# Quiet Planting for k-SAT, Multiple Solutions of Arbitrary Geometry

**Ali Ahmadi** AHMADIA@UMD.EDU
**Kiarash Banihashem** KIARASH@UMD.EDU
**Iman Gholami** IGHOLAMI@UMD.EDU
**Mohammad Taghi Hajiaghayi** HAJIAGHA@UMD.EDU
**Jan Olkowski** JAN.OLKOWSKI@GMAIL.COM
*University of Maryland, College Park, Maryland, USA*

**Editors:** Steve Hanneke and Tor Lattimore

## Abstract

Recent work on "quiet planting" in combinatorial optimization aims to generate instances with a hidden solution that is hard to recover, typically by making the planted distribution statistically indistinguishable from uniform for specific algorithms, such as statistical queries. A prominent example is planted $k$-SAT, where $O(n^{k/2})$ clauses can be planted while maintaining indistinguishability from uniform instances, evidenced by prior hardness results which also align with findings in SAT refutation. Despite extensive research and practical use in benchmarking SAT solvers, the challenge of quietly planting multiple solutions while preserving hardness has remained an open problem.

This work initiates the study of quiet planting with an arbitrary number of solutions, proposing the first method to construct quiet planting distributions for $k$-SAT formulas that accommodate more than one solution. We provide statistical query lower bounds for distinguishing these planted instances from uniform ones, and our method allows for planting solutions with arbitrary geometric relationships, including varying Hamming distances. A key innovation facilitating multiple solutions is the ability to incorporate arbitrary correlations between variable selection in clauses and their negation patterns, departing from prior approaches. We also investigate the worst-case complexity of SAT by showing the difficulty in distinguishing satisfiable instances with numerous solutions from unsatisfiable ones, addressing an open problem of Hsieh, Mohanty, and Xu (CCC'22). From a technical standpoint, we generalize the concept of $(r-1)$-wise uniformness in clause distributions, proving hardness holds if the marginal distribution over negation patterns is $(r-1)$-wise uniform, and reveal a connection to binary linear codes, demonstrating how a $[k, t, r]$ code can guide the planting of up to $2^t - 1$ solutions on $k$ variables with $(r-1)$-wise uniform negation distributions.

**Keywords:** Satisfiablity, Quiet Planting, Statistical Query

## 1. Introduction

The $k$-satisfiability problem ($k$-SAT) involves determining whether a solution exists for a boolean formula with $n$ logical variables. The formula consists of $m$ clauses, each containing $k$ different variables or their negations. A solution satisfies the formula if, in each clause, at least one variable or its negation evaluates to true. The NP-completeness of $k$-SAT, for $k \geq 3$, is crucial for understanding computational complexity and demonstrating the difficulty of problems in NP. It finds applications in fields such as automated reasoning and cryptography. The efficacy of $k$-SAT in encoding decision problems has led to its widespread use in reduction techniques, particularly

in addressing problems like Max-SAT and Max Clique, highlighting its adaptability in tackling intricate computational challenges both theoretically and in real-world problem-solving contexts. Beyond the complexity theory applications, $k$-SAT can be used to model many real-world problems spanning areas such as AI planning, physical systems, and cryptography.

Understanding the hardness of k-SAT is particularly important because it is a central benchmark for computational difficulty, used in the evaluation of SAT solvers[1] and as a basis for hardness assumptions in cryptography. A single instance cannot reliably demonstrate hardness because an algorithm can incorporate the specifics of that particular instance into its implementation. Consequently, efforts have focused on characterizing classes of hard instances. One extensively studied class is derived from the random satisfiability problem (random $k$-SAT). This class consists of instances where $m$ clauses are independently drawn from a distribution over all clauses of $k$ literals, without repetitions, across $n$ variables.

For the class of random $k$-SAT instances, the seminal result of Ding et al. (2015) (STOC'15) shows that for sufficiently large $k$ there exists a threshold $\tau_k \in \mathbb{R}$ such that if $m$ is a random variable drawn from $\text{Pois}(n\alpha)$ and $m$ clauses are subsequently drawn independently from the uniform distribution over feasible clauses, then the instance has at least one solution whp[2] if $\alpha < \tau_k$, but has no solutions whp if $\alpha > \tau_k$. Comprehending the difficulty of various classes of random SAT involves focusing on several key aspects. Examples include the refutation problem, which aims to determine whether we can prove that a random SAT instance cannot be solved, and the search/recovery problem, which concentrates on reconstructing a solution from a random SAT instance. Studying these problems is essential for assessing computational boundaries and demonstrating the level of difficulty inherent in various classes of instances.

In this context, the concept of the planted $k$-satisfiability problem (planted $k$-SAT) arises. This version of $k$-SAT follows the same basic setup as random $k$-SAT but with a crucial difference: every instance created is guaranteed to have at least one specific solution, meaning a fixed assignment is always a solution, though other valid solutions may also exist. Naively planted instances for $k$-SAT, in which a solution is fixed and the distribution is uniform over clauses satisfied by this solution, are nevertheless solvable in linear time. This can be done by a majority vote over the number of clauses that evaluate to true or false for each variable. However, some planted k-SAT distributions closely resemble uniform k-SAT distributions, making them difficult to distinguish despite fundamental differences. These distributions, often called *quietly planted distributions*, provide solvable instances that mimic the behavior of uniform random $k$-SAT at certain thresholds—especially where uniform random $k$-SAT is typically unsatisfiable. While having universal results about the hardness of this class of instances remains an open problem, significant progress has been made in developing hardness in restricted models of computation based on plausible conjectures in complexity theory.

In this work, we focus on the statistical query model of computation introduced by Kearns (1993, 1998) (STOC'93). In this model, we are given access to a statistical oracle capable of as estimating expectations of functions of random clauses without explicit access to each clause.

1. The planting method described in this paper has been used in the DARPA program called QuICC (Quantum-Inspired Classical Computing), which aims to apply insights from quantum algorithms to develop Quantum-Inspired (QI) solvers for complex DOD optimization problems, targeting a reduction in computational energy. The SAT instances generated using the method in this paper not only provide theoretical results but also serve as effective tools for testing SAT solvers (Quantum-Inspired). Although many methods exist for generating "hard" test cases for SAT solvers, those generated by the planting method in this paper are considered among the most challenging instances generated for this project.
2. An event occurs with high probability (whp) if its probability tends to one in the limit $n \to \infty$.

Focusing on these algorithms allows us to provide formal hardness results for the the problem of distinguishing between a planted distribution from some reference distribution, typically taken to be the uniform distribution. Formally, considering two probability distributions, denoted as $Q$ and $D$, over the set of clauses with $k$ variables, the planted distribution $Q$ is subject to specific conditions, while the reference distribution $D$ represents the baseline or null distribution. The goal of the corresponding testing problem is to devise a statistical query algorithm capable of determining, based on queries to the statistical oracle, whether the underlying distribution is closer to $Q$ (planted distribution) or $D$ (reference distribution). The model of computation is very general, as many algorithms from various domains, including machine learning, cryptography, and anomaly detection, where distinguishing between different distributions is essential for making informed decisions based on observed data, can be implemented in this model.

Since the introduction of the statistical query, significant results have extended the understanding of the difficulty of planted $k$-SAT in this model. The distribution complexity parameter $r$, a key focus, characterizes the number of clauses needed for efficient recovery of the planted assignment. Subsequent works have focused on developing algorithms depending on $r$ that efficiently identify the planted solution (Bogdanov and Qiao, 2009; Charikar and Wirth, 2004; Applebaum, 2016). Significant advancement in this field was achieved through the work of Feldman et al. (2015a, 2018). Their study focuses on the complexity of distinguishing between a planted $k$-SAT distribution and a random $k$-SAT distribution for the class of *statistical query* algorithms. The lower bounds in Feldman et al. (2015a) effectively justify the barrier of $n^{k/2}$ clauses observed for recovering planted solutions, as well as the closely related refutation problem.

The barrier of $n^{k/2}$ is a known limit for many problems arising around $k$-SAT satisfiability, such as recovery and refutation. Many works have obtained lower bounds showing that specific classes of algorithms cannot go beyond this barrier. In addition to the lower bound for statistical query implied by Feldman et al. (2015a) (STOC'15), algorithms for the refutation problem were introduced by Allen et al. (2015); Raghavendra et al. (2017) (FOCS'15, STOC'17), matching the lower bound established in Kothari et al. (2017) (STOC'17). In this work, we follow the model of computation presented in Feldman et al. (2015a) while at the same time proposing novel and significantly more general results, removing many limitations of the previous methods. While our result relies on the SQ framework of Feldman et al. (2015a), extending it to low-degree tests, directly or via the equivalences in Brennan et al. (2021), is a promising future direction. It is worth noting that they use a stronger notion of statistical dimension.

We note that the recovery problem is closely related to the refutation problem, in which the goal is to provide an algorithm that can refute the existence of a solution for a random instance (i.e., prove that the instance has no solution) with high probability. Hsieh et al. (2022) (CCC'22) studies a version of this problem with multiple solutions, focusing on the problem of certifiable upper bounds for random SAT instances. However, this does not capture the aspect of arbitrary geometry, which we consider in this work. Moreover, in general, lower bounds for refutation do not imply lower bounds for recovery (e.g., see Wein (2023); Banks et al. (2021); Bandeira et al. (2021, 2020)).

Furthermore, Hsieh et al. (2022) constructs a quiet planted distribution of $k$-SAT clauses based on a random $k$-uniform hypergraph with a planted independent set. In this distribution, the clauses are satisfied by $2^{|S|}$ assignments, where $S$ is an independent subset of variables. Therefore, the solution space is determined by all possible values for variables of $S$. Although their approach is outlined without formal proof and it is not clear how large $|S|$ can be compared to $n$ while maintaining indistinguishability, a notable distinction is that their construction does not incorporate the

geometry of the solutions. Another difference is that they focus on selecting variables to achieve multiple solutions, whereas in our paper, for a randomly chosen subset of variables, we plant multiple solutions with arbitrary geometry using the negation pattern. This matters in practice because overly structured or overly similar planted solutions can be exploited by heuristics, making benchmarks less informative. By generating hard instances with many planted solutions and controlled geometry, our approach improves both solver evaluation and our understanding of where hardness comes from.

**Our Contributions.** This paper presents two key results: First, we introduce a novel construction that quietly plants multiple solutions in $k$-SAT instances. Our method generalizes existing quiet planting techniques by allowing arbitrary geometry of solutions, leveraging binary linear codes and $(r-1)$-wise uniformness. Second, building on the statistical query framework, we establish hardness guarantees for our construction by extending existing approaches. These advances significantly expand the applicability of quiet planting methods for generating hard $k$-SAT instances.

We begin by selecting a binary linear code. Assume this binary linear code is $[k, t, r]$, which has block size $k$, dimension $t$, and Hamming distance between codes of $r$ (see Section A). Next, a solution set $A$ is chosen such that its size is bounded by $2^t-1$, ensuring the set remains small relative to the code's dimension. Using the generator matrix, for any choice of $k$ variables $I$, we then define sign pattern distributions $Q_I$, which capture the distributional behavior of the code under different subsets. A central step is proving that these distributions exhibit $(r-1)$-uniformness, meaning that they retain uniformity properties up to dimension $r-1$. Finally, leveraging this structure, we establish a lower bound on the power of statistical query algorithms when applied to instances generated according to this distribution, highlighting inherent limitations in their ability to solve such problems efficiently.

Our result generalizes the work of Feldman et al. (2015a) by allowing the planting of multiple solutions with arbitrary geometry. More specifically, their work focused on the case $r = k$, which corresponds to a planted random $k$-XOR-SAT instance, and proved a similar lower bound to the one we obtain in this paper. It is important to note that instances constructed by both our method and the prior approach can be solved using Gaussian elimination-based methods; however, introducing a small amount of noise can mitigate this vulnerability.

## 2. Preliminaries

**General Notations.** We write $[n] = \{1, \ldots, n\}$. For a vector $a$ of length $n$ and an index $i \in [n]$, we denote its $i^{\text{th}}$ coordinate by $a[i]$. For a sequence $a$ of length $n$ and any index sequence $b \in [n]^\ell$, we write $a[b] = (a[b[1]], \ldots, a[b[\ell]])$. For binary vectors $u, v \in \{0,1\}^n$, the coordinate-wise "XOR" is denoted by $u \oplus v$, and $-u$ denotes the coordinate-wise "NOT". Moreover, for any set $S \subseteq [n]$, we define $\chi_S(u) := \bigoplus_{i \in S} u[i]$.

In many parts of the analysis, we work with vectors in $\{-1,1\}^n$ (where the correspondence is $1 \leftrightarrow 0$ and $-1 \leftrightarrow 1$). In this setting, we extend the $\oplus$ operator so that for any $u, v \in \{-1,1\}^k$, $u \oplus v$ is their coordinate-wise product, and $-u$ flips each sign.

**SAT Notation.** A $k$-SAT formula is a CNF boolean formula in which each clause contains exactly $k$ literals; in a $k$-XOR-SAT, literals within a clause are connected via XOR. We let $n$ denote the total number of variables and $k$ the clause width. Each clause is represented as a tuple $C = (I, x)$, where, $I \in [n]^k$ is a sequence (or set) of $k$ distinct variables (the *variable sequence*), and $x \in \{0,1\}^k$ is

the *sign pattern*. We denote by $X_k$ the set of all clauses of size $k$, and by $\mathcal{I}_k \subseteq [n]^k$ the set of all variable sequences. Bold letters (e.g., $\mathbf{x}$) indicate random objects, while non-bold letters denote fixed values.

An assignment is a vector in $\{0,1\}^n$. A clause $(I, x)$ is said to be *satisfied* by an assignment $\sigma$ if there exists an $i \in [k]$ such that $x[i] = \sigma[I[i]]$. For an assignment $\tau \in \{0,1\}^n$, define $C \oplus \tau := (I,\ x \oplus \tau[I])$, and for any index subset $S \subseteq [k]$, define the derived clause $C[S] := (I[S],\ x[S])$.

When clauses (or their components) are sampled from a distribution $\mathcal{Q}$, we extend the above notation naturally. For example, for an assignment $\tau \in \{-1,1\}^n$, $\mathcal{Q}_\tau = \mathcal{Q} \oplus \tau$ denotes the distribution of $C \oplus \tau$ with $C \sim \mathcal{Q}$, and for a fixed $I$, $Q_I$ denotes the conditional distribution of the sign pattern given that the variable sequence is $I$.

**Statistical Algorithms.** Under the class of *statistical algorithms*, we consider algorithms that access data through statistical oracles. Given a distribution $D$ over a domain $X$, we use the following statistical oracles (see, e.g., Feldman et al. (2015a)):

- **1-MSTAT**$(L)$**:** For a query function $h : X \to \{1, \ldots, L\}$, the oracle returns $h(x)$ for a random $x \sim D$.
- **VSTAT:** Given a parameter $t > 0$ and a query function $h : X \to [0,1]$, it returns a value $v \in [p - \tau,\ p + \tau]$, where

$$p = \mathbb{E}_D[h(x)] \quad \text{and} \quad \tau = \max\left\{\frac{1}{t},\ \sqrt{\frac{p(1-p)}{t}}\right\}.$$

- **MVSTAT**$(t, L)$**:** For $t > 0$, a query function $h : X \to \{0, \ldots, L-1\}$, and a collection $\mathcal{S}$ of subsets of $\{0, \ldots, L-1\}$, the oracle returns a vector $v \in \mathbb{R}^L$ such that for every $Z \in \mathcal{S}$,

$$\left|\sum_{\ell \in Z} v_\ell - p_Z\right| \le \max\left\{\frac{1}{t},\ \sqrt{\frac{p_Z(1-p_Z)}{t}}\right\},$$

  where $p_Z = \Pr_{x \sim D}[h(x) \in Z]$. The cost of a query is $|\mathcal{S}|$.

The 1-MSTAT oracle, in the above general version, was defined by Feldman et al. (2015a). Related versions can also be found in Feldman et al. (2017); Brennan et al. (2021). This oracle corresponds to a single query to a distribution where the output complexity is limited to $L$ values.

A statistical algorithm's complexity is measured by the number of queries multiplied by the cost per query. This model is broad enough to capture many standard approaches, including EM, MCMC, method of moments, simulated annealing, and convex optimization.

**Distinguishing Distributions.** We study the following *distributional decision problem* $\mathcal{B}(\mathcal{D}, D)$: Given a reference distribution $D$ over a domain $X$ and a set $\mathcal{D}$ of alternative (or planted) distributions, decide whether an unknown distribution $D' \in \{D\} \cup \mathcal{D}$ is equal to $D$ or belongs to $\mathcal{D}$, using $t > 0$ samples from $D'$. In many applications (e.g., in $k$-SAT), one sets $X = X_k$, takes $\mathcal{D}$ to be a collection of distributions with planted solutions, and $D$ as the uniform distribution over $X_k$.

For any function $h : X \to \mathbb{R}$, with $\|h\|_D = \sqrt{\mathbb{E}_D[h^2]}$, the *discrimination norm* relative to $D$ is

$$\kappa_2(\mathcal{D}, D) = \max_{h:\ \|h\|_D = 1} \mathbb{E}_{D' \sim \mathcal{D}}\Big[\ \big|\mathbb{E}_{D'}[h] - \mathbb{E}_D[h]\big|\Big].$$

This norm measures the average effectiveness of the query function $h$ in distinguishing distributions from $\mathcal{D}$ from the reference $D$. In particular, as shown in Feldman et al. (2015a), if $\kappa_2(\mathcal{D}, D) = \kappa$, then a single query to $\mathrm{VSTAT}(1/(3\kappa^2))$ is insufficient to differentiate a typical $D' \in \mathcal{D}$ from $D$.

**Statistical Dimension.** The *statistical dimension* $\mathrm{SDN}(\mathcal{B}(\mathcal{D}, D), \kappa)$ is defined as the largest integer $d$ for which there exists a finite set $\mathcal{D}_D \subseteq \mathcal{D}$ such that every subset $\mathcal{D}' \subseteq \mathcal{D}_D$ with $|\mathcal{D}'| \geq |\mathcal{D}_D|/d$ satisfies $\kappa_2(\mathcal{D}', D) \leq \kappa$. Intuitively, a high statistical dimension indicates that even large subsets of $\mathcal{D}_D$ are hard to distinguish from $D$.

This concept immediately leads to lower bounds on the query complexity of statistical algorithms. In particular, (Feldman et al., 2015a, Theorem 3.4) shows that if $\mathrm{SDN}(\mathcal{B}(\mathcal{D}, D), \kappa) = d$ and $L \geq 2$, then any randomized statistical algorithm that solves $\mathcal{B}(\mathcal{D}, D)$ with probability at least $2/3$ requires:

- $\Omega(d/L)$ queries to *MVSTAT*$(L, 1/(12\kappa^2 L))$, and
- $\Omega\big(\min\{d, 1/\kappa^2\}/L\big)$ queries to *1-MSTAT*$(L)$.

## 3. Quiet planting of multiple solutions

Our main result in this paper is a method for generating random $k$-SAT instances with quietly planted solutions that are "hard" to distinguish from uniformly random instances, where the notion of hardness is formalized by studying the problem in the statistical query framework and characterizing its statistical dimension. Specifically, we study algorithms that only have indirect access to the underlying distribution of the clauses using a statistical oracle, and provide a lower bound on the number of queries they need to distinguish the underlying distribution from a uniformly random one.

Our approach is novel in two key aspects. First, it enables the planting of multiple solutions. Second, it allows the planted solutions to exhibit arbitrary geometry in $\mathbb{F}_2^n$. More precisely, by *arbitrary geometry*, we mean that for any set $S \subseteq \mathbb{F}_2^n$, our method constructs a random instance whose solution set contains a subset $S' \subseteq \mathbb{F}_2^n$ such that there exists a bijection $\phi : S \to S'$ satisfying $\phi(x) \oplus \phi(y) = x \oplus y$ for all $x, y \in S$. In other words, the coordinate-wise XOR structure among the original vectors in $S$ is preserved under the mapping to the solution set.

It is noteworthy that the concept of arbitrary geometry is crucial for effectively testing SAT solvers. Quiet planting is primarily motivated by the need to evaluate the robustness of these solvers; thus, in scenarios with nearly identical sets of solutions, the solvers may not be rigorously tested. Additionally, certain heuristics could potentially compromise the integrity of the test cases. We also remark that planting an exact set of solutions into a random instance is, to some extent, an infeasible task, as it can be defeated by a statistical algorithm that explicitly checks for these very specific solutions. Please note that the trivial way of planting a solution by randomly taking satisfying clauses is vulnerable to statistical query algorithms because there is a bias in the number of times that a variable appears as true or false regarding its value in the chosen assignment for planting.

In this paper, we follow the statistical query model used by Feldman et al. (2015a) for studying planted $k$-SAT instances. Their work showed that the primary parameter measuring the hardness of the testing problem for distinguishing planted instances from uniformly random ones is the *statistical dimension* of the corresponding testing problem. Here, the statistical dimension, which we will also denote with *SDN*, serves as a proxy for the hardness of the problem, with higher dimensions

corresponding to harder problems (see section 2 for a formal definition) This is demonstrated in Feldman et al. (2015a), where the authors prove that bounds on statistical dimension imply lower bounds on the query complexity of algorithms for the problem for various statistical query oracles. Given these results, we focus on specifying distributions for which the statistical dimension of this testing problem is high. Formally, we prove the following result (see the appendix for the formal proof).

**Theorem 1** *Assume that there exists a binary linear code $[k, t, r]$ and a variable $L < 2^t$. Let $\mathcal{A}$ be an arbitrary set of assignments of size at most $L$. A set $\mathcal{D}$ of distributions over clauses exists such that*

1. *The statistical dimension of the testing problem $\mathcal{B}(\mathcal{D},\mathcal{U})$ of distinguishing a $\mathcal{U}$ (Uniform distribution of $k$-clauses) from a distribution $\mathcal{Q}$ chosen uniformly at random from $\mathcal{D}$ satisfies the bound for any $q \geq 1$*

$$\text{SDN}\left(\mathcal{B}(\mathcal{D},\mathcal{U}), \frac{c(\log q)^{r/2}}{n^{r/2}}\right) \geq q, \tag{1}$$

   *where c is a constant and SDN refers to the statistical dimension of the problem (see Section 2)*

2. *For any $\mathcal{Q} \in \mathcal{D}$, there exists $\tau \in \{0,1\}^n$ such that any clause $C \sim \mathcal{Q}$ satisfies all of the assignments $\sigma \oplus \tau$ for $\sigma \in \mathcal{A}$ with probability 1, where $\oplus$ denotes the coordinate-wise XOR between two vectors.*

Intuitively, the bound for SDN suggests that any randomly chosen distribution from $\mathcal{D}$ remains indistinguishable from the uniform distribution over $k$-clauses up to $n^{r/2}$ clauses. In the second point, we focus on the arbitrariness of the geometry of our solutions. Ideally, for any $\mathcal{A}$, we would want to generate instances where the solutions are exactly $\mathcal{A}$ and are hard to distinguish from random for all algorithms. However, if $\mathcal{A}$ is prespecified, this is not possible, as an algorithm could simply check for the solutions in $\mathcal{A}$. Therefore, we adopt a modified approach, where instead of $\mathcal{A}$, the solution is a random XOR-rotation of $\mathcal{A}$. Intuitively, the geometry of the solution set remains unchanged, as all solutions are transformed using the same rotation. We emphasize that $\tau$ is fixed once per distribution $\mathcal{Q}$, not per clause: a single $\tau$ is chosen for $\mathcal{Q}$, and all clauses of the instance are then sampled i.i.d. from this fixed $\mathcal{Q}$, so the planted set $\{\sigma \oplus \tau : \sigma \in \mathcal{A}\}$ is the global set of satisfying assignments.

As an immediate result and evidence of tightness, we can achieve the bound provided in Feldman et al. (2015a) by using the *repetition code* as a linear code. Since a repetition code of length $k$ is a $[k, 1, k]$ binary linear code, we can apply the above theorem to conclude that planting one assignment is quiet up to $n^{k/2}$ clauses. Given the structure of the repetition code, planting a solution in our method is equivalent to planting one in a $k$-XOR-SAT instance.

Another result can be achieved by applying Gilbert-Varshamov bound (Gilbert, 1952; Varshamov, 1957) which proved the existence of binary linear code of $[k, (1 - H(\delta))k, \delta k]$ for any $\delta < 1/2$ ($H(.)$ denotes the binary entropy function). Consequently, planting less than $2^{(1-H(\delta))k}$ solutions is quiet up to $n^{\delta k/2}$ clauses. This means we can plant an exponential number of solutions with arbitrary geometry.

As an immediate corollary, using known reductions in the literature, the above theorem implies statistical query lower bounds for various statistical oracles. We refer to Section 2 for a formal

definition of these oracles and an overview of the reductions we use We note that the bounds we obtain here correspond to the hardness bounds for distinguishing random $r$-XOR-SAT from uniform in the same frameworks. As previously noted by Feldman et al. (2015a) and Kothari et al. (2017), these bounds suggest that any efficient algorithm for detecting the existence of our planted solutions requires at least $\Omega(n^{r/2})$ clauses.

Here, we elaborate on the most challenging part of our findings, which is the ability to plant many assignments of arbitrary geometry. In particular, our method allows for planted solutions of arbitrary Hamming distance, where Hamming distance refers to the number of variables for which two solutions differ.

We first recap the state-of-the-art approach for generating quiet planting instances of $k$-SAT for a single solution. Let us view each clause $C$ as a tuple $(I, x)$ where $I$ is an ordered set of $k$ variables and $x \in \{0,1\}^k$ is a *sign pattern* that shows whether a variable or its negation appears in the clause. We will often use bold letters to emphasize that a vector or a set is random (e.g., $\mathbf{x}$ vs $x$). Prior works construct a distribution over the set of all feasible clauses, denoted $X_k$, by combining two independent distributions. The first distribution, denoted by $I_k$, uniformly chooses an ordered set $\mathbf{I}$ of $k$ variables with no repetitions. The second one, $Q$ is defined over the set of sign patterns $\{0,1\}^k$, with the important property that the all ones vector $(1,\ldots,1)$ has probability 0 under $Q$. If the goal is to plant a reference assignment $\sigma$, a random clause is generated according to the distribution $(\mathbf{I}, \sigma[\mathbf{I}] \oplus \mathbf{x})$ where $\mathbf{I} \sim I_K$, $\mathbf{x} \sim Q$. Since $(1,\ldots,1)$ is not sampled under $Q$, this ensures that the clause is always satisfied under $\sigma$. While this approach allows us to construct distributions hard to distinguish from the uniform random distribution in the statistical query model (cf. Theorem 3.1 in Feldman et al. (2015a)), it inherently cannot be used to plant another solution of a large Hamming distance to $\sigma$, as we note below.

**Proposition 2** *For any $Q$, if $\sigma$ is a planted solution in the above construction, there is no $\sigma' \in \{0,1\}^n$ having Hamming distance $k$ to $\sigma$ that is satisfied by every clause that belongs to the distribution specified by the above construction.*

**Proof** [Proof of Proposition 2] Assume that such $\sigma'$ exists. Let $x \in \{0,1\}^k$ be any element from the support of $Q$. Since Hamming distance of $\sigma$ and $\sigma'$ is $k$, thus there exists an ordered set $I$ of different literals such that $\sigma(I[i]) = \sigma'(I[i])$ if and only if $x[i] = 1$. Since $x$ was chosen from the support, the clause $(\mathbf{I}, \sigma[I] \oplus x)$ has a positive probability of appearing in the random formula. However, based on the specific choice of the variables $I$, if we XOR the sign patterns in this clause with the corresponding assignment under $\sigma'$, we obtain $\sigma'[I] \oplus \sigma[I] \oplus x = (1,\ldots,1)$ . This means that no variable in the clause is satisfied under $\sigma'$. contradicting the assumption made at the beginning of the proof. ■

In contrast, in our approach for quiet planting, we deviate from using one distribution $Q$ in conjunction with all possible ordered sets of variables $I$. Rather, we allow the distribution of sign patterns to depend on $I$. This allows for better alignment of the sign patterns to the set of variables of a clause, which in turn allows us to plant assignments of arbitrary geometry, overcoming the counterexample of Proposition 2. It also allows for the quiet planting of more than one solution. But at the same time, it poses new challenges of characterizing properties that would yield lower bounds on the statistical dimension of such a distribution and, in consequence, using known reductions, would imply the hardness of distinguishing this distribution from the uniform one in the statistical queries model.

## 4. Universal statistical lower bound for $(r-1)$-wise uniform distribution

Firstly, we discuss the property of a distribution over $k$-SAT clauses, called $(r-1)$-*wise uniformness*, that is responsible for formalizing hardness. This property assures that when fewer than $r$ variables are considered, the marginal distribution on these variables is the uniform distribution. We refer the reader to Appendix B containing Definition 12 for a formal definition of this property. Although our definition is based on the same notion in Feldman et al. (2015a) and Kothari et al. (2017) (in the former the notion appears as "distribution complexity"), our definition generalizes the previous works by allowing for dependencies between the variables of a clause and possible negations added to the variables. This change is crucial to present later a distribution that every clause is satisfied by a fixed $> 1$ number of solutions, which, as noted above, can not be done without the relaxed independence assumption.

As the main result of this part, we prove that $(r-1)$-wise uniform is a property that implies a large statistical dimension of a distribution. As a large statistical dimension of a distribution yields lower bounds on the query complexity of statistical algorithms, this property is also ultimately responsible for the hardness of distinguishing the distributions constructed in this paper from the uniform random one. In conclusion, we prove the following hardness result (See Appendix C for the formal proof). This theorem generalizes Theorem 3.5 in Feldman et al. (2015a).

**Theorem 3** *For every $(r-1)$-wise uniform distribution $\mathcal{Q}$ over $X_k$, there exists a constant $c > 0$, depending on $k$, such that, for any $q \geq 1$,*

$$\mathrm{SDN}\left(\mathcal{B}(\mathcal{D}_{\mathcal{Q}}, \mathcal{U}), \frac{c(\log q)^{r/2}}{n^{r/2}}\right) \geq q$$

*where $\mathcal{U}$ is a uniform distribution over $X_k$ and $\mathcal{D}_{\mathcal{Q}} := \{\mathcal{Q}_\tau : \tau \in \{-1,1\}^n\}$ denote the set of all XOR-rotation of distribution $\mathcal{Q}$.*

Combining the above theorem with the reduction results from Feldman et al. (2015a) yields hardness for statistical queries using up to $n^{r/2}$ queries (see Section 2).

## 5. Construction of $(r-1)$-wise uniform distributions

The second part of our approach involves constructing an $(r-1)$-wise uniform distribution $\mathcal{Q}$ over the set of clauses that satisfies a given set of assignments, called $\mathcal{A}$. Let us fix an ordered set of $k$ variables $I$ in the clause $(\mathbf{I} = I, \mathbf{x}) \sim \mathcal{Q}$ and let $Q_I$ denote the distribution of sign patterns $\mathbf{x} \in \{0,1\}^k$ conditioned on $I$. The standard approach in prior work for ensuring a fixed assignment $\sigma \in \{0,1\}$ satisfied $(\mathbf{I}, \mathbf{x})$ is to let $Q_I$ be the uniformly random distribution over all these elements of $\{0,1\}^k$ that agree with $\sigma[I]$ on an odd number of coordinates. Intuitively, such constructed distribution is $(k-1)$-wise uniform because sampling from $Q_I$ is equivalent to sampling any $k-1$ coordinates uniformly at random in an arbitrary order and choosing the last one to ensure the number of agreements with $\sigma[I]$ is odd. By the last property, any $\mathbf{x}$ sampled from this distribution is satisfied by $\sigma$ because the number of coordinates agreeing with $\sigma$ cannot be 0.

To generalize this idea into multiple solutions $\sigma_1, \ldots, \sigma_t$, a natural approach is to select multiple subsets of $[k]$ as $S_1, \ldots, S_t$ and require that for each $i \in [t]$, the sign pattern $\mathbf{x}$ agrees with $\sigma_i$ on an odd number of coordinates of $S_i$. With such an $\mathbf{x}$, it is clear that all assignments are satisfied. However, even if we pick $\mathbf{x}$ uniformly at random from all such sign patterns, it is not clear whether

the $(r-1)$-wise uniformness property holds; indeed, the sign pattern $x[S_i]$ is definitely not uniform because it only takes half of all possible values. Therefore, the distribution cannot be $(r-1)$-wise uniform for $r \geq \min |S_i|$.

To solve the above issue, a natural idea is to choose $|S_i| \geq r$; however, this alone is not sufficient. For example, suppose $S_1$ and $S_2$ both have size $\geq r$ but differ by only one coordinate: $s_1 \in S_1 \setminus S_2$ and $s_2 \in S_2 \setminus S_1$. Assume $\sigma_1$ and $\sigma_2$ agree on $S_1 \cap S_2$. Since each $S_i$ must have an odd number of agreements with $\sigma_i$, and $\sigma_1, \sigma_2$ agree on the intersection, it is impossible for $x$ to agree with $\sigma_1$ on $s_1$ but disagree with $\sigma_2$ on $s_2$. Specifically, agreement on $s_1$ implies an even number of agreements on $S_1 \cap S_2$, which in turn implies agreement on $s_2$. Thus, the distribution is not 2-wise uniform, as the set $S' = \{s_1, s_2\}$ does not take all possible values.

Intuitively, the issue in the above example is excessive dependence among the sets $S_i$. We show that avoiding dependencies of size $r$ when choosing the $S_i$ is essentially equivalent to designing a linear code. Specifically, suppose there exists a $[k, t, r]$ binary linear code with generator matrix $V \in \{0,1\}^{t \times k}$ (a code with $k$-bit codewords, message length $t$, and minimum Hamming distance $r$). We use this code to construct a distribution $Q_I$ that is $(r-1)$-wise uniform and has $t$ distinct solutions by taking each $S_i$ to be the set of 1-coordinates in the $i^{\text{th}}$ row of $V$.

We further propose a novel idea to extend the number of solutions from $t$ to up to $L = 2^t - 1$. Our main insight is that for each row of $V$, instead of satisfying only one $\sigma_i$, we can satisfy at least half of the remaining solutions. This allows us to divide the number of unsatisfied solutions in each row, ensuring that all solutions are satisfied since $L \leq 2^t - 1$. Figure 1 illustrates the use of linear code to achieve a $(r-1)$-wise uniform distribution, satisfying $2^t - 1$ solutions.

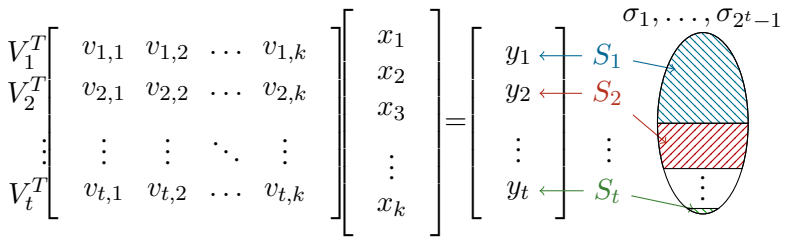


Figure 1: This figure illustrates how a $[k, t, r]$ linear code results in a $(r-1)$-wise uniform distribution $Q_I$. The left matrix shows a code matrix $V$. For each row of $V$ denoted as $V_i^T$, the set of 1 coordinates of $V_i^T$ is $S_i$. For $S_i$, we can ensure that $x$ agrees with half of the remaining solutions by setting $y_i$ to a proper value. It is shown in the figure that the first at least half of solutions are satisfied by $S_1$, the next at least half of the remaining solutions are satisfied by $S_2$, and so on. Therefore, this approach supports up to $2^t - 1$ solutions.

Combining these two, we present the following theorem (with the proof in Appendix B):

**Theorem 4** *Suppose a $[k, t, r]$ linear code exists. Then for any set $I$ of $k$ variables and any collection $\mathcal{A}$ of assignments with $|\mathcal{A}| \leq 2^t - 1$, there exists a distribution $\mathcal{Q}_I$ over sign patterns on $I$ such that:*

*(i) For every sign pattern $x$ in the support of $\mathcal{Q}_I$, the clause $C = (I, x)$ is satisfiable by some assignment in $\mathcal{A}$, and*

*(ii) If $x \sim \mathcal{Q}_I$, then the marginal distribution on any subset of $r-1$ coordinates of $x$ is uniform.*

We can now apply any linear code to balance the number of possible assignments and the complexity of the distribution. The Gilbert-Varshamov bound for linear codes leads to the following bound.

**Theorem 5** *For a given $k$, $\delta_0$, where $0 \leq \delta_0 \leq \frac{1}{2}$, and a given set of assignments $\mathcal{A}$ of size at most $2^{k(1-H(\delta_0))}-1$, there exists a $(\delta_0 k-1)$-wise uniform distribution $\mathcal{Q}$ over $k$-clauses, where $H(x) = -x\log_2(x) - (1-x)\log_2(1-x)$).*

This result shows that a linear reduction in complexity allows an exponential expansion in the set of assignments. Although this theorem does not match the bound provided in Feldman et al. (2015a), our linear code planting technique allows planting a solution in up to $n^{k/2}$ clauses using the *repetition code*, whose generator vector is $V = [1, 1, \ldots, 1]$. The equation $Vx = y$ holds when the parity of $x$ matches $y$, equivalent to the XOR-SAT planting method in Feldman et al. (2015a).

## 6. Worst-case analysis of SAT instances with multiple solutions

Lastly, we introduce a framework for assessing the hardness of SAT instances with numerous solutions, focusing on scenarios where the SAT problem is either unsolvable or has a predetermined minimum number of solutions. This is achieved by exponentially increasing the number of potential solutions and leveraging the Exponential Time Hypothesis (ETH), which is particularly effective in tackling the open problem posed by Hsieh, Mohanty, and Xu Hsieh et al. (2022) (CCC'22).

**Theorem 6** *Assuming the Exponential Time Hypothesis (ETH), there is no polynomial-time algorithm for solving SAT$(n, 2^{n-\log(n)^{1+c}})$ for any $c > 0$, even when it is guaranteed that the input is (7/8 + $\epsilon$)-satisfiable. Note that SAT$(n, s)$ is the class of SAT instances with $n$ variables that are either unsatisfiable or have at least $s$ solutions.*

We further introduce a randomized algorithm for solving *SAT*$(n, 2^{n-\log(n)})$. The algorithm randomly selects assignments until a satisfying assignment is found or a predefined threshold of attempts is reached. If a solution exists, each random assignment satisfies the instance with probability $\frac{1}{n}$. By checking multiple random assignments, if we find a satisfying assignment, we conclude that the instance is satisfiable; otherwise, whp, it is unsatisfiable. Proofs are provided in Appendix D.

## 7. Related work

Planting has recently been the subject of research in various problems, such as planting structures in graphs like cliques (Buhai et al., 2023; Kothari et al., 2023; Bandeira et al., 2021) or planting solutions for satisfiability problems (Guruswami et al., 2023). In the context of planted $k$-SAT, the development of hard-to-solve, quietly planted instances has motivated researchers from diverse fields, including theoretical computer science and statistical physics. A common way to test SAT solvers is to use random instances—often generated by selecting an assignment $\sigma$ and randomly choosing clauses, rejecting those unsatisfied by $\sigma$. These instances are typically easy for local search approaches. Prior works Jia et al. (2005); Barthel et al. (2002), experimentally studied generating quietly planted instances that are empirically challenging for such local methods or for other known

$k$-SAT solvers. Another line of work has tried to build a theory for the hardness of instances with a single planted solution Krzakala and Zdeborová (2009); Krzakala et al. (2014); Feldman et al. (2015a). The work of Blocki et al. (2017) extends this concept beyond predicates over $\mathbb{F}_2$.

Over the years, constructing planting distributions for the $k$-SAT problem with *multiple* solutions has been explored experimentally Liu et al. (2015); Zhao et al. (2023) and theoretically for up to two solutions or in hypothesis testing settings Achlioptas et al. (2004); Berthet and Ellenberg (2019). However, our work is the first to plant more than two solutions of arbitrary geometry and establish theoretical hardness for this class of instances.

Other research efforts have focused on algorithms for the planted $k$-SAT problem. Spectral methods were studied in Flaxman (2003). Bogdanov and Qiao Bogdanov and Qiao (2009) demonstrated the applicability of a Semidefinite Programming (SDP)-based algorithm by Charikar and Wirth Charikar and Wirth (2004), which efficiently identifies the planted assignment for predicates lacking pairwise independence using only $O(n)$ evaluations. Their approach also extends to recovering inputs for $(r-1)$-wise (but not $r$-wise) independent predicates with $O(n^{r/2})$ evaluations Applebaum (2016). These results enhance the toolkit for recovering planted assignments and deepen understanding of algorithmic strategies for planted $k$-SAT. The analysis also sheds light on challenges posed by planted instances, with implications for complexity theory, cryptography Goldreich (2000), and algorithms for optimization problems with planted solutions Feldman et al. (2013). A key milestone by Feldman et al. (2015b) unified prior efforts under a model covering specific planted satisfiability distributions. Their algorithm recovers planted solutions in both the stochastic block model and planted CSPs via a common generalization using random bipartite graphs.

Finally, many works have developed the theory of hardness under other plausible conjectures in the complexity theory. The result of Kothari et al. (2017) gives a lower bound on the refutation of any CSP problem whose predicates are supported by $(r-1)$-wise uniform distributions for the class of sum of squares algorithms. This result is asymptotically tight due to the upper bounds given in Allen et al. (2015); Raghavendra et al. (2017).

## Acknowledgments

This work is partially supported by DARPA QuICC, DARPA expMath, ONR MURI 2024 award on Algorithms, Learning, and Game Theory, Army-Research Laboratory (ARL) grant W911NF2410052, NSF AF:Small grants 2218678, 2114269, 2347322.

## References

Dimitris Achlioptas, Haixia Jia, and Cristopher Moore. Hiding satisfying assignments: Two are better than one. In Deborah L. McGuinness and George Ferguson, editors, *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, San Jose, California, USA, July 25-29, 2004*, pages 131–136. AAAI Press / The MIT Press, 2004. URL http://www.aaai.org/Library/AAAI/2004/aaai04-021.php.

Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to refute a random CSP. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, October 17-20, 2015*, pages 689–708. IEEE Computer Society, 2015. doi: 10.1109/FOCS.2015.48. URL https://doi.org/10.1109/FOCS.2015.48.

Benny Applebaum. Cryptographic hardness of random local functions - survey. *Comput. Complex.*, 25(3):667–722, 2016. doi: 10.1007/S00037-015-0121-8. URL https://doi.org/10.1007/s00037-015-0121-8.

Afonso S. Bandeira, Dmitriy Kunisky, and Alexander S. Wein. Computational hardness of certifying bounds on constrained PCA problems. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, Seattle, Washington, USA, January 12-14, 2020*, volume 151 of *LIPIcs*, pages 78:1–78:29. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi: 10.4230/LIPICS.ITCS.2020.78. URL https://doi.org/10.4230/LIPIcs.ITCS.2020.78.

Afonso S. Bandeira, Jess Banks, Dmitriy Kunisky, Cristopher Moore, and Alexander S. Wein. Spectral planting and the hardness of refuting cuts, colorability, and communities in random graphs. In Mikhail Belkin and Samory Kpotufe, editors, *The Thirty Fourth Annual Conference on Learning Theory, COLT 2021, Boulder, Colorado, USA, August 15-19, 2021*, volume 134 of *Proceedings of Machine Learning Research*, pages 410–473. PMLR, 2021. URL http://proceedings.mlr.press/v134/bandeira21a.html.

Jess Banks, Sidhanth Mohanty, and Prasad Raghavendra. Local statistics, semidefinite programming, and community detection. In Dániel Marx, editor, *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1298–1316. ACM/SIAM, 2021. doi: 10.1137/1.9781611976465.79. URL https://doi.org/10.1137/1.9781611976465.79.

Wolfgang Barthel, Alexander K Hartmann, Michele Leone, Federico Ricci-Tersenghi, Martin Weigt, and Riccardo Zecchina. Hiding solutions in random satisfiability problems: A statistical mechanics approach. *Physical review letters*, 88(18):188701, 2002.

Elwyn R. Berlekamp. *Algebraic coding theory*. McGraw-Hill series in systems science. McGraw-Hill, 1968. ISBN 0070049033. URL https://www.worldcat.org/oclc/00256659.

Quentin Berthet and Jordan S. Ellenberg. Detection of planted solutions for flat satisfiability problems. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, Naha, Okinawa, Japan, April 16-18, 2019*, volume 89 of *Proceedings of Machine Learning Research*, pages 1303–1312. PMLR, 2019. URL http://proceedings.mlr.press/v89/berthet19b.html.

Jeremiah Blocki, Manuel Blum, Anupam Datta, and Santosh S. Vempala. Towards human computable passwords. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, Berkeley, CA, USA, January 9-11, 2017*, volume 67 of *LIPIcs*, pages 10:1–10:47. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi: 10.4230/LIPICS.ITCS.2017.10. URL https://doi.org/10.4230/LIPIcs.ITCS.2017.10.

Andrej Bogdanov and Youming Qiao. On the security of goldreich's one-way function. In Irit Dinur, Klaus Jansen, Joseph Naor, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, volume 5687 of *Lecture Notes in Computer Science*, pages

392–405. Springer, 2009. doi: 10.1007/978-3-642-03685-9_30. URL https://doi.org/10.1007/978-3-642-03685-9_30.

Matthew S. Brennan, Guy Bresler, Samuel B. Hopkins, Jerry Li, and Tselil Schramm. Statistical query algorithms and low degree tests are almost equivalent. In Mikhail Belkin and Samory Kpotufe, editors, *The Thirty Fourth Annual Conference on Learning Theory, COLT 2021, Boulder, Colorado, USA, August 15-19, 2021*, volume 134 of *Proceedings of Machine Learning Research*, page 774. PMLR, 2021. URL http://proceedings.mlr.press/v134/brennan21a.html.

Rares-Darius Buhai, Pravesh K. Kothari, and David Steurer. Algorithms approaching the threshold for semi-random planted clique. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1918–1926. ACM, 2023. doi: 10.1145/3564246.3585184. URL https://doi.org/10.1145/3564246.3585184.

Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendieck's inequality. In *IEEE 45th Annual Symposium on Foundations of Computer Science ,FOCS 2004, Rome, Italy, October 17-19, 2004*, pages 54–60. IEEE Computer Society, 2004. doi: 10.1109/FOCS.2004.39. URL https://doi.org/10.1109/FOCS.2004.39.

Jian Ding, Allan Sly, and Nike Sun. Proof of the satisfiability conjecture for large k. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM SIGACT Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 59–68. ACM, 2015. doi: 10.1145/2746539.2746619. URL https://doi.org/10.1145/2746539.2746619.

Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh S. Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Proceedings of the Forty-Fifth Annual ACM SIGACT Symposium on Theory of Computing Conference, STOC 2013, Palo Alto, CA, USA, June 1-4, 2013*, pages 655–664. ACM, 2013. doi: 10.1145/2488608.2488692. URL https://doi.org/10.1145/2488608.2488692.

Vitaly Feldman, Will Perkins, and Santosh S. Vempala. On the complexity of random satisfiability problems with planted solutions. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM SIGACT Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 77–86. ACM, 2015a. doi: 10.1145/2746539.2746577. URL https://doi.org/10.1145/2746539.2746577.

Vitaly Feldman, Will Perkins, and Santosh S. Vempala. Subsampled power iteration: a unified algorithm for block models and planted csp's. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2836–2844, 2015b. URL https://proceedings.neurips.cc/paper/2015/hash/9597353e41e6957b5e7aa79214fcb256-Abstract.html.

Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh S. Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. *J. ACM*, 64(2):8:1–8:37, 2017. doi: 10.1145/3046674. URL https://doi.org/10.1145/3046674.

Vitaly Feldman, Will Perkins, and Santosh S. Vempala. On the complexity of random satisfiability problems with planted solutions. *SIAM J. Comput.*, 47(4):1294–1338, 2018. doi: 10.1137/16M1078471. URL https://doi.org/10.1137/16M1078471.

Abraham Flaxman. A spectral technique for random satisfiable 3cnf formulas. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2003, Baltimore, Maryland, USA, January 12-14, 2003*, pages 357–363. ACM/SIAM, 2003. URL http://dl.acm.org/citation.cfm?id=644108.644166.

Edgar N Gilbert. A comparison of signalling alphabets. *The Bell system technical journal*, 31(3): 504–522, 1952.

Oded Goldreich. Candidate one-way functions based on expander graphs. *IACR Cryptol. ePrint Arch.*, page 63, 2000. URL http://eprint.iacr.org/2000/063.

Venkatesan Guruswami, Jun-Ting Hsieh, Pravesh K. Kothari, and Peter Manohar. Efficient algorithms for semirandom planted csps at the refutation threshold. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 307–327. IEEE, 2023. doi: 10.1109/FOCS57990.2023.00026. URL https://doi.org/10.1109/FOCS57990.2023.00026.

Jun-Ting Hsieh, Sidhanth Mohanty, and Jeff Xu. Certifying solution geometry in random csps: Counts, clusters and balance. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, Philadelphia, PA, USA, July 20-23, 2022*, volume 234 of *LIPIcs*, pages 11:1–11:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi: 10.4230/LIPICS.CCC.2022.11. URL https://doi.org/10.4230/LIPIcs.CCC.2022.11.

Haixia Jia, Cristopher Moore, and Doug Strain. Generating hard satisfiable formulas by hiding solutions deceptively. In Manuela M. Veloso and Subbarao Kambhampati, editors, *Proceedings of the Twentieth National Conference on Artificial Intelligence, Seventeenth Innovative Applications of Artificial Intelligence Conference, Pittsburgh, Pennsylvania, USA, July 9-13, 2005*, pages 384–389. AAAI Press / The MIT Press, 2005. URL http://www.aaai.org/Library/AAAI/2005/aaai05-061.php.

Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *Proceedings of the Twenty-Fifth Annual ACM SIGACT Symposium on Theory of Computing, STOC 1993, San Diego, CA, USA, May 16-18, 1993*, pages 392–401. ACM, 1993. doi: 10.1145/167088.167200. URL https://doi.org/10.1145/167088.167200.

Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998. doi: 10.1145/293347.293351. URL https://doi.org/10.1145/293347.293351.

Pravesh Kothari, Santosh S. Vempala, Alexander S. Wein, and Jeff Xu. Is planted coloring easier than planted clique? In Gergely Neu and Lorenzo Rosasco, editors, *The Thirty Sixth Annual Conference on Learning Theory, COLT 2023, Bangalore, India, July 12-15, 2023*, volume 195 of *Proceedings of Machine Learning Research*, pages 5343–5372. PMLR, 2023. URL https://proceedings.mlr.press/v195/kothari23a.html.

Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the Forty-Ninth Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 132–145. ACM, 2017. doi: 10.1145/3055399.3055485. URL https://doi.org/10.1145/3055399.3055485.

Florent Krzakala and Lenka Zdeborová. Hiding quiet solutions in random constraint satisfaction problems. *CoRR*, abs/0901.2130, 2009. URL http://arxiv.org/abs/0901.2130.

Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Reweighted belief propagation and quiet planting for random K-SAT. *J. Satisf. Boolean Model. Comput.*, 8(3/4):149–171, 2014. doi: 10.3233/SAT190096. URL https://doi.org/10.3233/sat190096.

Ran Liu, Wenjian Luo, and Lihua Yue. Hiding multiple solutions in a hard 3-sat formula. *Data Knowl. Eng.*, 100:1–18, 2015. doi: 10.1016/J.DATAK.2015.09.003. URL https://doi.org/10.1016/j.datak.2015.09.003.

Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random csps below the spectral threshold. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the Forty-Ninth Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 121–131. ACM, 2017. doi: 10.1145/3055399.3055417. URL https://doi.org/10.1145/3055399.3055417.

Rom Rubenovich Varshamov. Estimate of the number of signals in error correcting codes. *Docklady Akad. Nauk, SSSR*, 117:739–741, 1957.

Alexander S. Wein. Average-case complexity of tensor decomposition for low-degree polynomials. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the Fifty-fifth Annual ACM SIGACT Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1685–1698. ACM, 2023. doi: 10.1145/3564246.3585232. URL https://doi.org/10.1145/3564246.3585232.

Dongdong Zhao, Lei Liao, Wenjian Luo, Jianwen Xiang, Hao Jiang, and Xiaoyi Hu. Generating random SAT instances: Multiple solutions could be predefined and deeply hidden. *J. Artif. Intell. Res.*, 76:435–470, 2023. doi: 10.1613/JAIR.1.13909. URL https://doi.org/10.1613/jair.1.13909.

## Appendix A. Prelimineries - Linear Codes

Linear codes, utilized in information and coding theory, are error-correcting codes crucial for dependable data transmission over noisy channels. Here, we present essential definitions concerning linear codes for future reference, an exhaustive summary can be found in Berlekamp (1968).

The formal definition of a code in information theory can be summarized as follows.

**Definition 7 (Code)** *An error-correcting code $\mathcal{C}$ of length $n$ over a finite alphabet $\Sigma$ is a subset of $\Sigma^n$. The elements of $\mathcal{C}$ are termed codewords, and the length $n$ of the codewords in $\mathcal{C}$ is the block length.*

*The alphabet of $\mathcal{C}$ is $\Sigma$, and if $|\Sigma| = q$, $\mathcal{C}$ is referred to as a $q$-ary code. For $q = 2$, it is a binary code.*

*A code is associated with an encoding map $E$, which maps the message set $M$ of size $|\mathcal{C}|$ to codewords in $\Sigma^n$. The code is then the image of this encoding map.*

Here is the definition of the rate of a code:

**Definition 8 (Rate)** *The rate of a code $\mathcal{C} \subseteq \Sigma^n$, denoted $R(\mathcal{C})$, is defined by*

$$R(\mathcal{C}) = \frac{\log |\mathcal{C}|}{n \log |\Sigma|}.$$

The dimension of $\mathcal{C}$ is defined as $\log_{|\Sigma|} |\mathcal{C}|$. It is noteworthy that a $q$-ary code of dimension $l$ comprises $q^l$ codewords. In essence, $R(\mathcal{C})$ represents the amount of non-redundant information per bit in the codewords of $\mathcal{C}$.

Next, we define the distance of a code as follows. We start by providing the definition of Hamming distance.

**Definition 9 (Hamming distance)** *The Hamming distance between two strings $x$ and $y$ of equal length over a finite alphabet $\Sigma$, denoted $\Delta(x, y)$, counts the positions where the strings differ, i.e., $\Delta(x, y) = |\{i \,|\, x_i \neq y_i\}|$. The fractional Hamming distance or relative distance between $x$ and $y \in \Sigma^n$ is expressed as $\delta(x, y) = \frac{\Delta(x,y)}{n}$, ranging from 0 to 1 indicates the proportion of symbols that differ between the two strings, providing a normalized measure of their dissimilarity.*

**Definition 10 (Distance)** *The distance of a code $\mathcal{C}$, denoted $\Delta(\mathcal{C})$, is the minimum Hamming distance between two distinct codewords of $\mathcal{C}$, given by:*

$$\Delta(\mathcal{C}) = \min_{\substack{\mathcal{C}_1, \mathcal{C}_2 \in \mathcal{C} \\ \mathcal{C}_1 \neq \mathcal{C}_2}} \Delta(\mathcal{C}_1, \mathcal{C}_2)$$

*In particular, for every pair of distinct codewords in $\mathcal{C}$, the Hamming distance between them is at least $\Delta(\mathcal{C})$. The relative distance of $\mathcal{C}$, denoted $\delta(\mathcal{C})$, is defined as the normalized quantity $\frac{\Delta(\mathcal{C})}{n}$, where $n$ is the block length of $C$. Thus, any two codewords of $\mathcal{C}$ differ in at least a fraction $\delta(\mathcal{C})$ of positions.*

Finally, we define a linear code.

**Definition 11 (Linear Code)** *A linear code $\mathcal{C}$ over a field $\Sigma$ and of length $n$ is a subspace of $\Sigma^n$. For such a code, if the field size is $q$, the code is termed a $q$-ary linear code. The notation $[n, k]_q$ represents a $q$-ary linear code of block length $n$ and dimension $k$, and if the code also specifies a minimum Hamming distance $d$, it is denoted as $[n, k, d]_q$ code.*

In more detail, a field $\Sigma$ is a set equipped with two operations (addition and multiplication) that adhere to the rules of arithmetic, including the presence of additive and multiplicative inverses for every element. A subspace of $\Sigma^n$, where $n$ represents the length of codewords, is a set of sequences over $\Sigma$ that is closed under these operations, meaning any linear combination of elements (codewords) within the subspace remains in the subspace. The dimension $k$ of a code $C$ reflects the number of linearly independent vectors in $C$, indicating the minimum set of vectors required to generate all elements of $C$ through linear combinations.

For this paper, we only use binary codes, i.e., $\Sigma = \mathbb{F}_2$. Therefore, we omit the subscript in the notation of a linear code, and we use $[n, k, d] := [n, k, d]_2$.

## Appendix B. Construction of $(r-1)$-wise uniform distributions - Proof of Theorem 4

In this part, we present a construction of a distribution $\mathcal{Q}$ over the set of clauses $X_k$ that ensures the following two (informal) properties.

1. **Planted solutions.** Any instance $F$ consisting of clauses drawn from $\mathcal{Q}$ has some $L$ distinct solutions $\sigma_1, \ldots, \sigma_L$ with probability 1. We will formalize the specific number of solutions later in this section.

2. **Indistinguishability.** It is "difficult" to distinguish $F$ from an instance generated uniformly at random provided that the number of clauses in $F$ is limited. We will formalize this notion by providing statistical query lower bounds for the corresponding testing problem in Section C.

For the first property we require $\mathcal{Q}$ to be supported over clauses satisfied by some fixed assignments $\sigma_1, \ldots, \sigma_L$. For the second property to hold, we require that for some parameter $r$, the marginal distribution of $\mathcal{Q}$ on subsets of less than $r$ variables are uniform. Formally, we define the notion of $(r-1)$-wise uniform distributions as follows.

**Definition 12** *We say a distribution $Q$ over $\{-1,1\}^k$ is* $(r-1)$-wise uniform *if for any subset $S \subseteq [k]$ of size $|S| < r$, the marginal distribution $Q[S]$ is uniform. We say a distribution $\mathcal{Q}$ over clauses is $(r-1)$-wise uniform if the distribution of $\mathbf{I}$ where $(\mathbf{I}, \mathbf{x}) \sim \mathcal{Q}$ is uniform over $\mathcal{I}$ and for any $I$, the distribution $Q_I$ is $(r-1)$-wise uniform, where we recall that $Q_I$ denotes the conditional distribution $(\mathbf{x} \mid \mathbf{I} = I)$ for $(\mathbf{I}, \mathbf{x}) \sim \mathcal{Q}$ (see Section 2). The complexity of a distribution is the largest integer $r$ for which the distribution is $(r-1)$-wise uniform.*

Definition 12 generalizes the definition of distribution complexity in Feldman et al. (2015a) and $(r-1)$-wise uniformness in Kothari et al. (2017). While these works only consider these definitions for distributions $Q$ over $\{-1,1\}^k$, we have extended it to distributions over clauses. For the special case studied by these works where $\mathcal{Q}$ is a product distribution (i.e., $\mathbf{I}$ and $\mathbf{x}$ are sampled independently), the two definitions coincide.

Utilizing the generality of the above definition, we will construct distributions according to the following recipe. First, a sequence of $k$ different variables of a clause, denoted by $\mathbf{I}$, is sampled uniformly, without replacement from the set $[n]$. Then, conditioned on the set $\mathbf{I}$, a sign pattern $x$ is sampled according to a marginal distribution $Q_I$, to be specified later. The final clause is a tuple of the form $\mathbf{C} := (\mathbf{I}, \mathbf{x})$. The marginal distribution is chosen such that the final clause is satisfied by all the assignments from the predetermined set with probability 1 and such that it is $(r-1)$-wise

uniform. The generality is crucially used in the fact that the distribution of the sign pattern can depend on the set of variables.

Formally, we can define the problem of finding the right distribution using linear programming. Fix an arbitrary set of assignments $\sigma_1, \ldots, \sigma_L \in \{-1,1\}^n$. Definition 12 requires that for any $I$, the marginals of $Q_I$ over less than $r$ variables are uniform. This, in turn, is equivalent to the Fourier coefficients of $Q_I$ being zero for all non-empty subsets of size less than $r$, i.e.,

$$\hat{Q}_I(S) := \frac{1}{2^k} \sum_{x \in \{-1,1\}^k} (\, Q_I(x) \chi_S(x) \,) = 0 \quad \forall 0 < |S| < r, \tag{2}$$

where $\chi_S(x) := \oplus_{s \in S} x$ denotes the XOR of $x[S]$. In order to specify $\mathcal{Q}$, it suffices to specify a distribution $Q_I$ for any $I$ such that:
**(a)** with probability 1 the clause $(I, \mathbf{x})$ for $\mathbf{x} \sim Q_I$ satisfies $\sigma_1, \ldots, \sigma_L$,
**(b)** $Q_I$ satisfies Equation (2).
Condition **(a)** is equivalent to requiring the probability assigned under $Q_I$ to $-\sigma_i[I]$ to be 0 as this is the unique sign pattern for which $\sigma_i$ is not feasible. As for **(b)**, if we view $Q_I(x)$ as variables for $x \in \{-1,1\}^k$, then (2) turns into a linear equation for $Q_I(x)$. Putting everything together, we formulate the problem of finding a suitable $Q_I$ as the following linear program where the variables are $Q_I(x)$ for $x \in \{-1,1\}^k$:

$$\begin{aligned} \text{Find a distribution} \quad & Q_I && \text{(LP1)} \\ \text{subject to} \quad & \underset{0<|S|<r}{\forall} \hat{Q}_I(S) = 0, && (3) \\ & \underset{i \in [L]}{\forall} Q_I(-\sigma_i[I]) = 0, \\ & \sum_x Q_I(x) = 1 \text{ and } Q_I \geq 0 \end{aligned}$$

This way, in order to sample a clause $(\mathbf{I}, \mathbf{x})$ from $\mathcal{Q}$, we first sample $\mathbf{I}$ uniformly at random. We then solve linear program (LP1) with $I$ set to $\mathbf{I}$ and sample $\mathbf{x}$ from the resulting distribution. We note that while the above linear program can always be formulated for any set of assignments $\sigma_i$, it is not guaranteed to be feasible. For instance, in the extreme case where we consider all solutions in $\{-1,1\}^n$, the program is always infeasible.

In the next part of the section, we show that the problem is always feasible provided the number of solutions $L$ is at most $2^t - 1$. The proof explicitly constructs a solution to the above linear problem by reducing it to the problem of designing a binary linear code. The remainder of this section is organized as follows. We first give an abstract framework of how, based on a $[k, t, r]$ binary linear code, one can construct a distribution $\mathcal{Q}$ over $X_k$ that satisfies $2^t - 1$ arbitrary predetermined assignments. The distribution $\mathcal{Q}$ will form a solution to the linear program (LP1) assuming $L = 2^t - 1$. Finally, at the end of this section, we discuss Gilbert's construction of a binary linear code and how it influences bounds attainable by the construction.

### B.1. Linear code reduction

In the light of the above reasoning, and specifically by the generality of the definition 12, we can reduce the construction of the distribution $Q$ to constructing $(r-1)$-wise uniform distribution $\mathcal{Q}_I$

assuming that $I$ is a given set of variables. In this part, we show how to solve the latter part by reducing it to the existence of a binary linear code with good properties. Specifically, let us fix a sequence of variables $I$. For a binary linear code with parameters $[k, t, r]$, our construction is such that $\mathcal{Q}_I$ contains a planted set of $2^t - 1$ predetermined solutions $\mathcal{A} = \{\sigma_1, \ldots, \sigma_{2^t-1}\}$ and is $(r-1)$-wise uniform. We next formally state a lemma that describes the dependency between the existence of a linear code and properties of the distribution $\mathcal{Q}_I$. The proof of this lemma is the central result of this section. A pseudocode incorporating both the reduction and the specifics of the construction conditioned on the sequence of variables $I$ is given in Figure 1. To align better with notation used for linear codes, without loss of generality, in the section we treat the sign patterns $x$ in the clauses and the assignments $\sigma_1, \ldots, \sigma_t$ as binary vectors over $\mathbb{F}_2$ field taking values in $\{0,1\}^k$ and $\{0,1\}^n$ respectively.

**Lemma 13** *Let $I \subseteq [n]$ of size $k$ be a set of variables and let $\mathcal{A} \subseteq \{0,1\}^n$ be a set of assignments where $|\mathcal{A}| \leq 2^t - 1$. Assume that there is a $[k, t, r]$ linear code. There exists a distribution $\mathcal{Q}_I$ over $\{0,1\}^k$ such that*
***(i)** for any $x$ such that $\mathcal{Q}_I(x) > 0$ the clause $C = (I, x)$ is satisfiable under $\sigma_1, \ldots, \sigma_{|\mathcal{A}|}$, where $\{\sigma_1, \ldots, \sigma_{|\mathcal{A}|}\} = \mathcal{A}$ and*
***(ii)** for any sequence $S$ of size $r-1$ of elements of $[k]$, the marginal distribution of $x[S]$ for $x \sim \mathcal{Q}_I$ is uniform.*

We first give a sketch of the proof. Let $V \in \{0,1\}^{t \times k}$ denote the generator matrix of the binary linear code and let $v_1, \ldots, v_t$ denote the transposed rows of $V$, i.e. $v_i \in \mathbb{F}_2^k$. By definition of a $[k, t, r]$ linear code, any two vectors $u \neq v$ in the subspace of $v_1, \ldots, v_t$ have Hamming distance $\geq r$.

We set the distribution $\mathcal{Q}_I$ to be uniform over the solutions of the linear system $Vx = b$ over $\mathbb{F}_2$, where the value of $b$ will be specified later. Each row in the linear system will correspond to a subset of the assignments, the corresponding linear equation ensures feasibility for all of these assignments. This effectively proves condition **(a)**. To prove condition **(b)**, we will rely on the properties of the linear code, namely the large Hamming distance between any linear combination of the rows, to prove that the solutions of the system of linear equations are uniform when focusing on any subset of size $< r$.

We now proceed to a formal proof of Lemma 13. (Described in Algorithm 1) For a binary vector $v$ of the same number of elements as the variables set $I$, we define $I[v]$ as the variable subset of $I$ consisting of elements on these positions $j$ that have $v[j] = 1$. Define the following, inductive, partitioning of $\mathcal{A}$. Let $R_0 = \emptyset$. For $i \geq 1$ and as long as $R_{i-1} \neq \mathcal{A}$, we repeat the following operations. Partition the set of the assignments $\mathcal{A} \setminus R_{i-1}$ into two subsets, those with the even number of positive literals on positions in $I[v_i]$ and those with the odd number of positive literals (Recall that for an assignment $a \in \{0,1\}^n$, a literal on the position $i \in [n]$ is positive if and only if $a(i) = 1$). Let $\mathcal{A}_i$ be the subset with the larger number of assignments. Define $R_i = R_{i-1} \cup \mathcal{A}_i$ and let

$$y[i] := \bigoplus_{j=1}^{|I[v_i]|+1} 1 \oplus \bigoplus_{j \in I[v_i]} a[j]$$

for an arbitrary $a \in \mathcal{A}_i$. Since all assignments have the same parity of positive variables on positions $I[v_i]$, this definition does not depend on the choice of $a$, and it proves its validity. Also, note that

**Algorithm 1:** Generating a clause of size $k$ that satisfies a set of given assignments
**Input:** Assignments $\mathcal{A}$, number of variables $n$, $t \times k$ matrix of a $[t, k, r]$ linear code
**Output:** A random clause with $k$ variables that is satisfied by all assignments
**Function** `GenerateClause`($\mathcal{A}$, $n$, $V_{t\times k}$)**:**

```
Let I be a uniformly selected set of variables of size k
Let y = [0]^k be a vector to be filled in the construction
R_0 ← ∅
for i = 1 to t do
    Let R_i^(0) and R_i^(1) be empty sets of assignments
    for σ ∈ A \ R_{i-1} do
        if ⨁_{j=1}^{|I[v_i]|+1} 1 ⊕ ⨁_{j∈I[v_i]} a[j] = 0 then
            Add σ to R_i^(0)
        else
            Add σ to R_i^(1)
        end
    end
    if |R_i^(0)| > |R_i^(1)| then
        R_i ← R_{i-1} ∪ R_i^(0)
        y[i] ← 0
    else
        R_i ← R_{i-1} ∪ R_i^(1)
        y[i] ← 1
    end
end
Let x be a uniformly random solution for the equation Ax = y
return C = (I, x) as a random clause
```

this construction halves the size of the set $R_i$ in each iteration. Since $|\mathcal{A}| \leq 2^t - 1$, thus after at most $t$ iterations the construction ends. Also, in the following, without loss of generality, we assume that the construction ends exactly after $t$ iterations; otherwise, we can use a subcode (i.e., a linear subspace) of the original code of the dimension equal to the desired number of iterations. Thus, we get that the construction satisfies the following.

**Lemma 14** *Let $v_1, \ldots, v_t$ be a base of a $[k, t, r]$ binary linear code. The sequence of sets $\mathcal{A}_1, \ldots, \mathcal{A}_t$ and the sequence of numbers $y[i]$ give the following guarantees:*
*$(i)$ the family $\{\mathcal{A}_1, \ldots, \mathcal{A}_t\}$ is a partitioning of the set $\mathcal{A}$,*
*$(ii)$ for any $a \in \mathcal{A}$, let $i_a$ be the unique index such that $a \in \mathcal{A}_{i_a}$. Then, it holds that*

$$y[i_a] = \bigoplus_{j=1}^{|I[v_{i_a}]|+1} 1 \oplus \bigoplus_{j \in I[v_{i_a}]} a[j].$$

Let $y \in \mathbb{F}_2^t$ denote a vector having entries $y[i]$. Consider the linear system over $\mathbb{F}_2$ field

$$\begin{aligned} v_1^T x &= y[1] \\ v_2^T x &= y[2] \\ &\vdots \\ v_t^T x &= y[t], \end{aligned}$$

which can be written $Vx = y$ in the matrix form. Finally, let us define the distribution $Q_I$ as the uniform distribution over the space of the solutions of the linear system $V \cdot x = y$.

**Lemma 15** *A clause* $(\mathbf{I}, \mathbf{x})$ *where* $x$ *is the sign pattern drawn from* $Q_I$ *is satisfied by all assignments from the set* $\mathcal{A}$ *with probability* 1.

**Proof** Let $x^* \in \mathbb{F}_2^k$ be any solution to the system $Vx = y$. Let $a \in \mathcal{A}$ be one of the assignments and denote $i^*$ be the index so that $a \in \mathcal{A}_{i^*}$. In particular, we get that $x^*$ satisfies

$$v_{i^*}^T x^* = y[i^*],$$

which means that

$$\bigoplus_{j \in I[v_{i^*}]} x^*[j] = \bigoplus_{1=j}^{|I[v_{i^*}]|+1} 1 \oplus \bigoplus_{j \in I[v_{i^*}]} a[j],$$

and as a consequence

$$\bigoplus_{1=j}^{|I[v_{i^*}]|+1} 1 = \bigoplus_{j \in I[v_{i^*}]} \left(a[j] \oplus x^*[j]\right). \tag{4}$$

This means that there must be at least one index $j \in I[v_{i^*}]$ such that the positivity/negativity of the variable on the position $j$ of $x^*$ matches the literal $a[j]$. Otherwise,

$$\bigoplus_{j \in I[v_{i^*}]} (a(j) \oplus x^*(j)) = \bigoplus_{j \in I[v_{i^*}]} 1 \neq \bigoplus_{1 \leq j}^{|I[v_{i^*}]|+1} 1,$$

which contradicts 4. This proves that $a$ is satisfied by any clause belonging to the support of $Q_I$. ■

The next lemma guarantees that the generated distribution is $(r-1)$-wise uniform.

**Lemma 16** *The distribution* $Q_I$ *over sign patterns is* $(r-1)$*-wise uniform.*

We first provide an intuition for why linear codes show up. Assume that the rows of $V$ do not satisfy the Hamming distance condition, i.e., there exists some $u \neq 0$ in the subspace of $v_1, \ldots, v_t$ with Hamming distance $< r$ from the origin. We will provide a simple proof that the distribution is not $(r-1)$-wise uniform.

This means that there exists $\alpha \in \{0,1\}^n$ such that $u = \sum \alpha_i v_i$ . Summing the rows of the linear system $Vx = y$ with coefficients $\alpha[i]$, we conclude that for $x$ that is a solution to the system satisfies $u^T \cdot x = \sum \alpha[i] y[i]$. This means that the coordinates in $x$ corresponding to $u[i]$ always have a fixed

sum, and therefore are not distributed uniformly. Since $u$ has at most $(r-1)$ coordinates equal to 1, this means the distribution is not $(r-1)$-wise uniform.

The following lemma shows the converse of the above sketch, i.e., it shows that when the Hamming property does hold, the distribution is $(r-1)$-wise uniform.

**Proof** [Proof of Lemma 16] To prove the theorem, we calculate the number of solutions of the linear system $Vx = y$ for a vector $x$ that has any $r-1$ coordinates fixed. Denote the set of the $r-1$ positions $\ell_1, \ldots, \ell_{r-1} \in [k]$ and consider a fixed pattern of values $z_1, \ldots, z_{r-1} \in \{0,1\}$ that must be preserved on these positions. To calculate the number of solutions we consider the following system of equations.

$$\begin{aligned} v_1^T x &= y[1] \\ v_2^T x &= y[2] \\ &\vdots \\ v_t^T x &= y[t] \\ e_{\ell_1}^T x &= z[1] \\ &\vdots \\ e_{\ell_{r-1}}^T x &= z[r-1], \end{aligned}$$

where $e_{\ell_1}, \ldots, e_{\ell_{r-1}}$ are the vectors from the standard basis. We denote the matrix form of the system as $V_e x = y_e$. We first observe the set of vectors $\{v_1, \ldots, v_t, e_{\ell_1}, \ldots, e_{\ell_{r-1}}\}$ is linearly independent.

**Lemma 17** *Vectors $v_1, \ldots, v_t, e_{\ell_1}, \ldots, e_{\ell_{r-1}}$ are linearly independent.*

**Proof** Consider any subset $B$ of the set of vectors and partition it into $B_1 \subseteq \{v_1, \ldots, v_t\}$ and $B_2 \subseteq \{e_{\ell_1}, \ldots, e_{\ell_{r-1}}\}$. It holds that $\{v_1, \ldots, v_t\}$ is a linearly independent set of vectors, as it is the basis (codewords) of the linear code. Vectors $\{e_{\ell_1}, \ldots, e_{\ell_{r-1}}\}$ are linearly independent because they are a subset of the standard basis. Consider now a mixed linear combination $u_1 + u_2$, where $u_1 \in span\,(v_1, \ldots, v_t)$ and $u_2 \in span\left(e_{\ell_1}, \ldots, e_{\ell_{r-1}}\right)$. Since $u_1$ is a linear combination of codewords of the subspace $C$, it has at least $d$ distance from the codeword $(0, \ldots, 0)$, thus, it has at least $d$ non-zero entries. On the other hand, $dim\left(span\left(e_{\ell_1}, \ldots, e_{\ell_{r-1}}\right)\right) = r-1$, which means that $u_2$ has at most $r-1$ non-zero entries. Thus, it must be that $u_1 + u_2 \neq 0$ and the linear independence of the set of vectors $\{v_1, \ldots, v_t, e_{\ell_1}, \ldots, e_{\ell_{r-1}}\}$ follows. ■

Using Lemma 17, we obtain that the matrix $V_e$ has full rank. Therefore, it must also be that the augmented matrix $[V_e|y_e]$, derived from the linear system $V_e x = y_e$, is also a full rank matrix. We can apply then the Rouché–Capelli theorem to argue that the linear system $V_e x = y_e$ has exactly $|\mathbb{F}_2|^{k-t-(r-1)} = 2^{k-t-(r-1)}$ solutions. Observe that the number is independent of the choice of the fixed positions and the pattern of 0's and 1's on these positions. In consequence, if $Q_I$ is the uniform distribution over the set of all solutions of $Vx = y$, it follows that

$$\Pr_{x \sim Q_I}\left[x[L] = z\right] = \frac{2^{k-t-(r-1)}}{2^{k-t}} = \frac{1}{2^{r-1}},$$

which proves $(r-1)$-uniformness of $Q_I$. ■

### B.2. Binary linear code construction

The above construction relies on the relationship between the parameters $t$ and $r$ in the choice of the $[k, t, r]$ binary code. Using Gilbert-Varshamov's bound Gilbert (1952); Varshamov (1957), we can realize the trade-off such that the number of planted assignments is exponential in the distance of the binary code. Formally, Gilbert-Varshamov's theorem, based on a random construction, guarantees the following.

**Theorem 18 (Gilbert-Varshamov's theorem)** *Let $H(x)$ be the entropy function defined as $H(x) = -x\log_2(x) - (1-x)\log_2(1-x)$). For any $k$, $0 \leq \delta_0 \leq \frac{1}{2}$, there exists a $[k, t, r]$ binary linear code $\mathcal{C}$ whose parameters $k, t, r$ satisfy $\delta(\mathcal{C}) = \delta_0$ and $R(\mathcal{C}) \geq 1 - H(\delta_0)$, where in this case $\delta(\mathcal{C}) = \frac{r}{k}$, $R(\mathcal{C}) = \frac{t}{k}$. .*

Using Lemma 13 together with the existence of this binary linear code gives the following result.

**Theorem 19** *Fix $k$, for a given $\delta_0$, $0 \leq \delta_0 \leq \frac{1}{2}$, there exists a $(\delta_0 k - 1)$-wise uniform distribution $\mathcal{Q}$ over clauses of size $k$ such that it satisfies a given set of assignments $\mathcal{A}$ of size at most $2^{k(1-H(\delta_0))} - 1$.*

**Proof** Consider a distribution $\mathcal{Q}$ defined by first randomly and uniformly selecting a set of variables $\mathbf{I}$ and then by applying Lemma 13 to the set of variables $\mathbf{I}$ to generate a clause $(\mathbf{I}, \mathbf{x})$. By Lemma 13, if the construction uses a code $\mathcal{C}$ with parameters $[k, t, r]$, and provided that $|\mathcal{A}| \leq 2^t - 1$, then the distribution $\mathcal{Q}$ is $(r-1)$-wise uniform and every assignment from $\mathcal{A}$ is satisfied by the support of the distribution.

To derive the achievable bounds on $k, t, r$, we refer to the result of Theorem 18. We first obtain the distance of the code. Theorem 18 specifies that $\delta(\mathcal{C}) = \delta_0$. Employing Definition 10 yields:

$$\begin{aligned} r &= \Delta(\mathcal{C}) \\ &= k \cdot \delta(C) && \text{(Definition 10)} \\ &= k \cdot \delta_0. && \text{(Theorem 18)} \end{aligned}$$

As for the dimension of the code, we derive a bound on it from the bound on $R(\mathcal{C})$. Theorem 18 asserts $R(\mathcal{C}) \geq 1 - H(\delta_0)$. By Definition 8, we get that

$$\begin{aligned} t &= k \cdot R(\mathcal{C}) && \text{(Definition 8)} \\ &\geq k(1 - H(\delta_0)). && \text{(Theorem 18)} \end{aligned}$$

Combining both bound together, we get that $\mathcal{C}$ can be a $[k, k(1 - H(\delta_0)), \delta_0 k]$ code. This code can cover at most $2^{k(1-H(\delta_0))}$ solutions. Thus, the theorem follows. ■

Remark that for a fixed $\delta_0$ the number of solutions can be $2^{k(1-H(\delta_0))} = 2^{\Theta(k)}$

## Appendix C. Universal statistical lower bound for $(r-1)$-wise uniform distribution - Proof of Theorem 1 and 3

In this section, we show how to generate hard instances via access to an $(r-1)$-wise uniform distribution $\mathcal{Q}$ over clauses of size $k$. Our main result is the following theorem. Specifically, we will show

that if we randomly "rotate" the distribution $\mathcal{Q}$ to obtain a new distribution $\mathcal{Q}'$ and repeatedly sample from $\mathcal{Q}'$, then the resulting instance will be hard. Our main result, Theorem 20 below, bounds the statistical dimension of the testing problem of distinguishing $\mathcal{Q}'$ from the uniform distribution $\mathcal{U}$. This further implies statistical query lower bounds for the problem (See Section 2).

Choose a *rotation* vector $\tau$ uniformly at random from $\{-1,1\}^n$ and let $\mathcal{Q}_\tau := \mathcal{Q} \oplus \tau$ denote the distribution obtained by applying the XOR with $\tau$ on the clause obtained from $\mathcal{Q}$ (see Section 2 for the definition). Formally, $\mathcal{Q}_\tau$ is the distribution of the clause $(\mathbf{I}, \mathbf{x} \oplus \tau[\mathbf{I}])$ where $(\mathbf{I}, \mathbf{x}) \sim \mathcal{Q}$. We generate the $k$-SAT instance $F$ by sampling $m$ clauses i.i.d from $\mathcal{Q}_\tau$. A formal pseudo code is shown in Algorithm 2. Our approach is a generalization of that of Feldman et al. (2015a) and Kothari et al. (2017) who obtain instances by fixing a distribution $Q$ over $\{-1,1\}^k$ and sampling clauses of the form $(\mathbf{I}, \mathbf{x} \oplus \tau[\mathbf{I}])$ where $\mathbf{I}$ is a uniformly random subset of size $k$ from $[n]$ and $\mathbf{x}$ is sampled from $Q$. Our model captures this special case by considering a product distribution for $\mathcal{Q}$; i.e., assuming that $\mathbf{I}$ and $\mathbf{x}$ are sampled independently. Our approach is more general, however as it allows the sampled value of $\mathbf{x}$ depend on the sampled value of $\mathbf{I}$, which is crucial for generating instances with multiple solutions, as mentioned in the introduction.

**Algorithm 2:** Generating an instance with $m$ clauses
**Input:** Distribution $\mathcal{Q}$, number of clauses $m$, number of variables $n$, number of variables in each clause $k$
**Output:** A SAT instance with $m$ clauses
**Function** `GenerateSATInstance` ($\mathcal{Q}$, $m$, $n$, $k$) **:**
  Let $\tau$ be a uniformly random vector in the space $\{-1,1\}^n$
  Initialize an empty SAT instance $F$
  **for** $1$ ***to*** $m$ **do**
    Sample a clause $(I, x)$ from $\mathcal{Q}$
    **for** $i = 1$ ***to*** $k$ **do**
      $x[i] \leftarrow x[i] \oplus \tau[I[i]]$
    **end**
    Add clause $C$ to $F$
  **end**
  **return** $F$

We next study the hardness of distinguishing the planted instance described above from an instance drawn uniformly at random by studying the problem in the statistical query model. Specifically, for any algorithm that has indirect access to the instance via a statistical query oracle for some distribution $\mathcal{P}$, we provide a lower bound on the number of queries it needs to determine whether $\mathcal{P} = \mathcal{U}$ or $\mathcal{P} \in \mathcal{Q}_\tau$. Formally, let $\mathcal{D}_\mathcal{Q} := \{\mathcal{Q}_\tau : \tau \in \{-1,1\}^n\}$ denote the set of all distributions $\mathcal{Q}$. Our main result is the following Theorem, which can be seen as a generalization of Theorem 3.5 in Feldman et al. (2015a) who only consider the case where $\mathcal{Q}$ is a product distribution.

**Theorem 20** *For every $(r-1)$-wise uniform distribution $\mathcal{Q}$ over $X_k$, there exists a constant $c > 0$, depending on $k$, such that, for any $q \geq 1$,*

$$\mathrm{SDN}\left(\mathcal{B}(\mathcal{D}_\mathcal{Q}, \mathcal{U}), \frac{c(\log q)^{r/2}}{n^{r/2}}\right) \geq q. \tag{5}$$

Combined with Theorem 19, we obtain the following result.

**Theorem 21** *Assume that $n, k, r, L$ satisfy $r \leq k/2$ and $L \leq 2^{k(1-H(\frac{r}{k}))}$, where $H$ is the binary entropy function. Let $\mathcal{A}$ be an arbitrary set of assignments of size at most $L$. There exists a distribution $\mathcal{Q}$ over $X_k$ such that*

1. *the testing problem $\mathcal{B}(\mathcal{D}_{\mathcal{Q}}, \mathcal{U})$ where $\mathcal{D}_{\mathcal{Q}} = \{\mathcal{Q} \oplus \tau : \tau \in \{-1, 1\}^n\}$ satisfies the bound in Equation (5).*
2. *Any clause $C \sim \mathcal{Q}$ satisfies all of the solutions in $\mathcal{A}$ with probability 1.*

Combining the above theorems with the reduction results obtained in Feldman et al. (2015a) (See Section 2) we get

**Theorem 22** *Any randomized statistical algorithm that correctly decides the testing problem $\mathcal{B}(\mathcal{D}_{\mathcal{Q}}, \mathcal{U})$ with a probability of at least $2/3$ (considering the random choice of $\mathcal{D}_{\mathcal{Q}}$ and the randomness of the algorithm) requires either:*

- *$m$ calls to the 1-MSTAT$(L)$ oracle, where $m \cdot L \geq c_1 \left(\frac{m}{\log n}\right)^r$ for a constant $c_1 = \Omega_k(1)$, or*
- *$q$ queries to MVSTAT$(L, \frac{c_2}{L}, \frac{n^r}{(\log q)^r})$ for a constant $c_2 = \Omega_k(1)$ and any $q \geq L$.*

### C.1. Proof of Theorem 20

The proof of the theorem is obtained by simplifying and generalizing the proof of Theorem 3.5 in Feldman et al. (2015a). Their proof effectively decomposes the problem into special cases of $\ell$-XOR-SAT for $\ell \leq k$ by considering the contribution of each subset $S \subseteq [k]$ in the Fourier expansion of the above difference. Specifically, they show that

$$\mathbb{E}_{\mathcal{Q}_\tau}[h] - \mathbb{E}_{\mathcal{U}h}[=] - 2^k \sum_{S \subseteq [k]} \hat{Q}(S) \left( \mathbb{E}_{Z_{\ell,\tau}}[h_S] - \mathbb{E}_{\mathcal{U}_\ell}[h_S] \right)$$

where $Z_{\ell,\tau}$ is the $\ell$-XOR-SAT distribution for $\tau$ and $h_S$ is defined as a projection of $h$ into $X_\ell$. Such a decomposition is not possible in our case, however, because the function $Q$ is not fixed and depends on $I$. As such, we cannot even define $\hat{Q}(S)$. To fix this issue, we opt for a more straightforward approach that directly handles the terms for each $S \subseteq [k]$.

**Proof** [Proof of Theorem 20] By definition of statistical dimension, it suffices to show that for any subset $\mathcal{D}' \subseteq \mathcal{D}_Q$ with size $|\mathcal{D}'| \geq |\mathcal{D}_Q|/q$ we have

$$\max_{h: ||\, h \,||_{\mathcal{U}}=1} \mathbb{E}_{\mathcal{Q}' \sim \mathcal{D}'} [\,|\, \mathbb{E}_{\mathcal{Q}'}[h] - \mathbb{E}_{\mathcal{U}}[h] \,|\,] \leq c \frac{(\log q)^{r/2}}{n^{r/2}}.$$

We first give a sketch of the proof. Set $\tau$ to satisfy $\mathcal{Q}' = \mathcal{Q}_\tau$. The expectation $\mathbb{E}_{\mathcal{Q}_\tau}[h]$ can be written as $\sum_{I,x} \mathcal{Q}_\tau(I, x) h(I, x)$. Let $Q_I$ denote the distribution, over sign patterns, of $\mathcal{Q}$ after conditioning on the variable sequence. Formally, $Q_I(x) = \Pr_{(\mathbf{I},\mathbf{x}) \sim \mathcal{Q}}[\mathbf{x} = x \mid \mathbf{I} = I]$. Since $\mathcal{Q}$ was assumed to be $(r-1)$-wise uniform, for any fixed $I$ we know that $\mathcal{Q}_\tau(I, x)$ is proportional to $Q_I(x \oplus \tau[I])$; specifically, $\mathcal{Q}_\tau(I, x) = \frac{1}{|\mathcal{I}|} Q_I(x \oplus \tau[I])$. We apply the Fourier transform on $Q_I$ to rewrite it as a (weighted) sum of $\chi_S(x \oplus \tau[I])$ terms for $S \subseteq [k]$. The term corresponding to $S \neq \emptyset$ cancels out with $\mathbb{E}_{\mathcal{U}}[h]$ because the Fourier coefficient for $\emptyset$ is $1/2^k$ for both $\mathcal{Q}$ and $\mathcal{U}$ since they are both distributions. We can therefore rewrite the difference $|\,\mathbb{E}_{\mathcal{Q}'}[h] - \mathbb{E}_{\mathcal{U}}[h]\,|$ as

$\left|\sum_{S\subseteq[k],S\neq\emptyset} M_S(\tau)\right|$, where $M_S(\tau)$ denotes the sum of the terms corresponding to $\chi_S(x\oplus\tau[I])$ in the expansion of $Q_I$. Crucially, we only need to consider $S$ of size at most $r$ since the smaller $S$ do not appear because of the $(r-1)$-wise independence condition.

Using Jensen's inequality, we show that it suffices to separately bound $\mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}'}[\,|\,M_S(\tau)\,|\,]$ for each $S$ and then combine these bounds for a valid upper bound on $\mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}'}[\sum_S|\,M_S(\tau)\,|]$. This is formally shown in Equation (7). We observe that each $M_S(\tau)$ is a polynomial in $\tau$ because it is a sum of $\chi_S(x\oplus\tau[I])$ terms for different $i$, and each such term can be written as $\chi_S(x)\prod_{i\in I[S]}\tau[i]$. Using the fact that $\tau$ is chosen from a large set, specifically $\mathcal{Q}_\tau\sim\mathcal{D}'$, we relate the expectation $\mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}'}[\,|\,M_S(\tau)\,|\,]$ to the norm of the coefficients of $M_S$. This norm can be bounded fairly easily, however using the assumption $||\,h\,||_{\mathcal{U}}=1$.

We proceed with a formal proof. We drop the subscript $k$ when it is clear from the context. Expanding the definition of $\mathcal{Q}_\tau$ and applying the Fourier transform, we obtain

$$\begin{aligned}
\mathbb{E}_{\mathcal{Q}_\tau}[h] &= \sum_{I\in\mathcal{I}}\sum_{x\in\{-1,1\}^k} h(I,x)\mathcal{Q}_\tau(I,x)\\
&= \sum_{I\in\mathcal{I}}\sum_{x\in\{-1,1\}^k} h(I,x)\mathcal{Q}(I,x\oplus\tau[I]) && \textit{(Definition of } \mathcal{Q}_\tau\textit{)}\\
&= \frac{1}{|\,\mathcal{I}\,|}\sum_{I\in\mathcal{I}}\sum_{x\in\{-1,1\}^k} h(I,x)Q_I(x\oplus\tau[I]) && \textit{(Definition of } Q_I\textit{)}\\
&= \frac{1}{|\,\mathcal{I}\,|}\sum_{I}\sum_{x}\sum_{S\subseteq[k]} h(I,x)\hat{Q}_I(S)\chi_S(x\oplus\tau[I]) && \textit{(Fourier expansion)}.
\end{aligned}$$

Let $\chi(x)=\oplus_{i=1}^{|x|}x[i]$ for an arbitrary vector $x$ denote the XOR of $x$. Note that $\chi$ is multiplicative, i.e., $\chi(x\oplus y)=\chi(x)\chi(y)$ for any two vectors $x,y$ of same length. Grouping the above sum in terms of the variable $S$, we can rewrite it as $\sum_S M_S(\tau)$ where

$$\begin{aligned}
M_S(\tau) &:= \frac{1}{|\,\mathcal{I}\,|}\sum_{I,x} h(I,x)\hat{Q}_I(S)\chi_S(x\oplus\tau[I])\\
&= \frac{1}{|\,\mathcal{I}\,|}\sum_{I,x} h(I,x)\hat{Q}_I(S)\chi((x\oplus\tau[I])[S]) && \textit{(Definition of } \chi_S\textit{)}\\
&= \frac{1}{|\,\mathcal{I}\,|}\sum_{I,x} h(I,x)\hat{Q}_I(S)\chi(x[S])\chi(\tau[I[S]]) && \textit{(Multiplicativity of } \chi\textit{)}. \qquad (6)
\end{aligned}$$

Similarly, for $\mathcal{U}$,

$$\begin{aligned}
\mathbb{E}_{\mathcal{U}}[h] &= \sum_{I\in\mathcal{I}}\sum_{x\in\{-1,1\}^k} h(I,x)\mathcal{U}(I,x)\\
&= \frac{1}{2^k\,|\,\mathcal{I}_k\,|}\sum_{I\in\mathcal{I}}\sum_{x\in\{-1,1\}^k} h(I,x) && \textit{(Definition of } \mathcal{U}\textit{)},
\end{aligned}$$

which is the same $M_\emptyset(\tau)$ because $\hat{Q}_I(\emptyset) = \frac{1}{2^k}\sum_x Q_I(x) = \frac{1}{2^k}$ and $\chi(x[\emptyset]) = \chi(\tau[I[\emptyset]]) = \chi(\emptyset) = 1$. It follows that

$$\begin{aligned}
\mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}_Q}\left[\left|\,\mathbb{E}_{\mathcal{Q}_\tau}\left[h\right] - \mathbb{E}_{\mathcal{U}}\left[h\right]\,\right|\right] &= \mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}_Q}\left[\left|\sum_{S\neq\emptyset} M_S(\tau)\right|\right] \\
&\leq \mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}_Q}\left[\sum_{S\neq\emptyset}\left|\,M_S(\tau)\,\right|\right] && \textit{(Jensen's inequality)} \\
&= \sum_{S\neq\emptyset}\mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}_Q}\left[\left|\,M_S(\tau)\,\right|\right] && \textit{(Linearity of expectation).} \quad (7)
\end{aligned}$$

It, therefore suffices to bound $\mathbb{E}_{\mathcal{Q}_\tau\sim\mathcal{D}_Q}\left[\left|\,M_S(\tau)\,\right|\right]$ for $S \neq \emptyset$. To do this, we view $M_S(\tau)$ as a polynomial of degree $\ell := |\,S\,|$ in $\tau$ and bound the norm of its coefficients. We then use the following lemma which turns any bound on the coefficients of a polynomial in $\tau$ into a bound on the expectation of the absolute value of this polynomial, where the expectation is over $\tau$ sampled from a large set (in this case $\mathcal{D}_\mathcal{Q}$).

**Lemma 23 (Lemma 5.5 of Feldman et al. (2015a))** *Let $p(x)$ be a polynomial of degree $\ell$ in $\{-1,1\}^n$, and let $\mathcal{S}\subseteq\{-1,1\}^n$ be a set of assignments. The following holds for $d = 2^n/\,|\,\mathcal{S}\,|$*

$$\mathbb{E}_{\tau\sim\mathcal{S}}\left[\left|\,p(\tau)\,\right|\right] \leq O_\ell\left(\left(\ln d\right)^{\ell/2}\left|\left|\,p\,\right|\right|_2\right).$$

We now proceed with a proof. For any $A\subseteq[n]$ of size $\ell$ we define

$$u_{S,A} = \frac{1}{|\,\mathcal{I}\,|}\sum_{I:I[S]=A}\sum_{x\in\{-1,1\}^k}\hat{Q}_I(S)\chi(x[S])h(I,x),$$

Grouping Equation (6) based on $I[S]$ we obtain $M_S(\tau) = \sum_A u_{S,A}\chi(\tau[A]) = \sum_A u_{S,A}\prod_{i\in A}\tau[i]$, which verifies that $M_S(\tau)$ is a polynomial in $\tau$.

We proceed to bound $||\,M_S\,||_2 = \sqrt{\sum_A u_{S,A}^2}$. For any fixed $A$, using the Cauchy–Schwarz inequality, we obtain

$$\begin{aligned}
|\,\mathcal{I}\,|^2\,u_{S,A}^2 &= \left(\sum_{I:I[S]=A}\sum_x \hat{Q}_I(S)\chi(x[S])h(I,x)\right)^2 \\
&\leq \left(\sum_{I:I[S]=A}\sum_x \hat{Q}_I^2(S)\chi^2(x[S])\right)\left(\sum_{I:I[S]=A}\sum_x h^2(I,x)\right). \quad (8)
\end{aligned}$$

We start by bounding the first term. By Parseval's identity,

$$\begin{aligned}
\hat{Q}_I^2(S) &\leq \sum_{S'}\hat{Q}_I^2(S') && \textit{(Non-negativity of } Q_I^2(S')\textit{)} \\
&= \mathbb{E}_{x\sim\{-1,1\}^k}\left[Q_I^2(x)\right] && \textit{(Parseval's identity)} \\
&\leq \mathbb{E}_{x\sim\{-1,1\}^k}\left[1\right] && \textit{(Since } Q_I(x)\leq 1\textit{)} \\
&= 1.
\end{aligned}$$

Since $\chi^2(x[S]) = 1$, this implies

$$\sum_{I:I[S]=A} \sum_x \hat{Q}_I^2(S)\chi^2(x[S]) \leq \sum_{I:I[S]=A} \sum_x 1 = |\{I : I[S] = A\}| 2^k,$$

We note however that since the constraint $\{I[S] = A\}$ groups all values of $I$ in terms of $I[S]$ and the number of possible values for $I$ and $I[S]$ is $|\mathcal{I}_k|$ and $|\mathcal{I}_\ell|$ respectively, $|\{I : I[S] = A\}| = \frac{|\mathcal{I}_k|}{|\mathcal{I}_\ell|}$. Plugging this back in (8) we obtain

$$\begin{aligned}
|\mathcal{I}_k|^2 ||M_S||_2^2 = |\mathcal{I}_k|^2 \sum_A u_{S,A}^2 &\leq 2^k \frac{|\mathcal{I}_k|}{|\mathcal{I}_\ell|} \left( \sum_A \sum_{I:I[S]=A} \sum_x h^2(I,x) \right) \\
&= 2^k \frac{|\mathcal{I}_k|}{|\mathcal{I}_\ell|} \left( \sum_{I,x} \sum_{A=I[S]} h^2(I,x) \right) && \textit{(Rearranging)} \\
&= 2^k \frac{|\mathcal{I}_k|}{|\mathcal{I}_\ell|} \left( \sum_{I,x} h^2(I,x) \right) && \textit{(Uniqueness of } I[S]\textit{)} \\
&\leq 2^{2k} \frac{|\mathcal{I}_k|}{|\mathcal{I}_\ell|} |\mathcal{I}_k|
\end{aligned}$$

where the last inequality uses the fact that

$$\frac{\sum_{I,x} h^2(I,x)}{|X_k|} = \mathbb{E}\left[h^2(I,x)\right] = ||h||_2^2 \leq 1 \implies \sum_{I,x} h^2(I,x) \leq |X_k| = 2^k |\mathcal{I}_k|.$$

It follows that $||M_S||_2 \leq \frac{2^k}{\sqrt{|\mathcal{I}_\ell|}}$. This bounds the norm of the coefficients of $M_S$ which is a polynomial in $\tau$. By assumption in the theorem, the set $\mathcal{D}'$ has size $\geq |\mathcal{D}|/q$. Therefore, Lemma 23 implies that

$$\mathbb{E}_{\mathcal{Q}_\tau \sim \mathcal{D}'} [|M_S(\tau)|] \leq O_k\left( \frac{\ln(q)^{\ell/2}}{\sqrt{|\mathcal{I}_\ell|}} \right) = O_k\left( \frac{\ln(q)^{\ell/2}}{n^{\ell/2}} \right)$$

Summing over $S$ and plugging back in Equation (7),

$$\mathbb{E}_{\mathcal{Q}_\tau \sim \mathcal{D}'} [|\mathbb{E}_{\mathcal{Q}_\tau}[h] - \mathbb{E}_{\mathcal{U}}[h]|] \leq O_k\left( \frac{\ln(q)^{r/2}}{n^{r/2}} \right),$$

which finishes the proof. ■

## Appendix D. Worst-case analysis of SAT instances with multiple solutions - Proof of Theorem 6

In this section, we formalize and analyze the computational complexity of satisfiability problems (SAT) when a large number of solutions is guaranteed. Specifically, we consider SAT instances where we know in advance that the instance either has no solutions or has at least a certain number of solutions.

**Definition 24** *Let SAT$(n, s)$ denote the class of SAT instances with $n$ variables, where it is known a priori that the instances have either zero solutions or at least $s$ solutions. The class of all SAT instances over $n$ variables can be denoted as SAT$(n, 1)$.*

**Theorem 25** *Assuming the Exponential Time Hypothesis (ETH), there is no polynomial-time algorithm for solving SAT$(n, 2^{n-\log(n)^{1+c}})$ for any $c > 0$.*

This statement underscores the profound computational challenge presented by SAT instances with exponentially many solutions, delineating a clear boundary in the solvability landscape. It is important to note that we say an algorithm $\mathcal{A}$ can *solve SAT$(n, s)$* if, for any formula $f$ from this class, $\mathcal{A}$ can determine whether $f$ is satisfiable or unsatisfiable.

**Proof** We construct a polynomial-time reduction from an arbitrary instance of *SAT*$(\log(n)^{1+c}, 1)$, known to be hard to solve in subexponential time under the Exponential Time Hypothesis (ETH), to *SAT*$(n, 2^{n-\log(n)^{1+c}})$. By augmenting the *SAT*$(\log(n)^{1+c}, 1)$ instance with $n-\log(n)^{1+c}$ additional variables, we enable each solution of the original problem to pair with $2^{n-\log(n)^{1+c}}$ assignments of the new variables, thus forming a *SAT*$(n, 2^{n-\log(n)^{1+c}})$ instance. Denote the size of the subinstance as $m = \log(n)^{1+c}$, leading to $n = 2^{m^{1/(1+c)}}$. The existence of a polynomial-time algorithm for *SAT*$(n, 2^{n-\log(n)^{1+c}})$, or $O(n^k)$, implies an algorithm for the subproblem within $O(2^{k \cdot m^{1/(1+c)}})$, which constitutes a subexponential time complexity. This contradicts the ETH by suggesting a subexponential time solution to a problem that is presumed to require exponential time, thereby affirming the infeasibility of such a polynomial-time algorithm for the given SAT formulation. ■

**Claim 26** *The introduction of a large solution space in SAT$(n, 2^{n-\log(n)^{1+c}})$ does not simplify the problem, preserving its computational complexity akin to that of general SAT instances.*

Furthermore, we demonstrate the tightness of our bound by considering the scenario where $c \geq 0$, specifically for $c = 0$, which implies $2^{n-\log(n)}$ solutions for the SAT instance. This scenario offers an intriguing contrast: a polynomial-time algorithm becomes feasible for *SAT*$(n, 2^{n-\log(n)})$.

To elucidate, consider a random assignment of variables in a *SAT*$(n, 2^{n-\log(n)})$ problem. The probability $P$ that a random assignment satisfies the instance is given by the ratio of the number of satisfying solutions to the total number of possible assignments:

$$P = \frac{2^{n-\log(n)}}{2^n} = \frac{1}{2^{\log(n)}} = \frac{1}{n}$$

Thus, the probability of a random assignment not satisfying the instance is $1 - \frac{1}{n}$. To find a satisfying assignment, we can repeatedly select random assignments. After $k$ independent trials, the probability that all trials fail to find a satisfying assignment is $(1-\frac{1}{n})^k$. To ensure a high probability of success, we aim for this probability to be less than a certain threshold, say $\epsilon$, where $\epsilon > 0$. To determine the required number of trials, we select $k$ satisfying $(1-\frac{1}{n})^k < \epsilon$, For a sufficiently large $n$, we can approximate $1 - \frac{1}{n}$ as $e^{-\frac{1}{n}}$, giving us: $e^{-\frac{k}{n}} < \epsilon$ Solving for $k$, we get: $k > n \log(\frac{1}{\epsilon})$

Since $k$ scales linearly with $n$, a polynomial number of executions (specifically, $O(n)$) suffices to provide a YES/NO answer to *SAT*$(n, 2^{n-\log(n)})$ with high probability, thereby illustrating that the bound of $2^{n-\log(n)^{1+c}}$ for $c > 0$ is tight. Reducing $c$ to 0 shifts the problem into a domain where polynomial-time solutions emerge, underscoring the critical nature of the $c > 0$ condition in maintaining the problem's hardness under our framework.

Furthermore, this technique can be utilized to demonstrate the hardness associated with specific restrictions. For example, in Hsieh et al. (2022) Hsieh, Mohanty, and Xu propose the open problem of distinguishing between 3-SAT instances that are either unsatisfiable or have at least $T$ solutions, under the stipulation that the input is guaranteed to be (7/8 + $\epsilon$)-satisfiable (refer to Question 1.22 in Hsieh et al. (2022)).

We can directly respond to and resolve this query by extending Theorem 25:

**Theorem 27** *Assuming the Exponential Time Hypothesis (ETH), there is no polynomial-time algorithm for solving SAT*$(n, 2^{n-\log(n)^{1+c}})$ *for any* $c > 0$*, even when it is guaranteed that the input is (7/8 + $\epsilon$)-satisfiable.*

**Proof** As noted by Hsieh et al. Hsieh et al. (2022), distinguishing between a (7/8 + $\epsilon$)-satisfiable 3-SAT instance and a fully satisfiable 3-SAT instance is NP-hard. Following the framework of Theorem 25, we consider a reduction from an instance of this problem to a *SAT*$(n, 2^{n-\log(n)^{1+c}})$ instance. By adding extra variables, we expand the solution space while preserving the (7/8 + $\epsilon$) satisfiability condition. The addition of variables exponentially increases the number of solutions only if the original instance is already satisfiable, without altering the unsatisfiability of cases where no solutions exist. Consequently, the complexity of distinguishing between these markedly different states—completely unsatisfiable versus possessing a multitude of solutions—remains NP-hard under the ETH. ■

It is also important to note that by maintaining the (7/8 + $\epsilon$) satisfiability condition during the reduction, all related justifications concerning this transformation are valid. This includes the argument around the tightness of this bound and the existence of a randomized algorithm for instances with at least $2^{n-\log(n)}$ solutions.