the regularity lemma itself [Gre05].

ALGORITHM 3.1.

Input: Query access to $A \subseteq \mathbb{F}_2^n$, quasirandomness parameter ε, density threshold τ

Output: An exact oracle $\mathcal{O}^A_{A'}$ and a parity decision tree $\mathcal{T}_{\text{regular}}$ with A' as in Proposition 3.1

CONSTRUCTDT(A, ε, τ):

1. Initialize the decision tree $\mathcal{T}_{\text{regular}}$ to contain no internal nodes and one leaf labelled by $A : \mathbb{F}_2^n \to \{0,1\}$. Define
$$\delta := \left(2 \uparrow\uparrow \frac{8}{\varepsilon^3}\right)^{-1} \cdot \frac{1}{30}$$
2. At each stage of growing $\mathcal{T}_{\text{regular}}$, do the following:
   (a) Let $\{H_1, \dots, H_M\}$ denote the cosets corresponding to the leaves of the decision tree at the current stage. The i^{th} leaf node is labelled by the function A_{H_i}.
   (b) For each coset H_i, call
$$\mathcal{S}_i \leftarrow \text{GOLDREICH–LEVIN}\big(A\restriction_{H_i}, \varepsilon/M, \delta\big).$$
   (c) For each non-empty $\mathcal{S}_i$, for each $\alpha \in \mathcal{S}_i$, estimate $\left|\widehat{A_{H_i}}(\alpha)\right|$ up to additive error $\pm\varepsilon/4$ with confidence δ. If the estimate is less than $3\varepsilon/4$, then remove α from $\mathcal{S}_i$.
   (d) If $\mathcal{S}_i = \emptyset$ for at least $(1-\varepsilon)$-fraction of the $\{\mathcal{S}_1, \dots, \mathcal{S}_M\}$, go to Step 3.
   (e) Let the collection of labels of all internal nodes be $\mathcal{L}$. For each non-empty S_i:
      i. Choose $\alpha \leftarrow \mathcal{S}_i$. Check if the collection $\mathcal{L} \cup \{\alpha\}$ is linearly independent.
      ii. If so, then add α to $\mathcal{L}$ and split all nodes at the current stage on α.†
   (f) Repeat Step 2.
3. For each leaf node—say, corresponding to the coset H_i—estimate $\widehat{\Theta}_i := \text{Vol}_{H_i}(A)$ up to an additive error of $\pm\tau/4$ with confidence δ.
   (a) If $\widehat{\Theta}_i \geq 3\tau/4$, set the function associated to the leaf node to be the identically-1 function.
   (b) Else set it to be the identically-0 function.
4. Define the oracle $\mathcal{O}^A_{A'}$ to be the function
$$\mathcal{O}^A_{A'}(x) = \mathcal{T}_{\text{regular}}(x) \cdot A(x).$$

† By "splitting" a leaf node on a parity $\alpha \in \mathbb{F}_2^n$, we mean replacing it with an internal node labeled by the parity α with two natural leaf nodes as children.

PROPOSITION 3.2. *Let $A \subseteq \mathbb{F}_2^n$ be an arbitrary subset. Given query access to A and $\varepsilon, \tau > 0$, the procedure* CONSTRUCTDT(A, ε, τ) *described in Algorithm 3.1:*

1. *Makes* $\text{poly}\big(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\big)$ *queries to A and does a* $\text{poly}\big(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\big)$ *time computation; and*

2. *With probability* 9/10 *outputs a deterministic* $(0, 1, O(n))$*-oracle* $\mathcal{O}^A_{A'}$ *for* A' *where* $A' \subseteq A$ *is as in Proposition 3.1.*

We note that the procedure ConstructDT makes queries to the oracle A in the course of running the Goldreich–Levin algorithm.

Proof. We first argue that Step 2 in the procedure ConstructDT terminates; this essentially follows from Green's original proof of the regularity lemma in $\mathbb{F}_2^n$. In particular, suppose, at the current stage, the subspace given by the internal nodes of the parity decision tree is H, and let $\{H_1, \dots, H_M\}$ denote the cosets corresponding to the leaves. Consider the potential function

$$\mathrm{ExpImb}[A, H] := \frac{1}{M} \sum_{i=1}^{M} |\widehat{A_{H_i}}(0^n)|^2,$$

where we recall that $\widehat{A_{H_i}}(0^n) = \mathrm{Vol}_{H_i}(A)$. Note that $\mathrm{ExpImb}[A, H] \in [0, 1]$. Informally, ExpImb captures the "expected imbalance" of A restricted to the leaf nodes of the tree at the current stage.

Lemma 2.2 of [Gre05] (alternatively, see [O'D07]) states that if there exists a leaf node A_{H_i} and a parity $\alpha \in \mathbb{F}_2^n$ such that $\left|\widehat{A_{H_i}}(\alpha)\right| \geq \varepsilon/2$, then upon splitting all nodes at the current level on the parity α—with $H' \leq H$ being the subspace corresponding to the resulting tree—we have

$$\mathrm{ExpImb}[A, H'] \geq \mathrm{ExpImb}[A, H] + \frac{\varepsilon^3}{4}.$$

It follows that if the condition in Line 2(d) of ConstructDTdoesn't hold, then after Step 2(e), the value of ExpImb increases by at least $4/\varepsilon^3$. It follows that Step 2 can be repeated at most $2 \uparrow\uparrow \frac{4}{\varepsilon^3}$ times.

Next, note that the Goldreich–Levin call in Step 2(b) makes at most $\mathrm{poly}\big(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\big)$ queries to A over the run of ConstructDT, and each call to Step 2(e) and Step 3 makes $O\big(\frac{1}{\varepsilon^2}\big)$ and $O\big(\frac{1}{\tau^2}\big)$ many queries (via a standard application of the Chernoff bound). The overall query complexity of ConstructDT follows. The runtime is similarly clear.

Note that we run the Goldreich–Levin algorithm in Step 2(b) on the function $A\restriction_{H_i}$ as opposed to A_{H_i}. It follows from Fact 2.1 that $A\restriction_{H_i}$ is ε/M-quasirandom if and only if A_{H_i} is ε-quasirandom (where M is the number of cosets at a particular stage of the algorithm). We also note that given query access to A_{H_i}, we can simulate query access to $A\restriction_{H_i}$ by checking whether an input x belongs to the coset H_i by querying it on the parity decision tree $\mathcal{T}_{\mathrm{regular}}$.

In the pruning procedure in Step 2(e), the size of each $\mathcal{S}_i$ is at most $O(1/\varepsilon^2)$. A union bound over the Goldreich–Levin and estimation procedures implies that with probability 9/10, the function computed by $\mathcal{T}_{\mathrm{regular}}$ indicates whether a point x is in a coset H' for which $A_{H'}$ is ε-quasirandom and also $\mathrm{Vol}_{H'}(A) \geq \tau$. It follows that $\mathcal{O}^A_{A'}$ is an exact oracle for A'; it also clearly makes exactly 1 query to A on any input. ∎

3.3 Approximately Simulating Sumsets Note that Proposition 3.1 asserts, for arbitrary $A \subseteq \mathbb{F}_2^n$, the existence of a structured subset $A' \subseteq A$ (which is "almost all of A") and a subspace $H \leq \mathbb{F}_2^n$ such that A'^{+y}_H is ε-quasirandom for all $y \in \mathbb{F}_2^n$. The following lemma indicates why such a decomposition is useful towards our goal of (approximately) simulating sumsets.

Lemma 3.1. *Let* $A \subseteq \mathbb{F}_2^n$ *be arbitrary and let* $H \leq \mathbb{F}_2^n$ *be a subspace. Suppose, for* $x, y \in \mathbb{F}_2^n$*,*

1. A_{x+H}, A_{y+H} *are* ε*-quasirandom (in the sense of Definition 3); and*
2. $\mathrm{Vol}_{x+H}(A), \mathrm{Vol}_{y+H}(A) \geq \tau$ *for some* $\tau > 0$*.*

Then we have

$$\mathrm{Vol}_{x+y+H}(A + A) \geq 1 - O\left(\frac{\varepsilon^2}{\tau^4}\right). \tag{3.9}$$

Proof. For ease of notation, define the ε-quasirandom functions $f : x + H \to \{0,1\}$ and $g : y + H \to \{0,1\}$ as

$$f := A_{x+H} \qquad \text{and} \qquad g := A_{y+H}.$$

Consider $h := f * g$ and note that $\operatorname{supp}(h) = A_{x+H} + A_{y+H}$. From Equation (2.4), we have that

$$h = \sum_{\alpha \in H} \widehat{f}(\alpha)\widehat{g}(\alpha)\chi_\alpha \geq \tau^2 + \underbrace{\sum_{0^n \neq \alpha \in H} \widehat{f}(\alpha)\widehat{g}(\alpha)\chi_\alpha}_{=:\Gamma}. \tag{3.10}$$

Note that $\mathbf{E}_{\boldsymbol{x} \sim H}\left[\Gamma(\boldsymbol{x})\right] = 0$ and

$$\mathop{\mathbf{E}}_{\boldsymbol{x} \sim H}\left[\Gamma(\boldsymbol{x})^2\right] = \sum_{0^n \neq \alpha \in H} \widehat{f}(\alpha)^2 \widehat{g}(\alpha)^2 \leq \max_{0^n \neq \alpha \in H} \widehat{f}(\alpha)^2 \left(\sum_{0^n \neq \alpha \in H} \widehat{g}(\alpha)^2 \right) \leq \varepsilon^2$$

as f is ε-quasirandom. It then follows from Chebyshev's inequality that

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim H}\left[\left|\Gamma(\boldsymbol{x})\right| \geq \frac{\tau^2}{2} \right] = O\left(\frac{\varepsilon^2}{\tau^4}\right) \qquad \text{and so} \qquad \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim H}\left[h(\boldsymbol{x}) > 0\right] \geq 1 - O\left(\frac{\varepsilon^2}{\tau^4}\right),$$

completing the proof. ∎

Remark 6. *Note that the lower bound of* $1 - O(\varepsilon^2/\tau^4)$ *in Equation* (3.9) *cannot be improved to* 1, *as witnessed by the following example: Let* $A \subseteq \mathbb{F}_2^n$ *be defined as*

$$A(x) = \begin{cases} 0 & \sum x_i \geq \frac{n}{2} \\ 1 & \sum x_i \leq \frac{n}{2} - 1 \end{cases}$$

and let $H = \mathbb{F}_2^n$. *As* A *is a symmetric function,* $\widehat{A}(\alpha)$ *only depends on* $\sum_i \alpha_i$. *It is easy to check using Parseval's identity that* $\left|\widehat{A}(\alpha)\right| \leq O\left(\frac{1}{\sqrt{n}}\right)$, *and that* $A + A \subsetneq \mathbb{F}_2^n$ *(as we clearly have* $1^n \notin A + A$*).*

Lemma 3.1 suggests a natural approach towards our goal of approximately simulating sumsets: Given the parity decision tree $\mathcal{T}_{\text{regular}}$ as in Algorithm 3.1, for every pair of leaves—say, corresponding to cosets $x + H$ and $y + H$—with non-trivial $\operatorname{Vol}_{x+H}(A'), \operatorname{Vol}_{y+H}(A')$, we set $\operatorname{Vol}_{x+y+H}\left(A' + A'\right) = 1$. This procedure is outlined in Algorithm 3.2; more formally, we have Proposition 3.3.

ALGORITHM 3.2.

Input: Query access to $A \subseteq \mathbb{F}_2^n$, quasirandomness parameter ε, density threshold τ

Output: An approximate oracle $\mathcal{O}_{A'+A'}$ where A' as in Proposition 3.1t

SIMULATE-SUMSET(A, ε, τ):

1. Obtain $\left(\mathcal{O}^A_{A'}, \mathcal{T}_{\text{regular}}\right)$ via
$$\left(\mathcal{O}^A_{A'}, \mathcal{T}_{\text{regular}}\right) \leftarrow \text{CONSTRUCTDT}(A, \varepsilon, \tau).$$
Let $\alpha_i \in \mathbb{F}_2^n$ denote the label associated to internal nodes at depth i.
2. We will write $(b_1, \ldots, b_k)$ with $b_i \in \mathbb{F}_2$ to denote the root-to-leaf path obtained by taking the outgoing edge labeled by b_i from the internal node α_i,and will identify leaves of $\mathcal{T}_{\text{regular}}$ with these tuples.
3. Initialize $\mathcal{T}_{\text{sum}}$ as a copy of $\mathcal{T}_{\text{regular}}$, and associate all leaves with the identically-0 function.
4. For all pairs of leaf nodes $(b_1^{(1)}, \ldots, b_k^{(1)})$ and $(b_1^{(2)}, \ldots, b_k^{(2)})$ in $\mathcal{T}_{\text{regular}}$:
   (a) If for both of the leaf nodes in the pair, the function associated with the leaf node is not the identically-0, function, then set the function associated to the leaf node $(b_1^{(1)} + b_1^{(2)}, \ldots, b_k^{(1)} + b_k^{(2)})$ in $\mathcal{T}_{\text{sum}}$ to be the identically-1 function.
5. Define the oracle $O_{A'+A'}$ to be the function computed by $\mathcal{T}_{\text{sum}}$.

PROPOSITION 3.3. *Let $A \subseteq \mathbb{F}_2^n$ be an arbitrary subset. Given query access to A, and $\varepsilon, \tau > 0$, let $A' \subseteq A$ as in Proposition 3.1. The procedure* SIMULATE-SUMSET(A, ε, τ) *described in Algorithm 3.2:*

1. *Makes* $\text{poly}\left(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\right)$ *queries to A and does a* $\text{poly}\left(n, 2 \uparrow\uparrow \frac{8}{\varepsilon^3}, \frac{1}{\tau}\right)$ *time computation; and*
2. *With probability* $9/10$*, outputs an* $\left(O\left(\varepsilon^2/\tau^4\right), 0, n\right)$*-oracle $\mathcal{O}^A_{A'+A'}$ for $A' + A'$.*

Proof. Note that the number of queries made to A follows from Proposition 3.2, and the runtime is immediate from Step 4. The second item above follows from Lemma 3.1. ∎

Note that Theorem 3.1 follows immediately from Propositions 3.2 and 3.3. Furthermore, we can easily estimate $\text{Vol}\left(A' + A'\right)$ (where A' as in Theorem 3.1) via random sampling. A standard application of the Chernoff bound shows that $O\left(\log(1/\delta)/\gamma^2\right)$ many samples suffice to get a $\pm\gamma$ additive approximation to $\text{Vol}\left(A' + A'\right)$ with probability at least $1 - \delta$.

4 An Implicit Regularity Lemma in $\mathbb{F}_2^n$

In this section, we present the following "implicit" version of Theorem 3.1 that makes only $O_\varepsilon(1)$ queries, independent of the ambient dimension n.

THEOREM 4.1. (MAIN THEOREM) *Let $A \subseteq \mathbb{F}_2^n$ be an arbitrary subset, and let $\varepsilon, \tau > 0$. Given query access to A, there exists an algorithm that makes $O_{\varepsilon,\tau}(1)$ queries to A and does an $O_{\varepsilon,\tau}(1) \cdot n$ time computation and outputs with probability at least $9/10$:*

1. *A* $\left(0, O_{\varepsilon,\tau}(1), O_{\varepsilon,\tau}(1) \cdot n\right)$*-oracle $\mathcal{O}^A_{A'}$ to the indicator function of $A' \subseteq A$ where* $\text{Vol}\left(A \setminus A'\right) \leq \varepsilon + \tau$*; and*

2. *A* $\left(O(\varepsilon^2/\tau^4), O_{\varepsilon,\tau}(1), O_{\varepsilon,\tau}(1)\cdot n\right)$*-oracle* $\mathcal{O}^A_{A'+A'}$ *to the indicator function of the sumset* $A'+A'$.

In Section 4.1, we state an "implicit" version of the Goldreich–Levin algorithm (which appears to be a folklore result in coding theory), which we then use to obtain query-efficient versions of Algorithms 3.1 and 3.2 in Sections 4.2 and 4.3 respectively.

4.1 Implicitly Finding Significant Fourier Coefficients To explain what we mean by the qualifier "implicit", recall the usual statement of the Goldreich-Levin algorithm (Theorem 2.2). Informally, the theorem states that given oracle access to $A \subseteq \mathbb{F}_2^n$, there exists an algorithm that outputs an *explicit* set $\mathcal{S} \subseteq \mathbb{F}_2^n$ with the elements of S corresponding to the "significant" Fourier coefficients of A. In the language of coding theory, this a *list decoding algorithm* for the Hadamard code.

The theorem as stated, however, is not useful for us—in particular, as our target query complexity is independent of n, we cannot hope to obtain $\mathcal{S}$ explicitly. We will instead obtain *implicit access* to the set $\mathcal{S}$. We next state the refined guarantee for the Goldreich-Levin algorithm that we require.

THEOREM 4.2. (IMPLICIT GOLDREICH–LEVIN THEOREM) *Given oracle access to set* $A \subseteq \mathbb{F}_2^n$*, significance threshold* θ *and confidence parameter* δ*, the algorithm* IMPLICIT-GL$(A, \varepsilon, \tau, \delta)$ *makes* $\mathrm{poly}(1/\theta)\cdot\log(1/\delta)$ *queries to* A *and with probability at least* $1-\delta$ *for some* $T \le 4/\theta^2$*, outputs* T *oracle machines* $\mathcal{O}_1^A, \ldots, \mathcal{O}_T^A$ *with the following guarantee:*

1. *For each* $1 \le i \le T$*, there is a distinct* $\alpha_i \in \mathbb{F}_2^n$ *such that* $\mathcal{O}_i^A$ *is a probabilistic oracle machine for the function* χ_{α_i}*. The query complexity of each oracle* $\mathcal{O}_i^A$ *is* $\mathrm{poly}(1/\theta)$.
2. *For each* $1 \le i \le T$*,* $|\widehat{A}(\alpha_i)| \ge \theta/2$.
3. *For any* $\beta \in \mathbb{F}_2^n$ *such that* $|\widehat{A}(\beta)| \ge \theta$*, there is a* $1 \le j \le T$*, such that* $\alpha_j = \beta$.

The crucial feature of Theorem 4.2 is that the query complexity of both the routine IMPLICIT-GL as well as the probabilistic oracle machines is independent of n and is just dependent on the significance parameter θ and confidence parameter δ. Further, note that the algorithm IMPLICIT-GL does not just give an oracle for α_i (which would be a procedure which, on input $j \in [n]$, outputs the value of the j-th coordinate of $\alpha_i \in \mathbb{F}_2$), but rather it gives an oracle for χ_{α_i} (which of course, on input $x \in \mathbb{F}_2^n$, outputs the value of $\chi_{\alpha_i}(x) \in \mathbb{F}_2$). In the parlance of coding theory, IMPLICIT-GL is a *constant query* algorithm for *local list correction*.

We note that in the usual formulation of Goldreich-Levin (see [GL89, AB09]), the algorithm outputs all the parities, i.e., the entire set $\mathcal{S}$. As the description size of $\mathcal{S}$ is $\Omega(n)$, the query complexity is necessarily $\Omega(n)$. However, the formulation in Theorem 4.2 can easily be obtained by the obvious modification of Rackoff's analysis [Gol01] of the Goldreich-Levin algorithm and seems to be folklore in coding theory [Sud21]. In fact, a weaker statement, namely that Goldreich-Levin is a *constant query local list decoding* algorithm has already been explicitly noted in literature [Tre04, KS13].

We describe the routine IMPLICIT-GL in detail in Algorithm 4.1. To do so, we first need to define the procedure LINEARITY-TEST.

DEFINITION 7. *The procedure* LINEARITY-TEST$(\mathcal{D}, \tau_c, \tau_\ell, \kappa)$ *takes as input oracle access to* $\mathcal{D}: \mathbb{F}_2^n \to \mathbb{F}_2$*, distance parameters* $\tau_c < \tau_\ell$ *and confidence parameter* κ*. With probability* $1-\kappa$*,* LINEARITY-TEST *can distinguish between the cases (i)* $\mathcal{D}$ *is* τ_c*-close to some parity* χ *and (ii)* $\mathcal{D}$ *is* τ_ℓ*-far from every parity* χ.

The Fourier analysis based proof [BCH+96] of the standard linearity tester [BLR93] can be (easily) used to obtain such a procedure LINEARITY-TEST as long as $\tau_c < \tau_\ell/3$. The query complexity of the procedure is $\log(1/\kappa)\cdot\mathrm{poly}(1/|\tau_\ell - 3\tau_c|)$.

At a high level, the routine IMPLICIT-GL starts exactly the same way as of the Goldreich-Levin algorithm—in particular, the standard analysis of Goldreich-Levin shows the following (for Step 3(b)): For any α such that $|\widehat{f}(\alpha)| > \theta$, there is some $b \in \mathbb{F}_2^t$ such that

$$\Pr_{\boldsymbol{x}\sim\mathbb{F}_2^n}\left[\chi_\alpha(\boldsymbol{x}) \ne \mathcal{D}_b^A(\boldsymbol{x})\right] \le 1/10.$$

It easily follows that for any α such that $|\widehat{f}(\alpha)| > \theta$, there is some $b \in \mathbb{F}_2^t$ such that $\mathcal{O}_b^A$ is a probabilistic oracle for χ_α. Further, for any $b \in \mathbb{F}_2^t$, $\mathcal{O}_b^A$ is a probabilistic oracle for some parity. In Step 3(d), we compute the correlation between $\mathcal{O}_b^A$ and A up to $\pm\theta/4$. This implies that all $\mathcal{O}_b^A$ which survive satisfy properties (2) and (3) of Theorem 4.2. We leave the detailed analysis to the interested reader.

ALGORITHM 4.1.

Input: Query access to $A : \mathbb{F}_2^n \to \{0,1\}$, confidence parameter $\delta > 0$ and significance parameter $\theta > 0$.

Output: Probabilistic oracles $\mathcal{O}_1^A, \ldots, \mathcal{O}_T^A$ for some $T \le 4/\theta^2$. The oracle machines $\mathcal{O}_1^A, \ldots, \mathcal{O}_T^A$ satisfy conditions (1) and (2) from Theorem 4.2.

IMPLICIT-GL(A, θ, δ):

1. Let
$$t = \log\left(\frac{1}{\theta^2}\right) + O(1)$$
and initialize $\mathcal{S} = \emptyset$.
2. Pick $X_1, \ldots, X_t \in \mathbb{F}_2^n$ uniformly at random.
3. For all $b := (b_1, \ldots, b_t) \in \mathbb{F}_2^t$:
   (a) For all $\emptyset \neq S \subseteq [t]$:
      i. Define $X^S := \sum_{i \in S} X_i$.
      ii. Define $b^S := \sum_{i \in S} b_i$.
   (b) Define $\mathcal{D}_b^A : \mathbb{F}_2^n \to \{0,1\}$ as
$$\mathcal{D}_b^A(x) := \operatorname*{maj}_{\emptyset \neq S \subseteq [t]} \left\{ A\Big(X^S + x\Big) + b^S \right\}.$$
   (c) Define $\delta_1 := \frac{\delta\theta^2}{4}$. Run LINEARITY-TEST$\big(\mathcal{D}_b^A, 1/20, 1/5, \delta_1\big)$. If LINEARITY-TEST does not accept, discard $\mathcal{D}_b^A$.
   (d) Define $\mathcal{O}_b^A : \mathbb{F}_2^n \to \{0,1\}$ as follows: choose $y_1, \ldots, y_R \in \mathbb{F}_2^n$ where $R = \Theta(\log(1/\delta))$.
$$\mathcal{O}_b^A(x) := \operatorname*{maj}_{1 \le j \le R} \left\{ \mathcal{D}_b^A(x + y) + \mathcal{D}_b^A(y) \right\}.$$
   (e) Estimate $\widehat{\Theta}_b := \langle A, \mathcal{O}_b^A \rangle$ up to an additive error of $\pm\theta/4$ and confidence δ_1. If the estimate $\widehat{\Theta}_b < 3\theta/4$, discard $\mathcal{O}_b^A$.
4. Output all $\mathcal{O}_b^A$ which survive.

4.2 A Query-Efficient Version of Algorithm 3.1 Recall that Algorithm 3.1 takes in as input query access to $A \subseteq \mathbb{F}_2^n$, a quasirandomness parameter $\varepsilon > 0$, and a density threshold $\tau > 0$, and outputs a $\big(0, 1, O(n)\big)$-oracle $\mathcal{O}_{A'}^A$ for A' where $A' \subseteq A$ is as in Proposition 3.1. The value of this oracle $\mathcal{O}_{A'}^A$ on an input $x \in \mathbb{F}_2^n$ is obtained by routing x through the decision tree $\mathcal{T}_{\text{regular}}(x)$ (recall that each internal node of $\mathcal{T}_{\text{regular}}(x)$ is labeled by an "explicit" parity function χ_{α_i} obtained from some call to the GOLDREICH–LEVIN algorithm), and outputting the value $A(x)$ if a 1-leaf is reached (if a 0-leaf is reached the output is 0). This call to A at the leaf that x reaches is

why $\mathcal{O}^A_{A'}$ makes one (and only one) call to the oracle for A.

In contrast, in the query-efficient regime we cannot use the standard Goldreich–Levin algorithm because its $\Omega(n)$ query complexity is prohibitively high; instead we replace each call to Goldreich–Levin with a call to Implicit-GL. While Goldreich–Levin returns explicit parity functions which label the various nodes of $\mathcal{T}_{\text{regular}}$, the ConstructImplicitDT procedure constructs an "implicit" decision tree in which each node queries some probabilistic oracle machine (that was returned by Implicit-GL) to obtain the value of the desired parity function. Consequently, a call to the oracle $\mathcal{O}^A_{A'}$ produced by ConstructImplicitDT makes $d \cdot \ell + 1$ calls to A, where d is the depth of the implicit decision tree and ℓ is the number of oracle calls to A that are made by each parity oracle produced by Implicit-GL. Crucially, both d and ℓ are values that are $O_{\varepsilon,\tau}(1)$ and completely independent of n.

In addition to constructing an implicit decision tree, ConstructImplicitDT also needs to check for linear independence of the obtained parity oracles (see Step 2(e)(i) of Algorithm 3.1) in a query-efficient way. We detail the performance guarantee of ConstructImplicitDT in the following proposition:

Proposition 4.1. *Let $A \subseteq \mathbb{F}_2^n$ be an arbitrary subset. Given query access to A and $\varepsilon, \tau > 0$, there exists an algorithm* ConstructImplicitDT *that:*

1. *Makes $O_{\varepsilon,\tau}(1)$ queries to A and does an $O_{\varepsilon,\tau}(1)$ time computation; and*
2. *With probability $9/10$, outputs a probabilistic $\big(0, O_{\varepsilon,\tau}(1), O(n)\big)$-oracle $\mathcal{O}^A_{A'}$ for A' where $A' \subseteq A$ is as in Proposition 3.1.*

Proof. The ConstructImplicitDT procedure is obtained by modifying the ConstructDT procedure presented in Algorithm 3.1 in the following ways.

1. In Line 2(a) of ConstructDT, instead of maintaining a list of explicit cosets $H_1, \dots, H_M$, the algorithm maintains a list of probabilistic oracle machines $\mathcal{O}^A_1, \dots, \mathcal{O}^A_{\log M}$ (obtained from calls to Implicit-GL) for the $\log M$ parities which define the cosets $H_1, \dots, H_M$.
2. In Line 2(b), to simulate access to $A\restriction_{H_i}$ on an input x, the algorithm queries the $\log M$ oracle machines and uses the obtained responses to determine whether or not x belongs to the relevant coset. In addition, each call to Goldreich–Levin$\big(A\restriction_{H_i}, \varepsilon/M, \delta\big)$ in Step 2(b) is replaced with a call to Implicit-GL$\big(A\restriction_{H_i}, \varepsilon/M, \delta\big)$. Note that each set $\mathcal{S}_i$ produced by a call to Implicit-GL is now a set of oracles for parity functions.
3. Each estimate of $\left|\widehat{A_{H_i}}(\alpha)\right|$ in Line 2(c) is obtained by random sampling, using the simulated version of $A\restriction_{H_i}$ described above and the oracle for the parity function for χ_α.
4. In Line 2(e)(i), since the algorithm does not explicitly have the vectors in $\mathbb{F}_2^n$ that define the parity functions, it instead uses the following simple sampling-based procedure to check linear independence:
   - Given a collection of oracles $\{\mathcal{O}_{\chi_1}, \mathcal{O}_{\chi_2}, \dots, \mathcal{O}_{\chi_k}\}$ where each χ_i is some parity function, the algorithm queries all of them on N independent uniform random points in $\mathbb{F}_2^n$ and builds the corresponding $k \times N$ matrix with entries in $\mathbb{F}_2$.
   - Then the algorithm checks if the rank of this matrix is k.

   It is clear that if the parities $\{\alpha_i\}$ are not linearly independent, then the $k \times N$ matrix constructed by this procedure will not have rank k. On the other hand, a simple probabilistic argument shows that if the k parities are linearly independent, then the matrix will have rank k except with failure probability at most 2^{k^2-N}.
5. In Line 3 the estimate of $\mathrm{Vol}_{H_i}(A)$ is obtained using the $\log M$ oracle machines mentioned above in the obvious way; and
6. Finally, the output oracle $\mathcal{O}^A_{A'}$ is the obvious analogue of $\mathcal{T}_{\text{regular}} \cdot A$ where again the $\log M$ oracle machines are used to route inputs through the implicit decision tree to the correct coset.

The analysis of correctness is essentially the same as that of Proposition 3.2. We note that while ConstructDT outputs a deterministic oracle, ConstructImplicitDT outputs a probabilistic oracle (because of the probabilistic oracles for parity functions that it uses). For the query complexity, a tedious but straightforward inductive argument shows that the values of δ (for each call to Implicit-GL) and N (for each execution of Line 2(e)(i)) can be taken to be independent of n, yielding the claimed query complexity. ∎

4.3 A Query-Efficient Version of Algorithm 3.2 Finally, the query-efficient version of Algorithm 3.2, which we call Implicit-Simulate-Sumset, works in the obvious way. In Line 1, the call to ConstructDT is replaced by a call to ConstructImplicitDT, and the "explicit" decision tree $\mathcal{T}_{\text{regular}}$ is replaced by the ensemble of parity oracles corresponding to the coset decomposition. We observe that while in the explicit algorithm Simulate-Sumset, the function $\mathcal{T}_{\text{sum}}$ can be evaluated on an input $x \in \mathbb{F}_2^n$ without making any calls to A, in our implicit setting we need to query the ensemble of parity oracles (and hence make queries to A) for each evaluation of $\mathcal{T}_{\text{sum}}$ on an input x (to route x to the correct leaf node in the implicit tree for $\mathcal{T}_{\text{sum}}$). Theorem 4.1 follows from Proposition 4.1 and the obvious analogue of Proposition 3.3 for Implicit-Simulate-Sumset.

5 Conclusion and Future Work

Our results suggest a number of interesting directions for future work. In particular, a broad goal is to develop query-efficient procedures for simulating oracle access to other types of sumsets, or sumsets over other domains. Our approach extends relatively straightforwardly to the sumset $A + B$ for distinct sets $A, B \subseteq \mathbb{F}_2^n$ given access to oracles to both A and B, and likewise to the iterated sumset $A + \cdots + A = kA$ for any constant k.

A more ambitious extension would be to handle the sumset $A + A$ when A is an arbitrary subset of some other Abelian (or potentially non-Abelian) group G. Green's regularity lemma is known to hold for general finite Abelian groups [Gre05], but to obtain constant query complexity independent of $|G|$ via our approach it seems that one would need an "implicit" procedure for finding large Fourier coefficients of functions from G to $\mathbb{R}$. As observed in [DGKS08], the algorithm of Goldreich and Levin does not generalize to finding large Fourier coefficients over arbitrary finite groups. There is an alternative algorithm, due to Kushilevitz and Mansour [KM93], for finding large Fourier coefficients of functions $\mathbb{F}_2^n \to \mathbb{R}$ which has been generalized to arbitrary finite Abelian groups G by Akavia et al. [AGS03], but the query complexity of the Kushilevitz-Mansour algorithm grows with n and the query complexity of the Akavia et al. algorithm grows with $|G|$. Developing a constant-query "implicit" version of the algorithm of Akavia et al. for general finite groups is an interesting specific direction for future work.

Yet another intriguing problem, as mentioned in Section 1.2, is to try to develop a query-efficient algorithm for simulating an oracle to $\frac{A+A}{2}$ (where addition denotes the Minkowski sum) or $\text{Conv}(A)$ (the convex hull of A) when A is a subset of $\mathbb{R}^n$ (and we view $\mathbb{R}^n$ as endowed with the standard Normal $\mathcal{N}(0,1)^n$ distribution).

Acknowledgements

A.D. is supported by NSF grants CCF-1910534, CCF-1926872, and CCF-2045128. S.N. is supported by NSF grants CCF-1563155 and by CCF-1763970. R.A.S. is supported by NSF grants CCF-1814873, IIS-1838154, CCF-1563155, and by the Simons Collaboration on Algorithms and Geometry. This material is based upon work supported by the National Science Foundation under grant numbers listed above. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[AGS03] Adi Akavia, Shafi Goldwasser, and Samuel Safra. Proving hard-core predicates using list decoding. In *Proc. 44th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 146–159. IEEE Computer Society Press, 2003.

[BCH+96] M. Bellare, D. Coppersmith, J. Hastad, M. Kiwi, and M. Sudan. Linearity testing in characteristic two. *IEEE Trans. on Information Theory*, 42(6):1781–1795, 1996.

[BFL13] Arnab Bhattacharyya, Eldar Fischer, and Shachar Lovett. Testing low complexity affine-invariant properties. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1337–1355. SIAM, 2013.

[BGS15] Arnab Bhattacharyya, Elena Grigorescu, and Asaf Shapira. A unified framework for testing linear-invariant properties. *Random Structures & Algorithms*, 46(2):232–260, 2015.

[Bha13] Arnab Bhattacharyya. Guest column: On testing affine-invariant properties over finite fields. *ACM SIGACT News*, 44(4):53–72, 2013.

[BK89] M. Blum and S. Kannan. Designing Programs That Check Their Work. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, page 86–97, 1989.

[Bla09] Eric Blais. Testing juntas nearly optimally. In *Proc. 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 151–158, 2009.

[BLR93] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993. Earlier version in STOC'90.

[CG92] F. R. K. Chung and R. L. Graham. Quasi-random subsets of $\mathbb{Z}_n$. *Journal of Combinatorial Theory. Series A*, 61(1):64–86, 1992.

[CRT05] Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the Minimum Spanning Tree Weight in Sublinear Time. *SIAM J. Comput.*, 34(6):1370–1379, 2005.

[DGKS08] Irit Dinur, Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. Decodability of group homomorphisms beyond the johnson bound. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 275–284. ACM, 2008.

[ERS18] Talya Eden, Dana Ron, and C. Seshadhri. On approximating the number of k-cliques in sublinear time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, page 722–734, 2018.

[Fis01] E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of the European Association for Theoretical Computer Science*, 75:97–126, 2001.

[FKR+04] E. Fischer, G. Kindler, D. Ron, S. Safra, and A. Samorodnitsky. Testing juntas. *J. Computer & System Sciences*, 68(4):753–787, 2004.

[Fre73] G.A. Freiman. *Foundations of a Structural Theory of Set Addition.* Translations of mathematical monographs. American Mathematical Society, 1973.

[GGL+00] O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32. ACM, 1989.

[Gol01] Oded Goldreich. *The Foundations of Cryptography: volume 1.* Cambridge University Press, Cambridge, 2001.

[Gol17] Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017.

[Gre05] B. Green. A Szemerédi-type regularity lemma in abelian groups, with applications. *Geometric and Functional Analysis (GAFA)*, 15:340–376, 2005.

[HHL16] Hamed Hatami, Pooya Hatami, and Shachar Lovett. General systems of linear forms: equidistribution and true complexity. *Advances in Mathematics*, 292:446–477, 2016.

[HL13] Hamed Hatami and Shachar Lovett. Estimating the distance from testable affine-invariant properties. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 237–242. IEEE, 2013.

[JPRZ04] Charanjit S. Jutla, Anindya C. Patthak, Atri Rudra, and David Zuckerman. Testing low-degree polynomials over prime fields. In *Proc. 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 423–432. IEEE Computer Society Press, 2004.

[KM93] E. Kushilevitz and Y. Mansour. Learning Decision Trees Using the Fourier Spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, December 1993.

[KMS18] S. Khot, D. Minzer, and S. Safra. On Monotonicity Testing and Boolean Isoperimetric-type Theorems. *SIAM Journal on Computing*, 41(6), 2018.

[KNOW14] Pravesh Kothari, Amir Nayyeri, Ryan O'Donnell, and Chenggang Wu. Testing surface area. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014*, pages 1204–1214. SIAM, 2014.

[KOS08] A. Klivans, R. O'Donnell, and R. Servedio. Learning geometric concepts via Gaussian surface area. In *Proc. 49th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 541–550, 2008.

[KS08] T. Kaufman and M. Sudan. Algebraic property testing: the role of invariance. In *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 403–412, 2008.

[KS13] Swastik Kopparty and Shubhangi Saraf. Local list-decoding and testing of random linear codes from high error. *SIAM J. Comput.*, 42(3):1302–1326, 2013.

[Nee14] Joe Neeman. Testing Surface Area with Arbitrary Accuracy. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, page 393–397, 2014.

[O'D07] Ryan O'Donnell. Lecture 28: Szemerédi's Regularity Lemma in $\mathbb{F}_2^n$. In *CMU 15-859S: Analysis of Boolean Functions.* 2007. Available at https://www.cs.cmu.edu/~odonnell/boolean-analysis/lecture28.pdf.

[O'D14] Ryan O'Donnell. *Analysis of Boolean Functions.* Cambridge University Press, 2014.
[PR07] Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theoretical Computer Science*, 381(1):183–196, 2007.
[Ron01] D. Ron. Property testing (a tutorial). In S. Rajasekaran, P. M. Pardalos, J. H. Reif, and J. D. P. Rolim, editors, *Handbook of Randomzied Computing, Volume II.* Kluwer, 2001.
[RRS+12] Dana Ron, Ronitt Rubinfeld, Muli Safra, Alex Samorodnitsky, and Omri Weinstein. Approximating the Influence of Monotone Boolean Functions in $O(\sqrt{n})$ Query Complexity. *ACM Trans. Comput. Theory*, 4(4), November 2012.
[RS96] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25:252–271, 1996.
[Rub06] R. Rubinfeld. Sublinear time algorithms. Proceedings of the International Congress of Mathematicians (ICM), 2006.
[Ruz99] Imre Z. Ruzsa. An analog of Freiman's theorem in groups. In Deshouilliers Jean-Marc, Landreau Bernard, and Yudin Alexander A., editors, *Structure theory of set addition*, number 258 in Astérisque. Société mathématique de France, 1999.
[RV19] Ronitt Rubinfeld and Arsen Vasilyan. Approximating the Noise Sensitivity of a Monotone Boolean Function. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019*, volume 145, pages 52:1–52:17, 2019.
[San12] Tom Sanders. On the Bogolyubov–Ruzsa lemma. *Analysis and PDE*, 5(3):627 – 655, 2012.
[STV01] Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, March 2001.
[Sud21] Madhu Sudan. Personal communication. 2021.
[SvH18] Yair Shenfeld and Ramon van Handel. The equality cases of the Ehrhard-Borell inequality. *Adv. Math.*, 331:339–386, 2018.
[Sze78] E. Szemerédi. Regular partitions of graphs. *Colloq. Internat. CNRS: Problèmes combinatoires et théorie des graphes*, 260:399–401, 1978.
[Tre04] Luca Trevisan. Some applications of coding theory in computational complexity. *ArXiv preprint cs/0409044*, 2004.
[TV06] T. Tao and V. Vu. *Additive Combinatorics.* Cambridge Studies in Advanced Mathematics, Cambridge University Press, Cambridge, 2006.
[Yos14] Yuichi Yoshida. A characterization of locally testable affine-invariant properties via decomposition theorems. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 154–163, 2014.