Electronic Colloquium on Computational Complexity, Report No. 185 (2025)

Computational Complexity in Property Testing

Renato Ferreira Pinto Jr.*
Columbia University

Diptaksho Palit†
Boston University

Sofya Raskhodnikova†
Boston University

Abstract

We initiate a systematic study of the computational complexity of property testing, focusing on the relationship between query and time complexity. While traditional work in property testing has emphasized query complexity—often via information-theoretic techniques—relatively little is known about the computational hardness of property testers. Our goal is to chart the landscape of *time-query* interplay and develop tools for proving time complexity lower bounds. Our first contribution is a pair of *time-query hierarchy* theorems for property testing. For all suitable nondecreasing functions $q(n)$ and $t(n)$ with $t(n) \geq q(n)$, we construct properties with query complexity $\widetilde{\Theta}(q(n))$ and time complexity $\widetilde{\Omega}(t(n))$. Our weak hierarchy holds unconditionally, whereas the strong version—assuming the Strong Exponential Time Hypothesis— provides better control over the time complexity of the constructed properties.

We then turn to halfspaces in $\mathbb{R}^d$, a fundamental class in property testing and learning theory. We study the problem of approximating the distance from the input function to the nearest halfspace within additive error ε. (The distance approximation problem is known to have roughly the same complexity as tolerant property testing for appropriate setting of parameters.) For the distribution-free distance approximation problem, known algorithms achieve query complexity $O(d/\varepsilon^2)$, but run in time $\widetilde{\Theta}(1/\varepsilon^d)$. We provide a fine-grained justification for this gap: assuming the (integer) k-SUM conjecture, any algorithm must have running time $(1/\varepsilon)^{\lceil (d+1)/2 \rceil - o(1)}$. This fine-grained lower bound yields a provable (under a well-established assumption) separation between query and time complexity for a natural and well-studied (tolerant) testing problem. We also prove that any randomized Statistical Query (SQ) algorithm under the standard Gaussian distribution requires $(1/\varepsilon)^{\Omega(d)}$ queries if the queries are answered with additive error up to $\varepsilon^{\Omega(d)}$, revealing a fundamental barrier even in the distribution-specific setting.

*Part of this work was done while R.F. was a student at the University of Waterloo and visiting Boston University. R.F. is supported by an NSERC Postdoctoral Fellowship, and was supported by an NSERC Canada Graduate Scholarship during the development of this article.

†D.P. and S.R. were supported by the U.S. National Science Foundation under Grant No. 2022446.

ISSN 1433-8092

Contents

1 Introduction

We initiate a systematic investigation of computational complexity of property testing, focusing in particular on the relationship between query and time complexity. Property testing [RS96, GGR98]—along with related models of tolerant testing and distance approximation [PRR06]—was introduced to study extremely efficient algorithms that run in sublinear time and therefore do not even have time to read the entire input. However, most prior work in the area analyzes only the *query complexity* of property testers, not their *running time*. Property testing textbooks (see, e.g., [Gol17, BY22]) also focus on query complexity because we understand it better than time complexity and can, in particular, prove lower bounds on query complexity using information-theoretic arguments. In many cases, property testers are computationally very simple and have similar query and time complexity. However, this is not always the case. The goal of our work is to understand the landscape of *query-time interplay* in property testing and to develop tools for proving computational hardness of property testers.

Our first contribution is time-query hierarchy theorems for property testing. For all appropriate[1] nondecreasing functions $q(n)$ and $t(n)$, with $t(n) = \widetilde{\Omega}(q(n))$, where n represents the length of the input, we exhibit properties with query complexity $\widetilde{\Theta}(q(n))$ and time complexity $\tilde{\Omega}(t(n))$. These results show the existence of properties with any desired query complexity and arbitrarily higher time complexity. We provide two hierarchy theorems: weak and strong. The weak hierarchy theorem holds unconditionally whereas the strong one assumes the Strong Exponential Time Hypothesis (SETH). The latter provides better control over the time complexity of the constructed properties: the weak hierarchy guarantees time complexity at most $2^{\mathrm{poly}(t(n))}$, whereas the strong version guarantees time complexity at most $\widetilde{O}(t(n)^{1+\gamma})$ for arbitrarily small $\gamma > 0$.

Next we focus on a specific, fundamental class of properties: halfspaces in $\mathbb{R}^d$. This class has been widely studied in property testing [MORS10a, MORS09, MORS10b, BBBY12, Har19, BFH21, CP22] and PAC learning [DEM96, KKMS08, DKZ20, DKK+21, BMR22, RV23, GKSV24]. It is one of the simplest classes that exhibit a gap between query and time complexity of known sublinear-time algorithms—specifically, for tolerant testing and the related task of distance approximation. In the (distribution-free) distance approximation problem for a property $\mathcal{P}$ (in our case, $\mathcal{P}$ is the set of halfspaces in $\mathbb{R}^d$), the goal is to estimate the distance from a function f to the nearest $g \in \mathcal{P}$ with additive error $\varepsilon \in (0,1)$ and high probability, given sample access to a distribution $\mathcal{D}$ over the domain and query access to f, where the distance between f and g is $\mathbb{P}_{x\sim\mathcal{D}}[f(x) \neq g(x)]$. The query and time complexity of this problem are nearly the same as for tolerant testing of $\mathcal{P}$ (with appropriate setting of parameters) [PRR06, PRW22].

To approximate the distance to $\mathcal{H}$, the class of halfspaces in $\mathbb{R}^d$, one can take a sample from distribution $\mathcal{D}$ and return the empirical distance, that is, the fraction of sample points that need to be changed to make the sample consistent with a halfspace. Since $\mathcal{H}$ has VC-dimension $d+1$, a sample of size $s := O(\frac{d}{\varepsilon^2})$ suffices [VC71, SB14]. However, no computationally efficient algorithms are known for this task. The fastest known algorithm is implied by the work of Matheny and Phillips [MP21] who showed how to approximate maximum bichromatic discrepancy for halfspaces in d-dimensions in time $\tilde{\Theta}(s + 1/\varepsilon^d)$, which allows us to estimate the empirical distance—and thus the distance to $\mathcal{H}$—in time $\tilde{O}(1/\varepsilon^d)$. This leaves a huge gap between the query complexity and the running time of the known distance approximation algorithms for halfspaces: $O(\frac{d}{\varepsilon^2})$ vs.

[1] In our hierarchy theorems, $t(n) = \Omega(\log^2 n)$. We do not optimize polylogarithmic factors, as they are sensitive to model details we consider insignificant; for instance, reading the input length n already takes $\Omega(\log n)$ time.

$\tilde{\Theta}(\frac{1}{\varepsilon^d})$. While this problem is NP-hard when the dimension d grows with the size of the input[2] [GR09, FGKP09], it is not known whether this gap is necessary for any constant dimension $d \geq 3$.

We justify this gap under the k-SUM conjecture, a widely used assumption for proving conditional lower bounds across computer science (see the lecture notes [WW20]). We demonstrate that, under the (integer) k-SUM conjecture, the exponential dependence on d is necessary for distribution-free distance approximation to $\mathcal{H}$: specifically, for all constant $d \in \mathbb{N}$ and $\gamma > 0$, there is no distribution-free distance approximation algorithm for halfspaces over $\mathbb{Z}^d$ running in time $(1/\varepsilon)^{\lceil (d+1)/2 \rceil - \gamma}$. E.g., for $d = 4$, our result gives a (conditional) separation of $O(\frac{1}{\varepsilon^2})$ vs. at least $(\frac{1}{\varepsilon})^{3-o(1)}$. Our fine-grained lower bound yields a provable (under a well-established assumption) separation between query and time complexity for a fundamental (tolerant) testing problem.

Our hardness result for distribution-free distance approximation for halfspaces raises a natural question: does the hardness persist for standard, well-structured input distributions, or does it arise only for carefully constructed worst cases? We provide evidence that the problem remains hard even for well-behaved distributions by proving a lower bound for Statistical Query (SQ) algorithms—a broad class of algorithms that can only access the input distribution via estimates of expectations of bounded query functions, rather than seeing individual samples directly. Specifically, we show that every SQ algorithm for distance approximation (and thus for agnostic learning) of halfspaces under the standard Gaussian distribution over $\mathbb{R}^d$ must use time at least $(1/\varepsilon)^{\Omega(d)}$ when the dimension d is constant. This unconditional lower bound in the SQ model implies that any faster general algorithm must exploit structure beyond what can be learned from simple expectation estimates alone, echoing the fine-grained distribution-free bound and revealing a fundamental computational barrier in the distribution-specific setting.

To summarize, our time-query hierarchies and time/query separations for concrete problems shed light on fundamental differences between information-theoretic and algorithmic barriers.

1.1 Motivation: known gaps between query and time complexity

There are many examples of property testing problems for which all known algorithms have much higher time complexity than query complexity. The first examples appear in the seminal work of Goldreich, Goldwasser, and Ron [GGR98], which studied multiple graph properties, including bipartiteness, k-colorability, ρ-clique, ρ-cut, ρ-bisection, and other graph partition problems. In the table of results on p. 667, all properties except bipartiteness exhibit significantly different query complexity and running time upper bounds: while the query complexity is polynomial in the relevant parameters, the running time is exponential. (The query complexity bounds for some of these problems have been improved in subsequent work [FR21, BS23, SS24], but the bounds on running time remain exponential.) Moreover, [GGR98] argued that if the time complexity of testing k-colorability (and other NP-complete problems) is polynomial in $1/\varepsilon$ then NP$\subseteq$BPP, because when ε is sufficiently small, a property tester solves the exact decision problem. Note that our time-query hierarchies hold for constant ε, so the argument that, for small enough ε, the corresponding property testing problems require solving exact decision problems does not apply.

Next, we mention several problems where the gap between the best upper bounds on query and time complexity scales with the input length n. A textbook [BY22, §10.3.5] highlights the following example: although *every* graph property is testable with a number of queries independent of n on minor-free graphs [NS11], the running time of the best known algorithms [NS11, Ona12], which

[2]The hardness results in [GR09, FGKP09] are stated for learning, but the hard instances used imply the same lower bounds for distribution-free distance approximation.

involve a brute-force search, has a large dependence on n. For testing monotonicity of functions over partially-ordered domains, [FLN+02, Ras03] gave an algorithm with query complexity $O(\sqrt{n})$, but no sublinear in n bound on the time complexity is known. For testing whether the input string has a sufficiently small period (up to $\frac{\log n}{6}$), [LN11] designed an algorithm that makes poly$(\log \log n)$ queries, but runs in time poly(n).

Recent work on testing and learning decision trees has established one setting where a query-time gap can be formally justified: [KST23] showed that distribution-free testing whether a Boolean function over n variables is a size-s decision tree (for $s = O(n)$), for constant distance parameter ε, is NP-hard by reducing from VertexCover instances of size $n^{\Omega(1)}$. In contrast, the sample complexity of testing this class is at most its VC-dimension, which is $O(s \log n)$. Thus, sufficiently strong superpolynomial time lower bounds for SAT would imply a query-time separation for this problem.

For tolerant testing and distance approximation, to our knowledge, the only formal justification for a query-time gap comes from recent work of [BKST23] on junta testing, which showed the NP-hardness of tolerantly testing k-juntas over n-variable Boolean functions in the distribution-free setting; as noted in [BHK25], implicit in [BKST23] is an $n^{k-o(1)}$ time lower bound for this problem under SETH, which is larger than the $O(2^k + \log \binom{n}{k})$ sample complexity upper bound from VC dimension theory. However, many gaps between query and time complexity of known algorithms remain unjustified. One family of examples comes from work on geometric properties [BMR22, EGR25]: halfspaces in $\mathbb{R}^d$ and triangles, convexity, and connectedness in $\mathbb{R}^2$. For connectedness, the gap is exponential in relevant parameters, while for halfspaces for constant dimension $d \geq 3$, triangles, and convexity, it is polynomial. In fact, [BMR22] (a journal version of [BMR16]) had a distance approximation algorithm for halfplanes with query complexity $\Theta(1/\varepsilon^2)$ and time $\Theta(1/\varepsilon^3)$, but it turns out the running time can be improved to $\tilde{O}(1/\varepsilon^2)$ using techniques in subsequent work of [MP21]. This raises the fundamental question: When are the gaps in the query and time complexity intrinsic to the problems and cannot be improved?

1.2 Overview of our results and techniques

Time-query hierarchy. Our (unconditional) weak and (conditional) strong time-query hierarchies are stated in Theorems 3.3 and 3.4, respectively. They are inspired by the query complexity hierarchies of Goldreich et al. [GKNR12]. To establish time–query hierarchies, we construct properties by combining two orthogonal sources of hardness: one component enforces the specified query complexity using a 3CNF property (constructed in [BHR05]) that requires a linear number of queries to test, but is easy to decide given the entire input, while the other enforces the specified time complexity by encoding a language that is either provably hard to decide (for the unconditional weak hierarchy) or assumed hard under SETH (for the strong hierarchy). To link the hardness of the language to property testing, we map the language through an error-correcting code of Spielman [Spi96], which is efficiently constructable, encodable, and decodable.

The final property stitches these parts together through repetition (to adjust input sizes to match the desired query and time bounds) and concatenation (to ensure both forms of hardness coexist). A key concatenation lemma guarantees that the combined property inherits both the query and time lower bounds while remaining testable within the claimed upper bounds. This modular design allows us to compose query and computational hardness in a clean, general way, yielding rich hierarchies.

Our hierarchy theorems cover bounds ranging from polylogarithmic to nearly exponential in the

input size, and therefore a precondition for formally stating and proving those results is a suitable computational model for property testing. In Section 2, we develop a model of property testing in random-access machines (RAMs), which are well-suited to algorithm design and analysis as well as standard in fine-grained complexity. Our model extends the logarithmic cost RAM model of Cook and Reckhow [CR73] and captures both property testing (with query access to the input) and classic computation; the logarithmic cost model is important to enable principled reductions between instances of vastly different input sizes, which is a key component of our proof.

Distribution-free distance approximation for halfspaces. Our fine-grained $(1/\varepsilon)^{\lceil (d+1)/2 \rceil - \gamma}$ time lower bound for distribution-free distance approximation for halfspaces over $\mathbb{Z}^d$ under the k-SUM conjecture is stated in Theorem 4.3. The proof builds on well-known connections between k-SUM and geometric degeneracy problems: specifically, the hardness of detecting whether any $d+1$ given points in $\mathbb{R}^d$ lie on a non-vertical hyperplane. Our reduction adapts a standard construction from computational geometry—embedding k-SUM instances as configurations of points that are affinely dependent iff the original instance is a YES instance—and then carefully modifies this point set to produce a labeled sample distribution that encodes the same combinatorial structure.

To ensure that distinguishing YES from NO instances reduces to estimating the distance to the class of halfspaces, the construction augments each point with a small vertical "witness" gadget: each original point is replaced by a pair of nearby points just above and below its hyperplane, labeled oppositely. Any halfspace that fits the data must cut correctly between these pairs, so approximate distance estimation with fine enough additive error effectively solves the underlying k-SUM instance. By working in the integer grid and bounding coordinates, the reduction avoids trivial hardness due to input encoding and matches a realistic RAM model. The result is a nearly tight time lower bound for distribution-free testing of halfspaces in low dimension under a standard fine-grained complexity assumption.

The cost of tolerance. One implication of our result is that, for distribution-free testing of halfspaces in low-dimensional space, tolerance is more expensive in terms of time than queries. Specifically, for constant dimension d, the query complexity of standard testing is $\widetilde{\Theta}(1/\varepsilon)$, while the query complexity of tolerant testing is $\tilde{\Theta}(1/\varepsilon^2)$ by VC theory and [BMR22]; hence the query gap between standard and tolerant testing is quadratic up to logarithmic factors. In contrast, standard testing can be done in $\mathrm{poly}(d/\varepsilon)$ time via linear programming, from which our $(1/\varepsilon)^{\Omega(d)}$ tolerant lower bound (under the k-SUM conjecture) establishes an arbitrarily large polynomial separation for time complexity.

SQ lower bound under the Gaussian distribution. The SQ model, introduced by [Kea98] as a natural restriction of the PAC learning model of [Val84], allows access to the input f only via approximations to the expectations $\mathbb{E}[q(x, f(x))]$ of bounded query functions q (see Section 5.1 for a formal definition). A sample complexity lower bound against SQ algorithms serves as evidence of a time complexity lower bound against general algorithms in the sense that any faster algorithm must go beyond estimating expectations $\mathbb{E}[q(x, f(x))]$ from independent samples. SQ is attractive because unconditional lower bounds can be proved for this model, yet it still captures a wide variety of learning algorithms (see [Rey20] for a survey).

Our $(1/\varepsilon)^{\Omega(d)}$ lower bound for (randomized) SQ distance approximation algorithms for halfspaces under the Gaussian distribution is stated in Theorem 5.2. The starting point of our proof is an SQ lower bound of $d^{\mathrm{poly}(1/\varepsilon)}$ for agnostically learning halfspaces under the Gaussian distribution in the high-dimensional setting where $d \gg \mathrm{poly}(1/\varepsilon)$ [DKZ20]. That work constructed, from packing

number lower bounds on the sphere, sets of Boolean functions which are correlated with halfspaces, yet hard to learn using statistical queries. Our proof builds upon their work via two key components: 1) a packing number lower bound for the low-dimensional sphere, which we obtain by analyzing the limit distribution of angles in random packings from [CFJ13] in our precise asymptotic regime; and 2) a "pseudorandom" Boolean function on $\mathbb{R}^d$, which is adversarially constructed to be uncorrelated with the queries of a given SQ algorithm, and serves as the NO instance in our proof.

1.3 Prior work on hierarchies in property testing

In addition to the query hierarchies in terms of the input size n by [GKNR12] that inspired our query-time hierarchy theorems, there is a query hierarchy in terms of the distance parameter ε by [Gol19]. It states that "for essentially every function $q : (0, 1] \to \mathbb{N}$, there exists a property of Boolean functions that is testable" with $O(q(\Omega(\varepsilon)))$ queries but not with $o(q(O(\varepsilon)))$ queries. We believe our techniques can be used to extend this result to query-time hierarchies in terms of ε.

Additionally, [CG18] showed an adaptivity hierarchy for property testing by constructing properties that are easy to test (using $\widetilde{O}(k)$ queries) for algorithms with k rounds of adaptivity, but require $\widetilde{\Omega}(n/k^2)$ queries for algorithms with $k - 1$ rounds.

1.4 Open questions

One open question is to resolve the exponential gap in the time upper and lower bounds in our unconditional query-time hierarchy theorem. A hierarchy of the form $\mathsf{BPTIME}[t(n)^{1+\gamma}] \not\subseteq \mathsf{BPTIME}[t(n)]$ in the RAM model would improve this gap to an upper bound $t(n)^{1+\gamma}$ for the lower bound $t(n)$. Currently, the best known randomized time hierarchy also has an exponential gap between the time upper and lower bounds (through brute forcing all possible random tape initializations).

In Section 1.1, we collected many specific testing problems with unexplained gaps in query and time complexity. Specifically, for halfspace distance approximation (and tolerant testing), our results partially justify this gap for dimension $d \geq 4$. For $d = 3$, the problem remains open. It is open as well (for all constant $d \geq 3$) for the uniform distribution (e.g., over a unit cube or sphere). For the Gaussian distribution, it is unknown whether $(1/\varepsilon)^{\Omega(d)}$ time is required for this problem, or whether our SQ lower bound can be circumvented by some non-SQ algorithm (in the high-dimensional setting, the $d^{\mathrm{poly}(1/\varepsilon)}$ SQ lower bound for agnostic learning shown by [DKZ20] is supported by a cryptographic hardness result based on the LWE problem [DKR23]). Can these algorithmic gaps be closed by new, more efficient algorithms, or do they reflect inherent complexity that can be formalized through fine-grained or cryptographic lower bounds?

Organization. The rest of this paper is organized as follows. In Section 2, we formalize our computational model and lay out technical preliminaries for the rest of the paper. In Section 3, we prove our time-query hierarchy theorems. In Section 4, we prove our fine-grained hardness result for distribution-free distance approximation for halfspaces. Finally, in Section 5 we prove our SQ lower bound for distance approximation for halfspaces under the Gaussian distribution.

2 Our model and setup

We use $\langle \mathcal{O} \rangle$ to denote the binary representation of $\mathcal{O}$: for integers, this is the standard base-2 encoding; for other objects, the representation will be specified as needed.

2.1 Computational model

To precisely characterize time complexity in property testing, we must choose an appropriate computational model. While oracle machines (e.g., [Gol17]) suffice for analyzing query complexity, RAM models are better suited for fine-grained time analysis and yield sharper bounds than Turing machines. (The best known simulation of a time-t RAM by a Turing machine takes time t^2). RAM models are standard in fine-grained complexity (e.g., [Vas15]).

We propose a natural augmentation of the classic logarithmic cost RAM model of [CR73] for property testing. In that model, the program has access to an infinite sequence of registers $X_1, X_2, \ldots$, each holding an integer; basic operations on register X_i incur cost[3] $\log(|X_i|)$, where $|X_i|$ is the absolute value of the integer stored at X_i; and the *indirect* operations of reading and writing from register X_{X_j} incur additional cost $\log(X_j)$. The machine can read from an input tape and print to an output tape.

In the original model, the input tape holds a finite binary string, and the operation $\textsc{Read}(X_i)$ reads the next symbol from the input tape into register X_i. To model property testing (see Definition 2.2), we replace the single input tape with two: the *parameter tape* holding p, readable bit by bit via the operation ReadParam, and the *input tape* holding a binary string x, accessible via arbitrary queries using the operation $\textsc{QueryInp}(X_j, X_i)$, which loads the X_j-th bit of x into X_i.

To handle property testing, we let $n := |x|$ and place $p = \langle n \rangle$ on the parameter tape,[4] so the algorithm can first read the input length and then query bits of x. We define the query complexity of a tester as the maximum number of QueryInp calls on inputs of length n. To recover standard computation (without queries), we place the input on the parameter tape and leave the input tape empty. Algorithm $\mathcal{A}$ *computes* a function $f : \{0,1\}^* \to \{0,1\}^*$ in time t if, given input p on the parameter tape and emtpy input tape, it prints $f(p)$ and halts after executing operations with total time cost at most t. In both settings, $\mathcal{A}$ accepts if it prints 1, and rejects if it prints 0. Randomized algorithms may fail with with some probability over their internal randomness.

Table 1 summarizes the list of operations in the model, along with the time cost of each operation. While [CR73] only included addition and subtraction as the built-in arithmetic operations in their model, noting that multiplication could be simulated but was unnecessary for the algorithms studied in that paper[5], we explicitly model multiplication and division between registers X_i and X_j as taking $O(\log n \log\log n) = O(\log^{1+o(1)} n)$ time[6] when $|X_i| \le n$ and $|X_j| \le n$. We also include the operation FlipCoin to support randomized algorithms.

2.2 Property testing

Next, we formalize property testing in our model.

Definition 2.1 (Property, distance)**.** *For $n \in \mathbb{N}$, a* property $\mathcal{P}_n$ *(over inputs of length n) is a set $\mathcal{P}_n \subseteq \{0,1\}^n$. Denote the property $\{\mathcal{P}_n\}_{n\in\mathbb{N}}$ (over inputs of all lengths) by $\mathcal{P}$. Given n-bit strings x and y, let $\mathsf{dist}(x,y) = \mathop{\mathbb{P}}_{i\in[n]}[x_i \neq y_i]$ denote the relative Hamming distance between them. For property $\mathcal{P}$, define $\mathsf{dist}(x,\mathcal{P}) := \min_{y\in\mathcal{P}_n} \mathsf{dist}(x,y)$. String x is ε-*far *from property $\mathcal{P}$ if $\mathsf{dist}(x,\mathcal{P}) \ge \varepsilon$.*

[3] Formally, the cost should be $\log(\max(2, |X_i|))$, but we omit this technical detail for simplicity.

[4] We fix the alphabet $\{-1, 0, 1\}$ and stipulate that the input and parameter tapes consist of the binary strings x and p, respectively, followed by the symbol -1 in all the remaining tape cells.

[5] The algorithms literature features a vast diversity of RAM model variations; see [GJ22] for a discussion.

[6] This matches the complexity in multitape Turing machines [HvdH21]; we do not optimize polylogarithmic factors.

Table 1: Operations and their costs in the logarithmic cost RAM model for property testing.

Category	Operation	Description	Time cost
Input & Output	$\textsc{ReadParam}(X_i)$	read next bit of p into X_i	$O(1)$
	$\textsc{QueryInp}(X_j, X_i)$	read X_j-th bit of x into X_i	$O(\log(X_j))$
	$\textsc{Print}(X_i)$	output X_i in binary notation	$O(\log(\lvert X_i\rvert))$
Arithmetic & Control	$X_i \leftarrow C$, C any integer	constant assignment	$O(1)$
	$X_i \leftarrow X_j + X_k$	addition	$O(\log(\lvert X_j\rvert \cdot \lvert X_k\rvert))$
	$X_i \leftarrow X_j - X_k$	subtraction	$O(\log(\lvert X_j\rvert \cdot \lvert X_k\rvert))$
	$X_i \leftarrow X_j \cdot X_k$	multiplication	$O(\log^{1+o(1)}(\lvert X_j\rvert \cdot \lvert X_k\rvert))$
	$X_i \leftarrow \lfloor X_j / X_k \rfloor$	integer division	$O(\log^{1+o(1)}(\lvert X_j\rvert \cdot \lvert X_k\rvert))$
	$\textsc{Jump}(m, X_i > 0)$	jump to line m if $X_i > 0$	$O(\log(\lvert X_i\rvert))$
Indirection	$X_i \leftarrow X_{X_j}$	indirect fetch	$O(\log(X_j) + \log(\lvert X_{X_j}\rvert))$
	$X_{X_j} \leftarrow X_i$	indirect store	$O(\log(X_j) + \log(\lvert X_i\rvert))$
Randomness	$\textsc{FlipCoin}(X_i)$	put uniformly random bit in X_i	$O(1)$

Definition 2.2 (Tester)**.** *For a property $\mathcal{P}$ and a parameter $\varepsilon \in (0,1)$, an ε-tester for $\mathcal{P}$ is a randomized algorithm $\mathcal{A}$ such that, for each input length $n \in \mathbb{N}$ and $x \in \{0,1\}^n$, when $\mathcal{A}$ is run with $\langle n \rangle$ on the parameter tape and x on the input tape, $\mathcal{A}$ accepts with probability at least $\frac{2}{3}$ if $x \in \mathcal{P}$ and rejects with probability at least $\frac{2}{3}$ if x is ε-far from $\mathcal{P}$. Tester $\mathcal{A}$ has* time complexity *$t(n)$ if the maximum time it takes to terminate on inputs of length n is $t(n)$. Tester $\mathcal{A}$ has* query complexity *$q(n)$ if the maximum number of queries it makes on inputs of length n is $q(n)$.*

Remark 2.3. Definition 2.2 places a lower bound of $\Omega(\log n)$ on the time complexity of most nontrivial property testing algorithms, as this is the time required to read the input length n (and to read a general bit of x). We limit our attention to $\Omega(\log n)$-time algorithms because below this threshold, the model choice becomes overly influential and the study too brittle to be meaningful.

Remark 2.4. In Section 3, we treat $\varepsilon > 0$ as a constant and part of the problem definition, whereas in Section 4, the parameter ε is given to the tester (on the parameter tape).

2.3 Notation

Asymptotic behavior of functions. A function $k : \mathbb{N} \to \mathbb{N}$ is *eventually surjective* if its image omits only finitely many natural numbers. Given nondecreasing $f : \mathbb{N} \to \mathbb{R}$ and function $g : \mathbb{N} \to \mathbb{R}$, f has *slope* $O(g(n))$ if there exists a constant $c \in \mathbb{N}$ such that $f(n) - f(n-1) \leq c \cdot g(n)$ for all but finitely many $n \in \mathbb{N}$. A set $S \subset \{0,1\}^*$ is *almost everywhere (a.e.) nonempty* if $S \cap \{0,1\}^n$ is nonempty for all but finitely many $n \in \mathbb{N}$.

Computable and invertible functions. A function $f : \mathbb{N} \to \mathbb{N}$ is computable in time $t : \mathbb{N} \to \mathbb{R}$ if there exists an algorithm $\mathcal{A}$ such that, on each $n \in \mathbb{N}$, when $\langle n \rangle$ is placed on the parameter tape, algorithm $\mathcal{A}$ computes $f(n)$ in time $t(n)$.

Given a function $f : \mathbb{N} \to \mathbb{N}$, the function $f^\dagger : \mathbb{N} \to \mathbb{N}$ is a *right inverse* of f if, for all n in the image

of f, it holds that $f(f^\dagger(n)) = n$. For nondecreasing f, we define its *minimum inverse* $f^{-1} : \mathbb{N} \to \mathbb{N}$ as follows: for each $n \in \mathbb{N}$, let $f^{-1}(n) := n'$ where $n' \in \mathbb{N}$ is the smallest number satisfying $f(n') \geq n$. Then f^{-1} is a right inverse of f, and we have $f(f^{-1}(n)) \geq n$ and $f^{-1}(f(n)) \leq n$.

String notation. For a string x and natural numbers $s \leq t$, we write x^t for the concatenation of t copies of x, and $x[s:t]$ for the length-$(t-s+1)$ substring of x between indices s and t, inclusive.

3 Time-query hierarchies in property testing

In this section, we prove our hierarchy theorems (Theorems 3.3 and 3.4).

Definition 3.1 (Admissible query and time functions)**.** *The set of* admissible query and time functions *is the set of nondecreasing, unbounded functions $q, t : \mathbb{N} \to \mathbb{N}$ (query and time functions, respectively) satisfying the following conditions*[7]*: $q(n) \leq \min\left\{n, \frac{t(n)}{\log^{2.01} n}\right\}$; $q(n)$ is eventually surjective and computable in time $O(q(n)\log q(n))$; $t(n)$ is computable in time $O(t(n))$; and $t^{-1}(n)$ is computable in time $O(n)$.*

Definition 3.2 (Query and time complexity functions and classes)**.** *Given a property $\mathcal{P}$ and a constant $\varepsilon \in (0,1)$, let $Q_{\varepsilon,n}(\mathcal{P})$ and $T_{\varepsilon,n}(\mathcal{P})$ be functions of $n \in \mathbb{N}$ that denote the query and time*[8] *complexity of ε-testing $\mathcal{P}$ on inputs of length n, respectively. Given $\varepsilon \in (0,1)$ and nondecreasing functions $q(n), t(n)$, let $\mathsf{QueryTime}_\varepsilon(q(n), t(n))$ be the class of properties ε-testable by algorithms that make $O(q(n))$ queries* and *run in time $O(t(n))$.*

Theorem 3.3 (Unconditional weak hierarchy)**.** *Let $\varepsilon \in (0,1)$ be a sufficiently small constant. Let $q, t : \mathbb{N} \to \mathbb{N}$ be admissible query and time functions (as in Definition 3.1) such that $q^{-1}(n)$ is computable in time $O(t \circ q^{-1}(n))$. Then there exists a property*[9] *$\mathcal{P} \in \mathsf{QueryTime}_\varepsilon\left(q(n), 2^{t(n)^{2.01}\log^{4.05} n}\right)$ such that $Q_{\varepsilon,n}(\mathcal{P}) \geq \Omega\left(\frac{q(n)}{\log q(n)}\right)$ and $T_{\varepsilon,n}(\mathcal{P}) \geq \Omega\left(t(n)\right)$.*

Assuming SETH, we obtain a similar hierarchy, where the time complexity of the properties is more tightly characterized. Specifically, for all suitable $q(n)$ and $t(n)$, there exist properties with query complexity $\widetilde{\Theta}(q(n))$ and time complexity within a small polynomial factor of $t(n)$.

Theorem 3.4 (Strong hierarchy from SETH)**.** *Let $\varepsilon \in (0,1)$ be a sufficiently small constant and $\gamma > 0$ be any arbitrary constant. Let $q, t : \mathbb{N} \to \mathbb{N}$ be admissible query and time functions (as in Definition 3.1) such that $t(n) \leq 2^{O\left(\frac{q(n)}{\log q(n)}\right)}$ and $\log t(n) \log\log t(n)$ has slope $o(1)$. Then, under SETH, there exists a property $\mathcal{P} \in \mathsf{QueryTime}_\varepsilon\left(q(n), t(n)^{1+\gamma}\log n\right)$ such that $Q_{\varepsilon,n}(\mathcal{P}) \geq \Omega\left(\frac{q(n)}{\log q(n)}\right)$ and $T_{\varepsilon,n}(\mathcal{P}) \geq \Omega\left(\frac{t(n)}{\log^{2.01} n}\right)$.*

The condition that $\log t(n) \log\log t(n)$ has slope $o(1)$ can be viewed as a pointwise analogue of

[7] The constant 2.01 could be replaced with $2+\gamma$ for any $\gamma > 0$. This is also true for the constant 2.01 appearing in Theorem 3.3 and Theorem 3.4.

[8] The query complexity $Q_{\varepsilon,n}$ is a well-defined number for each n. In contrast, the time complexity $T_{\varepsilon,n}(\mathcal{P})$ is defined for uniform algorithms and thus only admits asymptotic bounds.

[9] We do not try to optimize the constant 2.01 in the exponent. We remark that a direct diagonalization argument in the log-cost RAM model for randomized algorithms would suffice to make the constant 1.01 instead.

$t(n) \leq 2^{o(n/\log n)}$. We argue this requirement is mild and explain its role in our arguments. The slope condition is satisfied by natural choices such as $t(n) = \operatorname{polylog} n$, $\operatorname{poly}(n)$, $2^{\operatorname{polylog} n}$, 2^{n^α} for $\alpha \in (0,1)$, and $2^{n/\log^{1+\gamma} n}$ for $\gamma > 0$. It suffices for $\log t(n) \log\log t(n)$ to be $o(n)$ and concave. This condition arises because our lower bounds use a repetition scheme that embeds many small instances (of length k) into a larger input (of length $n \gg k$), reducing from the small to the large instance. For this to work, the mapping $n \mapsto k(n)$—essentially $\log t(n) \log\log t(n)$—must be surjective. The $o(1)$ slope ensures surjectivity and allows us to efficiently compute and invert this mapping within our reductions.[10] A similar surjectivity condition appears in [GKNR12] for query complexity hierarchies. Although a weaker "dense range" requirement could work, it would complicate our reductions (e.g., requiring careful padding of k-SAT encodings). Since the current condition already covers typical $t(n)$, we prioritize simplicity.

Organization of Section 3. Section 3.1 constructs languages that are hard for randomized algorithms in our model. Section 3.2 describes codes used to map these languages to hard properties. In Section 3.3, we design a property with near-maximal query complexity that is still efficiently testable. Finally, we combine these elements to obtain the main results: we present a general construction in Section 3.4, and then apply it to prove Theorems 3.3 and 3.4 in Section 3.5.

3.1 Hard languages

In this section, we construct languages that are hard to decide, i.e., with high randomized time complexity. In Section 3.1.1, we unconditionally prove the existence of such languages in the RAM model. The best upper bound known for deciding such languages is exponential in the lower bound.[11] In Section 3.1.2, we prove the existence of a language that is hard to decide and has a nearly matching upper bound, assuming SETH.

3.1.1 Hard language in the RAM model

In this section, we construct a language that is hard to decide in the randomized RAM model. Our construction proceeds via diagonalization for the classical Turing machine (TM) model. To translate between computation models, we use [CR73, Theorem 2], stated next.

Theorem 3.5 ([CR73])**.** *If a language A is decided by a RAM within time $T(n) > n$, then A is decided by some multitape TM within time $T(n)^2$. Conversely, if a TM decides A within time $T(n) \geq n$, then some RAM decides A within time $T(n) \log T(n)$.*

Theorem 3.5 is stated only for deterministic machines. However, since coin tosses have $O(1)$ cost in both the TM and RAM model, the theorem extends to randomized computation as well. Next, we use Theorem 3.5 to construct a hard language in the RAM model.

Theorem 3.6. *Given a function $t(n) > n$ that is computable in time $O(t(n))$, there exists a language L that is a.e. nonempty, which cannot be decided by a randomized RAM within time $O(t(n))$, but can be decided by a deterministic RAM in time $O\left(2^{t(n)^{2.01}}\right)$.*

Proof. Let L be the language decided by the following procedure:

"On input x of length n, where $x = \langle M \rangle$ for some probabilistic TM M,

[10] In fact, a sufficiently small constant slope would also suffice.

[11] A randomized time hierarchy theorem would suffice to overcome this shortcoming.

- *Simulate $M(x)$ for $t(n)^{2.005}$ steps on all random branches.*
- *If M halts on all branches and outputs bit b on at least $\frac{2}{3}$ of them, output $\overline{b}$; else, output 0".*

The language L cannot be decided by an $O(t(n)^2)$-time probabilistic TM. Moreover, L is a.e. nonempty, because beyond a certain length, the TM recognizing the empty language appears at every input length, and the negation of its output is a 1. By Theorem 3.5, no $O(t(n))$-time randomized RAM decides L. However, L is decidable in time $2^{t(n)^{2.005}}$ by a deterministic TM, and thus in time $O\left(t(n)^{2.005} \cdot 2^{t(n)^{2.005}}\right) = O\left(2^{t(n)^{2.01}}\right)$ by a deterministic RAM. □

Remark 3.7. The language L is hard to decide even infinitely often for RAMs running in time $O(t(n))$, because once the machine description $\langle M \rangle$ appears in our sequence of inputs, it appears at every input length hence (due to paddability of machine descriptions), and is thus diagonalized against at every subsequent input length.

3.1.2 Hard language from SETH

Assumption 3.8 (SETH)**.** *For all $\gamma \in (0,1)$, there exists $k \in \mathbb{N}$ such that no randomized algorithm decides k-SAT instances on n variables in time $2^{(1-\gamma)n}$ with error probability at most $\frac{1}{3}$.*

By the Sparsification Lemma [IPZ01], we can assume that the k-SAT instances above have only $m = O(n)$ clauses.

Fact 3.9 (Application of the Sparsification Lemma [IPZ01])**.** *Under Assumption 3.8, for all $\gamma > 0$, there exist $k_\gamma \in \mathbb{N}$ and $C_\gamma > 0$ such that no randomized algorithm decides k_γ-SAT instances with n variables and at most $\lfloor C_\gamma n \rfloor$ clauses in time $2^{(1-\gamma)n}$ with error probability at most $\frac{1}{3}$.*

Next, we define a hard language that serves as a key ingredient in our reductions.

Definition 3.10. *Fix a binary encoding of CNF formulas and a constant $D > 0$, such that for every k-CNF formula ϕ with n variables and at most m clauses, its encoding $\langle \phi \rangle$ has $Dkm \log n$ bits. Fix $\gamma > 0$. Let $k_\gamma \in \mathbb{N}$, $C_\gamma > 0$ be the corresponding constants from Fact 3.9. For each $N \in \mathbb{N}$, let $n := \left\lfloor \frac{N}{DC_{\gamma/2} k_{\gamma/2} \log N} \right\rfloor$ and define the language $L^{(\gamma)} := \bigcup_{N \in \mathbb{N}} L_N^{(\gamma)}$, where*

$$L_N^{(\gamma)} := \left\{ \langle \phi \rangle : \phi \text{ is a satisfiable instance of } k_{\gamma/2}\text{-SAT on } n \text{ variables and } \lfloor C_{\gamma/2} n \rfloor \text{ clauses} \right\}.$$

This is well-defined because N bits suffice for the encoding by the choice of D.

Combining SETH with the naïve upper bound of $2^n \operatorname{poly}(n)$ for k-SAT gives the following.[12].

Lemma 3.11 (Bounds for hard language)**.** *For all constant $\gamma > 0$, there exists a constant $A_\gamma > 0$ such that the language $L^{(\gamma)}$ is decidable in $O\left(2^{\frac{N}{A_\gamma \log N}}\right)$ time. Furthermore, under Assumption 3.8, no randomized algorithm decides $L^{(\gamma)}$ in $2^{\frac{(1-\gamma)N}{A_\gamma \log N}}$ time with error probability at most $\frac{1}{3}$.*

[12] Although faster exponential-time randomized algorithms for k-SAT exist (e.g., [PPSZ05]), they do not help our argument, as our bounds are only tight up to constant factors in the exponent.

Proof. Let $A_\gamma := (1-\gamma/4)DC_{\gamma/2}k_{\gamma/2}$ for the constants $D, C_{\gamma/2}, k_{\gamma/2}$ from Definition 3.10. The naïve brute force algorithm decides $L^{(\gamma)}$ in time $2^n \operatorname{poly}(n)$, where $n = \left\lfloor \frac{N}{DC_{\gamma/2}k_{\gamma/2}\log N} \right\rfloor$ is the number of variables in the k-SAT instance. For all sufficiently large N, this upper bound is

$$2^n \operatorname{poly}(n) \le 2^{\frac{N}{DC_{\gamma/2}k_{\gamma/2}\log N}} \operatorname{poly}(N) = 2^{\frac{(1-\gamma/4)N}{A_\gamma \log N} + O(\log N)} \le 2^{\frac{N}{A_\gamma \log N}}.$$

On the other hand, under Assumption 3.8, by Fact 3.9, any randomized algorithm deciding $L^{(\gamma)}$ with error probability at most $\frac{1}{3}$ runs in time at least

$$2^{(1-\gamma/2)n} = 2^{(1-\gamma/2)\left\lfloor \frac{N}{DC_{\gamma/2}k_{\gamma/2}\log N} \right\rfloor} \ge 2^{(1-\gamma/2)\left(\frac{(1-\gamma//4)N}{A_\gamma \log N} - 1\right)} \ge 2^{(1-\gamma/2)\left(\frac{(1-\gamma/2)N}{A_\gamma \log N}\right)} \ge 2^{\frac{(1-\gamma)N}{A_\gamma \log N}}. \quad \square$$

3.2 Efficiently constructable, encodable, and decodable codes

Our reductions use error-correcting codes with constant rate and relative distance in two ways:

1. To transform inputs to a hard decision problem into well-separated codewords, which serve as inputs for a property that is hard to test.
2. To transform a family of hard property testing problems with an efficient *non-uniform* decision procedure into a problem with an efficient *algorithm* by encoding advice into the input. (The code ensures that distance is preserved.)

Since we care about both query and time complexity, we need codes with efficient construction, encoding, and decoding. We get this by combining the linear-time encodable and decodable codes of [Spi96] with efficient constructions of expander graphs. Specifically, we use the Zig-Zag construction of [RVW00], which, as noted in [BN13], is linear-time in the Word RAM model. In our log-cost RAM model, this yields near-linear time.

Theorem 3.12 (Adapted from [Spi96])**.** *There exist constants $r, \delta \in (0,1)$, where $\frac{1}{r} \in \mathbb{N}$, and a family $\mathcal{E} = \{\mathcal{E}_n\}_{n\in\mathbb{N}}$ of error-correcting codes such that each $\mathcal{E}_n : \{0,1\}^n \to \{0,1\}^{n/r}$ has relative distance at least δ (and rate r). Moreover, $O(n \log n)$ time suffices for constructing $\mathcal{E}_n$, encoding messages of length n, and decoding codewords of length $\frac{n}{r}$.*

In Theorem 3.12, we choose r so that $\frac{1}{r}$ is an integer, without loss of generality, as the theorem in [Spi96] holds for all sufficiently small constant r. Theorem 3.12 is obtained by replacing the Ramanujan graphs in the construction of [Spi96, Section 5] with the Zig-Zag graphs from [RVW00]. Spielman's proof (in [Spi96, Theorem 19]) requires base graphs with the second eigenvalue arbitrarily small compared to the degree d, a condition satisfied by the Zig-Zag graps, as stated next.

Theorem 3.13 (Implicit in [RVW00])**.** *For all constant $\varepsilon \in (0,1)$, there exist constant $d \in \mathbb{N}$ and an $O(n \log n)$-time algorithm that, given $n \in \mathbb{N}$, constructs a d-regular graph on $\Theta(n)$ vertices with second-largest eigenvalue at most εd.*

3.3 A property requiring $\Omega(n/\log n)$ queries and $O(n \log n)$ time to test

In this subsection, we construct a property $\mathcal{Q}$ such that testing $\mathcal{Q}_n$ has nearly maximal query complexity, but can be done in near-linear time. Our construction is based on the result of [BHR05] establishing the existence of 3CNF properties that are hard to test. They exhibit a family $\{\varphi_m\}_m$ of 3CNF formulas with $O(m)$ clauses on m variables such that testing whether an assignment

$x \in \{0,1\}^m$ satisfies φ_m requires $\Omega(m)$ queries. It is easy to check whether an assignment satisfies a *fixed* φ_m in $O(m \log m)$ time.

Fact 3.14 ([BHR05])**.** *Let $S(\varphi_m) \subset \{0,1\}^m$ be the set of satisfying assignments for a 3CNF formula φ_m on m variables and $\mathcal{S}_\varphi := \{S(\varphi_m)\}_m$ be the property obtained from a given family $\{\varphi_m\}_m$ of 3CNF formulas. There exist constants $\varepsilon_0 \in (0,1)$ and $A > 0$, and a family $\{\varphi_m\}_{m \in \mathbb{N}}$ of 3CNF formulas with at most Am clauses on m variables such that ε_0-testing $\mathcal{S}_\varphi$ requires $\Omega(m)$ queries.*

At first glance, the property $\mathcal{S}_\varphi$ appears to meet our needs. However, the issue is that the procedure for checking satisfying assignments is non-uniform: for each m, there exists a hard-to-test formula φ_m with a corresponding efficient verifier, yet no single algorithm works for all m. The formulas in [BHR05] are derived from random codes using random expander graphs with strong unique-neighbor expansion, for which no explicit (let alone near-linear time) constructions are known.[13]

To address this, we construct a new property $\mathcal{Q}$ as follows. For input length n, we define $\mathcal{Q}_n$ to contain strings formed by concatenating: (1) an encoding of a 3CNF formula ψ_m with at most Am clauses on m variables, and (2) a satisfying assignment for ψ_m. Since encoding ψ_m takes $\Theta(m \log m)$ bits, we choose $m = \Theta(n/\log n)$. To ensure that the strings obtained via this construction are far from $\mathcal{Q}_n$ when the assignment is far from $S(\psi_m)$, we encode ψ_m as $\mathcal{E}(\langle \psi_m \rangle)$, where $\mathcal{E}$ is the code from Theorem 3.12 and $\langle \psi_m \rangle$ is the binary representation of ψ_m. Because the assignment is only $\Theta(n/\log n)$ bits, we apply a repetition code to expand it to $\Theta(n)$ bits. The resulting property $\mathcal{Q}$ is hard to test (via Fact 3.14) but admits a linear-time tester that simply reads and decodes ψ_m, extracts the assignment, and checks whether it satisfies the formula.

Definition 3.15. *For all $m \in \mathbb{N}$, let Ψ_m be the set of 3CNF formulas on m variables with at most Am clauses, where A is as defined in Fact 3.14. Let $D > 0$ be a constant such that formulas in Ψ_m can be encoded in $\ell_m := \lfloor Dm \log m \rfloor$ bits.*

Definition 3.16 (Property with near-linear query and time complexity)**.** *Let $r \in (0,1)$ be the rate parameter from Theorem 3.12. Define the property $\mathcal{Q} = \{\mathcal{Q}_n\}_{n \in \mathbb{N}}$ as follows. Let $n \in \mathbb{N}$ be sufficiently large so that $m := \left\lfloor \frac{nr}{2D \log n} \right\rfloor \geq 1$. For each 3CNF formula $\psi_m \in \Psi_m$ with encoding $\langle \psi_m \rangle$ of length ℓ_m, and each satisfying assignment $x \in S(\psi_m)$, let $n' := n - \lfloor \ell_m / r \rfloor$, $t := \lceil n'/m \rceil$, and $y(x) := x^t[1:n']$. Finally, let*

$$\mathcal{Q}_n := \{\mathcal{E}(\langle \psi_m \rangle) \circ y(x) : \ \psi_m \in \Psi_m \text{ and } x \in S(\psi_m)\} ,$$

where $\mathcal{E}$ is the code from Theorem 3.12. For small $n \in \mathbb{N}$ yielding $m = 0$, let $\mathcal{Q}_n := \emptyset$.

We now show that $\mathcal{Q}_n$ satisfies our requirements on query and time complexity.

Lemma 3.17 (Query lower bound)**.** *Let $\varepsilon > 0$ be a sufficiently small constant. Then ε-testing property $\mathcal{Q}$ from Definition 3.16 requires $\Omega(n/\log n)$ queries.*

Proof. We reduce testing $\mathcal{S}_\varphi$ (on inputs of length m obtained from n as in Definition 3.16) to testing $\mathcal{Q}_n$. Suppose $\mathcal{T}$ is an ε-tester for $\mathcal{Q}_n$. We give a 4ε-tester $\mathcal{A}$ for $\mathcal{S}_\varphi$ that works as follows. On input

[13] Explicit constructions of related structures, like lossless expanders (see e.g., [CRVW02]), have been extensively studied. However, [BHR05] relies on two-sided expansion, which is generally harder to achieve and remains an active area of research [CGRZ24, HMMP24, HLM+25, Che25].

$x \in \{0,1\}^m$, tester $\mathcal{A}$ computes $\mathcal{E}(\langle\varphi_m\rangle)$ for formula φ_m from Fact 3.14 using no queries, and then simulates $\mathcal{T}$ on $z := \mathcal{E}(\langle\varphi_m\rangle) \circ y(x)$, using at most one query to x per query of $\mathcal{T}$ to z. The query complexity of $\mathcal{A}$ is at most that of $\mathcal{T}$. So, if $\mathcal{A}$ is a valid tester for $\mathcal{S}_\varphi$, then by Fact 3.14, $\mathcal{T}$ must make $\Omega(m) = \Omega(n/\log n)$ queries.

First, suppose $x \in \mathcal{S}_\varphi$. Then, by construction, $z \in \mathcal{Q}_n$, so $\mathcal{A}$ accepts with probability at least $\frac{2}{3}$.

Now suppose x is 4ε-far from $\mathcal{S}_\varphi$. We claim that z is ε-far from $\mathcal{Q}_n$. Indeed, let z^* be a closest string to z in $\mathcal{Q}_n$. Then $z^* = \mathcal{E}(\langle\psi_m^*\rangle) \circ y(x^*)$ for some $\psi_m^* \in \Psi_m$ and some satisfying assignment $x^* \in S(\psi_m^*)$. We claim that $\mathsf{dist}(z, z^*) \geq \varepsilon$, and consider two cases to prove the claim. First, suppose $\varphi_m \neq \psi_m^*$. Then $\mathsf{dist}(\mathcal{E}(\langle\varphi_m\rangle), \mathcal{E}(\langle\psi_m^*\rangle)) \geq \delta$, where δ is the distance of the code from Theorem 3.12. Since $\lfloor \ell_m/r \rfloor = \Omega(n)$ by Definition 3.15, it follows that $\mathsf{dist}(z, z^*) \geq \varepsilon$ as long as ε is smaller than δ by a sufficiently small constant factor. Second, suppose $\varphi_m = \psi_m^*$. Then $\mathsf{dist}(x, x^*) \geq 4\varepsilon$ because x is 4ε-far from $\mathcal{S}_\varphi$ while $x^* \in \mathcal{S}_\varphi$, and hence $\mathsf{dist}(y(x), y(x^*)) \geq 2\varepsilon$ (the factor of 2 accounts for a possibly partial last repetition of x inside $y(x)$, and similarly for x^*). But $|y(x)| = |y(x^*)| = n - \lfloor \ell_m/r \rfloor \geq n/2$, so $\mathsf{dist}(z, z^*) \geq \varepsilon$. Hence z is ε-far from $\mathcal{Q}_n$, so $\mathcal{A}$ rejects with probability at least $\frac{2}{3}$. □

Lemma 3.18 (Time upper bound)**.** *There exists an $O(n \log n)$-time decider (i.e., a 0-tester) for $\mathcal{Q}$.*

Proof. Let $\mathcal{T}$ be a tester that, on input z, queries all of z, computes m and ℓ_m from Definitions 3.15 and 3.16, and tries to use the decoder from Theorem 3.12 to compute a decomposition $z = \mathcal{E}(\langle\psi_m\rangle) \circ y(x)$. Tester $\mathcal{T}$ rejects if decoding the first $\lfloor \ell_m/r \rfloor$ bits fails or does not yield a valid formula in Ψ_m, or if the remaining bits are not of the form $y(x)$ for some $x \in \{0,1\}^m$. Otherwise, $\mathcal{T}$ accepts iff the assignment x satisfies the formula ψ_m.

Since querying the entire input, decoding the code from Theorem 3.12, and checking whether the assignment x satisfies ψ_m all take $O(n \log n)$ time, tester $\mathcal{T}$ runs in time $O(n \log n)$.

We claim that $\mathcal{T}$ answers correctly with probability 1. Clearly, $\mathcal{T}$ accepts if $z \in \mathcal{Q}_n$. Now, suppose $z \notin \mathcal{Q}_n$. Suppose z is of the form $\mathcal{E}(\langle\psi_m\rangle) \circ y(x)$, since otherwise $\mathcal{T}$ rejects. Then $x \notin S(\psi_m)$ (since otherwise z would be in $\mathcal{Q}_n$), and hence $\mathcal{T}$ rejects, as desired. □

3.4 Construction

We start by defining operations on properties and summarizing their features.

3.4.1 Concatenated properties

To prove the time-query hierarchy theorems, we first construct properties achieving the desired query and time bounds separately. The following definitions and lemmas show how to combine them to obtain both guarantees simultaneously.

Definition 3.19 (Concatenation of properties)**.** *Let $n_1, n_2 \in \mathbb{N}$. Let $\mathcal{P}_{n_1}^{(1)}$ and $\mathcal{P}_{n_2}^{(2)}$ be properties over inputs of length n_1 and n_2, respectively. Then their* concatenation $\mathsf{Concat}(\mathcal{P}_{n_1}^{(1)}, \mathcal{P}_{n_2}^{(2)})$ *is the property over inputs of length $n_1 + n_2$ given by* $\mathsf{Concat}(\mathcal{P}_{n_1}^{(1)}, \mathcal{P}_{n_2}^{(2)}) := \left\{ xy : x \in \mathcal{P}_{n_1}^{(1)}; y \in \mathcal{P}_{n_2}^{(2)} \right\}$.

Definition 3.20 (YES query provider)**.** *Let $t : \mathbb{N} \to \mathbb{N}$ be a function, possibly sublinear in n. A $t(n)$-time* YES query provider *for a property $\mathcal{P}$ is an algorithm $\mathcal{A}$ that, given $\langle n \rangle$ for which $\mathcal{P}_n \neq \emptyset$,*

runs in $t(n)$ time to produce an implicit representation of $x^{(n)} \in \mathcal{P}_n$, and then answers each query $i \in [n]$ with $x_i^{(n)}$ in $O(\log^{1+o(1)} n)$ time.[14]

We summarize features of properties obtained via concatenation using notation from Definition 3.2.

Lemma 3.21 (Concatenation Lemma)**.** *Given a.e. nonempty properties $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$, define property $\mathcal{P}$ by setting $\mathcal{P}_{2n} := \mathsf{Concat}(\mathcal{P}_n^{(1)}, \mathcal{P}_n^{(2)})$ and $\mathcal{P}_{2n-1} := \emptyset$ for each $n \in \mathbb{N}$. Let $\varepsilon \in (0,1)$. Then the following hold.*

1. $Q_{\varepsilon,2n}(\mathcal{P}) = O\left(Q_{\varepsilon,n}(\mathcal{P}^{(1)}) + Q_{\varepsilon,n}(\mathcal{P}^{(2)})\right)$ *and* $Q_{\varepsilon/2,2n}(\mathcal{P}) = \Omega\left(Q_{\varepsilon,n}(\mathcal{P}^{(1)}) + Q_{\varepsilon,n}(\mathcal{P}^{(2)})\right)$.
2. $T_{\varepsilon,2n}(\mathcal{P}) = O\left(\left(T_{\varepsilon,n}(\mathcal{P}^{(1)}) + T_{\varepsilon,n}(\mathcal{P}^{(2)})\right)\log n\right)$.
3. *Suppose $T_{\varepsilon/2,2n}(\mathcal{P}) = O(t(n))$. For all $i \in [2]$, if $\mathcal{P}_n^{(i)}$ has an $O(t(n)\log^{1+o(1)} n)$-time YES query provider, then $T_{\varepsilon,n}(\mathcal{P}^{(3-i)}) = O(t(n)\log^{1+o(1)} n)$.*

Proof. **Item 1.** We prove the upper and lower bounds separately.

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be ε-testers for $\mathcal{P}_n^{(1)}$ and $\mathcal{P}_n^{(2)}$ that have error probability at most $\frac{1}{6}$. Construct the following ε-tester $\mathcal{A}$ for $\mathcal{P}_{2n}$. On input xy, where $|x| = |y| = n$, algorithm $\mathcal{A}$ runs $\mathcal{A}_1$ on x and $\mathcal{A}_2$ on y, accepting iff both $\mathcal{A}_1$ and $\mathcal{A}_2$ accept. We show that this is a valid ε-tester for $\mathcal{P}_{2n}$. If input $xy \in \mathcal{P}_{2n}$, then $\mathcal{A}_1$ and $\mathcal{A}_2$ accept x and y, respectively, with probability at least $\frac{5}{6}$ each, so $\mathcal{A}$ accepts with probability at least $\frac{2}{3}$ (by a union bound). If input xy is ε-far from $\mathcal{P}_{2n}$, then, by an averaging argument, x or y must be ε-far from $\mathcal{P}_n^{(1)}$ or $\mathcal{P}_n^{(2)}$, respectively. Thus, $\mathcal{A}_1$ or $\mathcal{A}_2$ rejects (and then so does $\mathcal{A}$) with probability at least $\frac{5}{6}$. By standard probability amplification, there exist algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$, as specified above, that make $O(Q_{\varepsilon,n}(\mathcal{P}^{(1)}))$ and $O(Q_{\varepsilon,n}(\mathcal{P}^{(2)}))$ queries, respectively. The query complexity of $\mathcal{A}$ constructed from such $\mathcal{A}_1$ and $\mathcal{A}_2$ is $O\left(Q_{\varepsilon,n}(\mathcal{P}^{(1)}) + Q_{\varepsilon,n}(\mathcal{P}^{(2)})\right)$.

Given an $\frac{\varepsilon}{2}$-tester $\mathcal{T}$ for $\mathcal{P}_{2n}$, we construct an ε-tester $\mathcal{T}_1$ for $\mathcal{P}_n^{(1)}$ as follows: on input $x \in \{0,1\}^n$, algorithm $\mathcal{T}_1$ fixes any $y \in \mathcal{P}_n^{(2)}$ (which exists since $\mathcal{P}^{(2)}$ is a.e. nonempty) and simulates $\mathcal{T}$ on xy. If $x \in \mathcal{P}_n^{(1)}$, then $xy \in \mathcal{P}_{2n}$, so $\mathcal{T}_1$ accepts with probability at least $\frac{2}{3}$. If x is ε-far from $\mathcal{P}_n^{(1)}$, then xy is $\frac{\varepsilon}{2}$-far from $\mathcal{P}_{2n}$, so $\mathcal{T}_1$ rejects with probability at least $\frac{2}{3}$. Since the query complexity of $\mathcal{T}_1$ is at most that of $\mathcal{T}$, we conclude that $\mathcal{T}$ makes at least $Q_{\varepsilon,n}(\mathcal{P}^{(1)})$ queries. Analogously, it also makes at least $Q_{\varepsilon,n}(\mathcal{P}^{(2)})$ queries. Thus, $\frac{\varepsilon}{2}$-testing $\mathcal{P}_{2n}$ has query complexity $\Omega\left(Q_{\varepsilon,n}(\mathcal{P}^{(1)}) + Q_{\varepsilon,n}(\mathcal{P}^{(2)})\right)$.

Item 2. By standard probability amplification, there exist algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$, as specified in the previous part, with running time $O(T_{\varepsilon,n}(\mathcal{P}^{(1)}))$ and $O(T_{\varepsilon,n}(\mathcal{P}^{(2)}))$, respectively. The running time of $\mathcal{A}$ constructed from such $\mathcal{A}_1$ and $\mathcal{A}_2$ (allowing for simulation overhead, including a potential $\log n$ blow-up in the cost of queries by the combined tester[15]) is $O\left(\left(T_{\varepsilon,n}(\mathcal{P}^{(1)}) + T_{\varepsilon,n}(\mathcal{P}^{(2)})\right)\log n\right)$.

Item 3. Let $\mathcal{T}$ be an $\frac{\varepsilon}{2}$-tester for $\mathcal{P}_{2n}$ running in time $O(t(n))$ and $\mathcal{B}$ be an $O(t(n)\log^{1+o(1)} n)$-time YES query provider for $\mathcal{P}_n^{(2)}$. We obtain an ε-tester $\mathcal{T}_1$ for $\mathcal{P}_n^{(1)}$ as follows. On input $x \in \{0,1\}^n$,

[14] Formally, we can define this via two algorithms, Preprocess and Query: Preprocess, given $\langle n \rangle$ on the parameter tape, runs in $t(n)$ time and puts $\mathcal{O} \in \{0,1\}^*$ to registers; Query, given $\langle i, \mathcal{O} \rangle$, computes $x_i^{(n)}$ in $O(\log^{1+o(1)} n)$ time, where $x^{(n)} \in \mathcal{P}_n$ is uniquely determined by $\mathcal{O}$. For clarity, we use the "interactive" version instead.

[15] Specifically, tester $\mathcal{A}_2$ may have low-indexed queries to y which are inexpensive, but those queries still cost $O(\log n)$ for the tester simulating it. While this blow-up does not occur in queries to x by $\mathcal{A}_1$, we allow a global $\log n$ overhead in our upper bound for simplicity. We do not incur a $\log t_1$ or $\log t_2$ overhead for simulating *register* access by either algorithm, since we can interleave the registers used by $\mathcal{A}_1$ and $\mathcal{A}_2$ in our construction of $\mathcal{A}$.

tester $\mathcal{T}_1$ uses $\mathcal{B}$ to fix some $y \in \mathcal{P}_n^{(2)}$ (which exists since $\mathcal{P}^{(2)}$ is a.e. nonempty) and simulates $\mathcal{T}$ on xy: when $\mathcal{T}$ queries a bit from xy, tester $\mathcal{T}_1$ either queries a bit from x or uses $\mathcal{B}$ to query a bit from y in $O(\log^{1+o(1)} n)$ time. Then, as in Item 1, algorithm $\mathcal{T}_1$ is an ε-tester for $\mathcal{P}^{(1)}$. By construction, the running time of $\mathcal{T}_1$ is $O(t(n) \log^{1+o(1)} n)$. An identical argument works for $\mathcal{T}_2$. □

Remark 3.22. For **Item 3** of Lemma 3.21, if the tester for $\mathcal{P}$ succeeds only on infinitely many input lengths (as opposed to the usual notion of almost every input length), then we get testers for $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ that also succeed on infinitely many input lengths.

3.4.2 Intermediate complexity regimes by repetition

We first construct properties with maximal query or time complexity, and then obtain intermediate complexities using instance repetition, as in [GKNR12, KLR25].

Definition 3.23 (Repeated instances [GKNR12])**.** *Given a property $\mathcal{P}$ and a function $k : \mathbb{N} \to \mathbb{N}$ satisfying $k(n) \leq n$, define the* repeated instances *property $\mathcal{P}^{(k)}$ by*

$$\mathcal{P}_n^{(k)} := \left\{ x^r[1:n] : \ x \in \mathcal{P}_{k(n)} \text{ and } r = \left\lceil \tfrac{n}{k(n)} \right\rceil \right\}.$$

We summarize features of properties obtained via repetition using notation from Definition 3.2.

Definition 3.24. *Given a property $\mathcal{P}$, we write* $\mathrm{LenSupp}(\mathcal{P}) := \{n \in \mathbb{N} : \mathcal{P}_n \neq \emptyset\}$. *A set $S \subseteq \mathbb{N}$ is* decidable *in time $t : \mathbb{N} \to \mathbb{R}$ if there is an algorithm which, on input $n \in \mathbb{N}$ given as $\langle n \rangle$, halts in time $O(t(n))$ and accepts iff $n \in S$. Given a function f, we write* $\mathrm{Im}(f)$ *for the image of f.*

Observation 3.25. *If $f : \mathbb{N} \to \mathbb{N}$ is eventually surjective, then* $\mathrm{Im}(f)$ *is decidable in time $O(1)$.*

Lemma 3.26 (Repeated instances lemma)**.** *Given a property $\mathcal{P}$ and a function $k : \mathbb{N} \to \mathbb{N}$ satisfying $k(n) \leq n$ with a right inverse $k^\dagger$ and such that* $\mathrm{LenSupp}(\mathcal{P}) \setminus \mathrm{Im}(k)$ *is finite, consider the repeated instances property $\mathcal{P}^{(k)}$. Let $\varepsilon \in (0,1)$ be a constant. The following are true.*

1. *[GKNR12] If $Q_{\varepsilon/2,n}(\mathcal{P}) = q(n)$ then $Q_{\varepsilon,n}\left(\mathcal{P}^{(k)}\right) = O\left(q(k(n))\right)$ and $Q_{\varepsilon/4,n}\left(\mathcal{P}^{(k)}\right) = \Omega\left(q(k(n))\right)$.*
2. *If $T_{\varepsilon/2,n}(\mathcal{P}) = O(t(n))$ and $k(n)$ is computable in $O(t(k(n)))$ time, then $T_{\varepsilon,n}\left(\mathcal{P}^{(k)}\right) = O(t(k(n)) + \log^{1+o(1)} n)$.*
3. *Suppose $T_{\varepsilon/2,n}\left(\mathcal{P}^{(k)}\right) = O(t(k(n)))$. For all $\gamma > 0$, if $\mathrm{Im}(k)$ is decidable and $k^\dagger(n)$ computable both in $O(t(n) \log^{1+\gamma} k^\dagger(n))$ time, then $T_{\varepsilon,n}(\mathcal{P}) = O(t(n) \log^{1+\gamma} k^\dagger(n))$.*

Proof. **Item 1.** We prove the upper and lower bounds separately.

Let $\mathcal{A}$ be an $\frac{\varepsilon}{2}$-tester for $\mathcal{P}$. Construct the following ε-tester $\mathcal{A}'$ for $\mathcal{P}^{(k)}$. First, $\mathcal{A}'$ repeats the following $O(1/\varepsilon)$ times: uniformly select a $j \in [k(n)]$ and $r \in [(n/k(n)) - 1]$ and check whether $x[r \cdot k(n) + j] = x[j]$. If the check succeeds, $\mathcal{A}'$ simulates $\mathcal{A}$ on $x[1 : k(n)]$. If $x \in \mathcal{P}_n^{(k)}$, then the first test always passes and $\mathcal{A}$ accepts with probability at least $\frac{2}{3}$, so $\mathcal{A}'$ accepts with probability at least $\frac{2}{3}$. Suppose x is ε-far from $\mathcal{P}_n^{(k)}$. If $x[1 : k(n)]$ is less than $\frac{\varepsilon}{2}$-far from $\mathcal{P}_{k(n)}$ and x is less than $\frac{\varepsilon}{2}$-far from being a repeated instance of $x[1 : k(n)]$, then x is less than ε-far from $\mathcal{P}_{k(n)}$. Thus, with probability at least $\frac{2}{3}$, at least one of the checks fails and $\mathcal{A}'$ rejects. If $\mathcal{A}$ makes $q(n)$ queries, then $\mathcal{A}'$ makes $q(k(n)) + O(1/\varepsilon)$ queries, which is $O(q(k(n)))$ since ε is a constant.

Given an $\frac{\varepsilon}{4}$-tester $\mathcal{T}$ for $\mathcal{P}_n^{(k)}$, we obtain an $\frac{\varepsilon}{2}$-tester $\mathcal{T}'$ for $\mathcal{P}$ as follows. On input x of sufficiently large length n, tester $\mathcal{T}'$ first checks whether $n \in \mathrm{Im}(k)$, and rejects if this is not the case. Since $\mathrm{LenSupp}(\mathcal{P}) \setminus \mathrm{Im}(k)$ is finite, this only ignores finitely many nontrivial input lengths. Otherwise, $\mathcal{T}'$ computes $n' = k^\dagger(n)$, which satisfies $k(n') = n$. Tester $\mathcal{T}'$ then creates the instance $y = x^{\lceil n'/n \rceil}[1 : n']$ and simulates $\mathcal{T}$ on y. If $x \in \mathcal{P}_n$, then $y \in \mathcal{P}_{n'}^{(k)}$, and $\mathcal{T}'$ accepts. If x is $\frac{\varepsilon}{2}$-far from $\mathcal{P}_n$, then y is $\frac{\varepsilon}{4}$-far from $\mathcal{P}_{n'}^{(k)}$ (the factor of 2 accounts for a possibly partial last repetition of x inside y), and $\mathcal{T}'$ rejects. If $\mathcal{T}$ makes $q'(n)$ queries, then $\mathcal{T}'$ makes $q'(n')$ queries, implying $q'(n') \geq q(n) = q(k(n'))$ for infinitely many $n' \in \mathbb{N}$.

Item 2. We use the same construction of $\mathcal{A}'$ as in the first part of Item 1. Correctness thus follows immediately, and we only have to prove the runtime. Computing $k(n)$ takes time $O(t(k(n))$, and then for each of the $O(1/\varepsilon)$ queries to check repetition, we incur a cost of $O(\log n)$ for sampling random bits, a cost of $O(\log^{1+o(1)} n)$ for computing the index modulo $k(n)$ (note that $k(n) \leq n$ here), and a cost of $O(\log n)$ to make the actual queries. If $\mathcal{A}$ runs in $O(t(n))$ time, then its simulation takes $O(t(k(n)))$ time. In total, the running time of $\mathcal{A}'$ is $O(t(k(n)) + \log^{1+o(1)} n)$

Item 3. We use the same construction of $\mathcal{T}'$ as in the second part of Item 1. Correctness thus follows immediately, and we only have to prove the runtime. By assumption, checking whether $n \in \mathrm{Im}(k)$ and subsequently computing $n' = k^\dagger(n)$ takes $O(t(n) \log^{1+\gamma k^\dagger(n)})$ time. For each query made by $\mathcal{T}$, tester $\mathcal{T}'$ requires $O(\log^{1+o(1)} n')$ time to convert it to the corresponding query in x via integer division. If $\mathcal{T}$ runs in time $O(t(k(n)))$, then $\mathcal{T}'$ runs in time $O(t(k(n')) \log^{1+o(1)} n' + t(n) \log^{1+\gamma} k^\dagger(n)) = O(t(n) \log^{1+\gamma} k^\dagger(n))$ time, as desired. □

Remark 3.27. For **Items 1** and **2** of Lemmas 3.21 and 3.26, if the same testers $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}$ achieve both the time and query upper bounds for $\mathcal{P}^{(1)}$, $\mathcal{P}^{(2)}$, and $\mathcal{P}$, respectively, then the constructed testers $\mathcal{A}$ and $\mathcal{A}'$ also achieve the time and query upper bounds simultaneously.

Remark 3.28. For **Item 3** of Lemma 3.26, if the tester for $\mathcal{P}^{(k)}$ succeeds only on infinitely many input lengths and $\mathrm{Im}(k) \Delta \mathrm{LenSupp}(\mathcal{P})$ is finite, then we get a tester for $\mathcal{P}$ that succeeds on an infinite subsequence of $\mathrm{LenSupp}(\mathcal{P})$.

3.4.3 Languages to properties

To construct properties with a target time complexity, we first start with a language that is hard to *decide*, and then map it through a suitable code so that YES and NO instances of the language are ε-far, meaning *testing* for this version of the language is as hard as *deciding* the original hard language. Below, we formalize this construction.

Definition 3.29 (Properties from languages)**.** *Let $L \subset \{0,1\}^*$ be a language. Define the property $\mathcal{C} = \mathcal{E}(L)$ by $\mathcal{C} := \{\mathcal{E}(x) : x \in L\}$, where $\mathcal{E}$ is the code from Theorem 3.12.*

The following lemma translates the computational complexity of deciding language L to that of testing property $\mathcal{E}(L)$.

Lemma 3.30. *Let $L \subset \{0,1\}^*$ be a language, and code $\mathcal{E}$ and rate r be as in Theorem 3.12. Let $\mathcal{C} := \mathcal{E}(L)$. Then for all sufficiently small constant $\varepsilon > 0$, the following statements hold:*

1. *If L is decidable with error probability at most $\frac{1}{3}$ in $O(t(n))$ time, then there exists an ε-tester for $\mathcal{C}$ using n queries and $O(n \log n + t(\lfloor r \cdot n \rfloor))$ time.*

2. *If ε-testing $\mathcal{C}$ can be done in $O(t(\lceil r \cdot n \rceil))$ time, then there exists a decider for L with error probability at most $\frac{1}{3}$ running in time $O(n \log n + t(n))$.*

Proof. **Item 1.** Let $\mathcal{A}$ be an $O(t(n))$-time randomized algorithm for deciding L. We give a decider (i.e., 0-tester) $\mathcal{T}$ which works as follows. On input w of length n, tester $\mathcal{T}$ immediately rejects if n is not an integer multiple of $\frac{1}{r}$. Otherwise, $\mathcal{T}$ runs the decoder from Theorem 3.12 on w. If w is not a valid codeword, $\mathcal{T}$ rejects. Otherwise, if there is a message x (with $|x| = r \cdot n = \lfloor r \cdot n \rfloor$) such that $\mathcal{E}(x) = w$, the tester simulates the algorithm $\mathcal{A}$ on input x and accepts iff $\mathcal{A}$ accepts.

Correctness and query complexity follow directly from the construction. For the time complexity, checking the divisibility of n by $\frac{1}{r}$ takes time $O(\log^{1+o(1)} n)$; querying all of w and decoding it both take time $O(n \log n)$; and running algorithm $\mathcal{A}$ takes time $O(t(|x|)) = O(t(\lfloor r \cdot n \rfloor))$.

Item 2. Let $\mathcal{T}$ be an ε-tester with running time $O(t'(n))$. We give a randomized algorithm $\mathcal{A}$ for deciding L as follows. On input x of length m, the algorithm first computes $n := \frac{m}{r}$ and runs the encoder from Theorem 3.12 to obtain string $w := \mathcal{E}(x)$ of length n. Then, $\mathcal{A}$ simulates the tester $\mathcal{T}$ on input w, and accepts iff $\mathcal{T}$ accepts.

We first show correctness. When $x \in L$, we have $w \in \mathcal{C}_n$ by the construction of $\mathcal{C}$, so $\mathcal{T}$ accepts with probability at least $\frac{2}{3}$, and so does $\mathcal{A}$. When $x \notin L$, the string w is ε-far from every other codeword for sufficiently small ε by Theorem 3.12, so in particular w is ε-far from $\mathcal{C}$. Hence $\mathcal{T}$ rejects with probability at least $\frac{2}{3}$, and so does $\mathcal{A}$.

We now analyze the time complexity of $\mathcal{A}$. Computing $w = \mathcal{E}(x)$ takes time $O(m \log m)$ by Theorem 3.12, and simulating $\mathcal{T}$ takes times $O(t'(n)) = O(t'(m/r))$. If $t'(n) = O(t(\lceil r \cdot n \rceil))$, then this is at most $O(t(m))$. Hence the overall running time of $\mathcal{A}$ is $O(m \log m + t(m))$. □

Remark 3.31. For **Item 2** of Lemma 3.30, if the tester for $\mathcal{C}$ succeeds on infinitely many input lengths that are multiples of $\frac{1}{r}$, then we get a decider for L that succeeds on infinitely many input lengths.

3.4.4 Combining the constructions

The next definition is used to produce problem instances with prescribed time complexity. Suppose a hard problem P has time complexity $T(k)$ for inputs of length k, with $T_\ell(k) \le T(k) \le T_u(k)$. Given a target function $t(n)$, we aim to construct, for each n, an instance of P of length $k = k(n)$ requiring time $T(k) \approx t(n)$ to solve. If T were invertible, we could take $k(n) = T^{-1}(t(n))$, but since this may not hold, we use the following definition.

Definition 3.32 (Attainment)**.** *Let $F, T_\ell, T_u, t : \mathbb{N} \to \mathbb{R}$ be nondecreasing and $k : \mathbb{N} \to \mathbb{N}$ be a function We say the pair (T_ℓ, T_u)* attains *t up to gap F with length function k if, for all large enough $n \in \mathbb{N}$, we have $T_\ell(k(n)) \ge t(n)$ and $T_u(k(n)) \le F(t(n))$.*

The next definition gives the "recipe" by which, given a property $\mathcal{Q}$ with nearly maximal query complexity and a language L that is hard to decide, as well as target query and time complexity functions, we construct a property attaining the desired query and time complexities. We then obtain our hierarchy theorems by plugging in specific choices for $\mathcal{Q}$ and L.

Definition 3.33 (Hard property)**.** *Let $\mathcal{Q} \subset \{0,1\}^*$ be an a.e. nonempty property satisfying the following conditions.*

1. $T_{\varepsilon,n}(\mathcal{Q}) = O(n \log n)$ *for all constant* $\varepsilon \in (0,1)$ *and* $Q_{\varepsilon,n}(\mathcal{Q}) = \Omega\left(\frac{n}{\log n}\right)$ *for some sufficiently small constant* $\varepsilon > 0$.

2. $\mathcal{Q}$ *has an* $O(n \log n)$*-time YES query provider (Definition 3.20).*

Let $T_\ell, T_u : \mathbb{N} \to \mathbb{R}$ *be* $\Omega(n \log n)$ *nondecreasing functions satisfying* $T_\ell(n) \le T_u(n)$ *for all large enough* $n \in \mathbb{N}$*, and let* $L \subset \{0,1\}^*$ *be an a.e. nonempty language satisfying the following condition.*

3. L *is decidable with error probability at most* $\frac{1}{3}$ *by an* $O(T_u(n))$*-time randomized algorithm, but no* $O(T_\ell(n))$*-time randomized algorithm.*

Let $q : \mathbb{N} \to \mathbb{N}$ *and* $t, F : \mathbb{N} \to \mathbb{R}$ *be nondecreasing functions and* $k_* : \mathbb{N} \to \mathbb{N}$ *be a function such that the following conditions hold.*

4. $q(n)$ *is eventually surjective, and satisfies* $q(n) \le n$ *for all* n *and* $q(n) \log q(n) = O\left(\frac{t(n)}{\log^{1.01} n}\right)$*. Moreover,* $q(n)$ *is computable in* $O(q(n) \log q(n))$ *time.*

5. *The pair* (T_ℓ, T_u) *attains* t *up to gap* $F : \mathbb{N} \to \mathbb{R}$ *with eventually surjective length function* k_* *(Definition 3.32).*

6. $k_*(n) = O(q(n))$ *and* $k_*(n) \le r \cdot n$*, where* $r \in (0,1)$ *is the constant from Theorem 3.12. Moreover,* $k_*(n)$ *is computable in time* $O(T_u(k_*(n)))$ *and has a right inverse* ${k_*}^\dagger$ *such that* ${k_*}^\dagger(n)$ *is computable in time* $O(T_\ell(n))$*.*

Let $\mathcal{C} := \mathcal{E}(L)$ *be the property from Definition 3.29, and define* $k : \mathbb{N} \to \mathbb{N}$ *by* $k(n) := \frac{k_*(n)}{r}$*. Define the* hard property $\mathcal{P}$ *as follows: for each* $n \in \mathbb{N}$*, let* $\mathcal{P}_{2n-1} := \emptyset$ *and* $\mathcal{P}_{2n} := \mathsf{Concat}\left(\mathcal{Q}_n^{(q)}, \mathcal{C}_n^{(k)}\right)$.

The following lemma bounds the complexity of testing the hard property from Definition 3.33.

Lemma 3.34 (Complexity of testing the hard property)**.** *Let* $\varepsilon > 0$ *be a sufficiently small constant. Let* $\mathcal{Q}$ *be a property,* L *be a language, and* $q : \mathbb{N} \to \mathbb{N}$ *and* $t, F, T_\ell, T_u : \mathbb{N} \to \mathbb{R}$ *be functions satisfying Definition 3.33 with theresulting property* $\mathcal{P}$*. Then the following statements hold:*

1. ε*-testing* $\mathcal{P}_{2n}$ *requires* $\Omega\left(\frac{q(n)}{\log q(n)}\right)$ *queries and* $\Omega\left(\frac{t(n)}{\log^{2.01}(n)}\right)$ *time.*

2. *There exists an* ε*-tester for* $\mathcal{P}_{2n}$ *using* $O(q(n))$ *queries and* $O(q(n) \log^2 n + F(t(n)) \log n + \log^{2+o(1)} n)$ *time.*

Proof. We prove the lower and upper bounds separately.

Item 1. First, we show the query complexity lower bound. By Lemma 3.26, using the assumption that q is eventually surjective and the assumed query complexity lower bound for testing $\mathcal{Q}_n$ (Definition 3.33.1), $Q_{2\varepsilon,n}\left(\mathcal{Q}^{(q)}\right) = \Omega\left(\frac{q(n)}{\log(q(n))}\right)$. Also, $\mathcal{Q}^{(q)}$ is a.e. nonempty by Definition 3.23 and the assumption that $\mathcal{Q}$ is a.e. nonempty. Since L is a.e. nonempty, so is $\mathcal{C}^{(k)}$ by Definitions 3.23 and 3.29 and because $k(n)$ is an integer multiple of $\frac{1}{r}$. Thus, $Q_{\varepsilon,2n}(\mathcal{P}) = \Omega\left(\frac{q(n)}{\log(q(n))}\right)$ by Lemma 3.21.1.

Next, we show the time complexity lower bound. Since by assumption (Definition 3.33.3), L cannot be decided by any randomized $O(T_\ell(n))$-time algorithm and $T_\ell(n) = \Omega(n \log n)$, Lemma 3.30.2 implies that there is no $O(T_\ell(\lceil r \cdot n \rceil))$-time tester for $\mathcal{C}$.

We claim this implies, via Lemma 3.26.3, that there is no $O(T_\ell(r \cdot k(n))/\log^{1.005} n)$-time tester for $\mathcal{C}_n^{(k)}$. First, note that $k(n) \le n$ since $k(n) = k_*(n)/r$ and $k_*(n) \le r \cdot n$ (Definition 3.33.6). Also,

$\mathrm{LenSupp}(\mathcal{C}_n) \setminus \mathrm{Im}(k)$ is finite because $\mathrm{LenSupp}(\mathcal{C}_n)$ only contains integer multiples of $1/r$ by Definitions 3.23 and 3.29, while $\mathrm{Im}(k)$ contains a.e. integer multiple of $1/r$ because $k_*(n)$ is eventually surjective. Since ${k_*}^\dagger(n)$ is computable in time $O(T_\ell(n))$ by Definition 3.33.6, a right inverse $k^\dagger(n)$ of k is computable (by first performing an integer division by $1/r$) in time $O(\log^{1+o(1)}(n)+T_\ell(\lfloor r \cdot n \rfloor)) = O(T_\ell(\lfloor r \cdot n \rfloor))$, and similarly $\mathrm{Im}(k)$ is also decidable in this time. Hence applying Lemma 3.26.3 with parameter $\gamma = 0.005$ and function $t'(m) = T_\ell(\lfloor r \cdot m \rfloor)/\log^{1.005} k^{-1}(m)$ yields that there is no $O(T_\ell(r \cdot k(n))/\log^{1.005} n)$-time tester for $\mathcal{C}_n^{(k)}$.

Since $T_\ell(r \cdot k(n)) = T_\ell(k_*(n)) \geq t(n)$ by attainment (Definition 3.32), we conclude that there is no $O(t(n)/\log^{1.005} n)$-time tester for $\mathcal{C}_n^{(k)}$. By assumption (Definition 3.33.2), $\mathcal{Q}$ has an $O(n \log n)$-time YES query provider, which we can use together with the computability of $q(n)$ in $O(q(n) \log q(n))$ time to derive an $O(q(n) \log q(n))$-time YES query provider for $\mathcal{Q}_n^{(q)}$. Because $O(q(n) \log q(n)) = O(t(n)/\log^{1.01} n)$ by assumption (Definition 3.33.4), an $O(t(n)/\log^{2.01} n)$-time tester for $\mathcal{P}_{2n}$ would yield by Lemma 3.21.3 an $O(t(n)/\log^{1.005} n)$-time tester for $\mathcal{C}_n^{(k)}$, which is a contradiction.

Item 2. First, we show the query complexity upper bound. Since $\mathcal{Q}_n$ is trivially testable using n queries, Lemma 3.26.1 implies that $Q_{\varepsilon,n}(\mathcal{Q}^{(q)}) = O(q(n))$. By Lemma 3.30, $Q_{\varepsilon/2,n}(\mathcal{C}) = O(n)$, so applying Lemma 3.26.1, $Q_{\varepsilon,n}(\mathcal{C}^{(k)}) = O(k(n))$, which is $O(q(n))$ by Definition 3.33.6. Applying Lemma 3.21.1 to $\mathcal{Q}_n^{(q)}$ and $\mathcal{C}_n^{(k)}$, we obtain that $Q_{\varepsilon,2n}(\mathcal{P}) = O(q(n))$.

Next, we show the time complexity upper bound. By assumption (Definitions 3.33.1 and 3.33.4), $T_{\varepsilon/2,n}(\mathcal{Q}) = O(n \log n)$ and $q(n)$ is computable in time $O(q(n) \log q(n))$, so applying Lemma 3.26.2, $T_{\varepsilon,n}(\mathcal{Q}^{(q)}) = O(q(n) \log q(n) + \log^{1+o(1)} n)$.

By Definition 3.33.3 and Lemma 3.30.1, $T_{\varepsilon/2,n}(\mathcal{C}) = O(n \log n + T_u(\lfloor r \cdot n \rfloor)) = O(T_u(\lfloor r \cdot n \rfloor))$, recalling that $T_u(m) = \Omega(m \log m)$. By Definition 3.33.6, $k_*(n)$ is computable in time $O(T_u(k_*(n)))$, and hence $k(n) = k_*(n)/r$ is computable in time $O(T_u(k_*(n)) + \log^{1+o(1)}(k_*(n))) = O(T_u(r \cdot k(n)))$. Hence Lemma 3.26.2 implies that $T_{\varepsilon,n}(\mathcal{C}^{(k)}) = O(T_u(r \cdot k(n)) + \log^{1+o(1)} n) = O(T_u(k_*(n)) + \log^{1+o(1)} n) = O(F(t(n)) + \log^{1+o(1)} n)$, where the last step is valid by attainment (Definition 3.32).

Applying Lemma 3.21.2 to $\mathcal{Q}_n^{(q)}$ and $\mathcal{C}_n^{(k)}$, we get that $T_{\varepsilon,2n}(\mathcal{P}) = O(q(n) \log q(n) \log n + F(t(n)) \log n + \log^{2+o(1)} n) = O(q(n) \log^2 n + F(t(n)) \log n + \log^{2+o(1)} n)$.

By Remark 3.27, the same tester achieves both the query and time complexity upper bound. □

Remark 3.35. Lemma 3.34.1 also admits the following **infinitely often** version. If we *strengthen* the hypothesis of Definition 3.33 by requiring that the $T_\ell(n)$ lower bound for deciding language L holds even against algorithms that succeed on infinitely many input lengths, and then *weaken* the notion of attainment in Definition 3.32 by only requiring the inequality $T_\ell(k(n)) \geq t(n)$ on infinitely many $n \in \mathbb{N}$, then a straightforward modification of the proof of Lemma 3.34.1 (using Remarks 3.22, 3.28 and 3.31) yields that any tester for $\mathcal{P}_{2n}$ that succeeds in the usual sense (i.e., at a.e. input length) must use $\Omega\left(\frac{t(n)}{\log^{2.01} n}\right)$ time (on infinitely many input lengths).

3.5 Hierarchy theorems

3.5.1 Unconditional weak hierarchy theorem (proof of Theorem 3.3)

In this subsection, we combine the property $\mathcal{Q}$ from Section 3.3 with the hard language L obtained in Section 3.1.1 to satisfy the requirements of Definition 3.33, and then use Lemma 3.34 to establish Theorem 3.3.

We start with the preconditions of Theorem 3.3. Specifically, we are given a constant $\varepsilon \in (0,1)$, and nondecreasing, unbounded functions $q, t : \mathbb{N} \to \mathbb{N}$ which satisfy the following conditions.

1. The function $n \mapsto q(2n)$ is eventually surjective, $q(n)$ is computable in time $O(q(n) \log q(n))$, and $q^{-1}(n)$ is computable in time $O(t \circ q^{-1}(n))$.
2. $t(n)$ is computable in time $O(t(n))$ and $t^{-1}(n)$ is computable in time $O(n)$.
3. $q(n) \leq \min\left\{\frac{r \cdot n}{2}, \frac{t(n)}{\log^{2.01} n}\right\}$, where r is the constant from Theorem 3.12.

We make the following definitions.

1. Let $\mathcal{Q}$ be the property from Definition 3.16.
2. Let $q_* : \mathbb{N} \to \mathbb{N}$ and $t_* : \mathbb{N} \to \mathbb{R}$ be given by $q_*(n) := q(2n)$ and $t_*(n) := t(2n) \log^{2.01} n$.
3. Let $T_\ell, T_u : \mathbb{N} \to \mathbb{R}$ be given by $T_\ell(n) := t_* \circ q_*^{-1}(n)$ and $T_u(n) := 2^{T_\ell(n)^{2.01}}$.
4. Let L be the language from Theorem 3.6 with parameter[16] T_ℓ.
5. Let $k_* : \mathbb{N} \to \mathbb{N}$ be given by $k_*(n) := q_*(n)$.
6. Let $F : \mathbb{N} \to \mathbb{R}$ be given by $F(n) := 2^{n^{2.01}}$.

Lemma 3.36. *The choices above of $\mathcal{Q}, T_\ell, T_u, L, q_*, t_*, k_*, F$ satisfy the conditions of Definition 3.33.*

Proof. **Conditions on $\mathcal{Q}$.** The first condition, that $\mathcal{Q}$ be a.e. nonempty, holds by the construction in Definition 3.16. Definition 3.33 also requires ε-testing property $\mathcal{Q}_n$ to have query complexity $\Omega(\frac{n}{\log n})$ and time complexity $O(n \log n)$, which is guaranteed by Lemmas 3.17 and 3.18, respectively. Finally, $\mathcal{Q}$ must have an $O(n \log n)$-time YES query provider. This is easy to ensure for any reasonable scheme for encoding 3CNF formulas into binary strings (which we leave implicit for brevity); for example, we can use an empty formula with a trivial all-0s assignment, and the encoder from Theorem 3.12.

Conditions on T_ℓ, T_u, L. The first condition is for $T_\ell(n)$ and $T_u(n)$ to be nondecreasing and $\Omega(n \log n)$, with $T_\ell \leq T_u$. By assumption, $t(n) \geq q(n) \log^2 n$ and $q(n) \leq n$. This yields

$$t_* \circ q_*^{-1}(n) \geq t(2 \cdot q^{-1}(n)/2) \geq q(q^{-1}(n)) \log^2 n \geq n \log^2 n\,,$$

giving us that $T_\ell(n) = \Omega(n \log n)$. Definition 3.33 also requires L to be a.e. nonempty; decidable by an $O(T_u(n))$-time randomized algorithm but not by any $O(T_\ell(n))$-time randomized algorithm, which are all true by Theorem 3.6.

Conditions on q_*, t_*. The conditions that q_*, t_* be nondecreasing, that $q_*(n)$ be eventually surjective and satisfy $q_*(n) \leq n$ and $q_*(n) \log q_*(n) = O(t_*(n)/\log^{1.01} n)$, and that $q_*(n)$ be computable in $O(q_*(n) \log q_*(n))$ time all hold by the assumptions on q, t and the construction of q_*, t_*.

Attainment. Definition 3.33 requires the pair (T_ℓ, T_u) to attain t_* up to gap F with length function k_*. By construction, $T_\ell(k_*(n)) = t_* \circ q_*^{-1}(q_*(n)) = t_*(n)$ for infinitely many $n \in \mathbb{N}$. Since

[16] Formally, we invoke Theorem 3.6, but with a slightly better exponent in the upper bound (say, 2.009) and a function $t'_*(n) \geq t_*(n)$ obtained by approximating $\log^{2.01} n$ by an integer (a constant-factor approximation suffices). By assumption, computing $q^{-1}(n)$ and hence $q_*^{-1}(n)$ takes time $O(t \circ q^{-1}(n)) = O(t_* \circ q_*^{-1}(n))$, and subsequently computing $t'_* \circ q_*^{-1}(n)$ takes time $O(t'_* \circ q_*^{-1}(n))$, meaning T_ℓ is efficiently constructible and the call to Theorem 3.6 is legal.

the construction from Theorem 3.6 rules out **infinitely often** decidability, then by Remark 3.35, relaxing the definition of attainment to infinitely many input lengths is still valid.

For the upper bound, we have

$$T_u(k_*(n)) = 2^{T_\ell(k_*(n))^{2.01}} = F(T_\ell(k_*(n))) = F(t_* \circ q_*^{-1} \circ q_*(n)) \le F(t_*(n)).$$

Conditions on k_*. The conditions that $k_*(n) = O(q_*(n))$ and $k_*(n) \le r \cdot n$ follow from construction and the assumptions on q. For computability of $k_*(n)$, we use the assumption on the computability of $q(n)$ and the lower bound on $T_u(n)$ to get that $k_*(n)$ is computable in time $O(k_*(n) \log k_*(n)) = O(T_u(k_*(n)))$. Similarly, for the computability of $k_*^\dagger(n)$, we use the assumption on the computability of $q^{-1}(n)$ and the lower bound on $T_\ell(n)$ to get that $k_*^\dagger(n)$ is computable in time $O(t \circ q^{-1}(n) + \log^{1+o(1)} q^{-1}(n)) = O(t_* \circ q_*^{-1}(n)) = O(T_\ell(n))$. □

Combining Lemmas 3.34 and 3.36 yields the following corollary, which implies Theorem 3.3.

Corollary 3.37. *Let $\mathcal{P}$ be the hard property obtained by applying Definition 3.33 with the choices above. Then ε-testing $\mathcal{P}_{2n}$ requires $\Omega\left(\frac{q_*(n)}{\log q_*(n)}\right) = \Omega\left(\frac{q(2n)}{\log q(2n)}\right)$ queries and $\Omega\left(\frac{t_*(n)}{\log^{2.01} n}\right) = \Omega\left(t(2n)\right)$ time. Moreover, there exists an ε-tester for $\mathcal{P}_{2n}$ using $O(q_*(n)) = O(q(2n))$ queries and $O(q_*(n) \log^2 n + F(t_*(n)) \log n + \log^{2+o(1)} n) = O(2^{t(2n)^{2.01} \log^{4.05}(2n)})$ time.*

3.5.2 Strong hierarchy theorem from SETH (proof of Theorem 3.4)

In this subsection, we combine the property $\mathcal{Q}$ from Section 3.3 with the hard language L obtained from SETH in Section 3.1.2 to satisfy the requirements of Definition 3.33, and then use Lemma 3.34 to establish Theorem 3.4. Toward this goal, assume the hypotheses of Theorem 3.4. Specifically, let $\varepsilon, \gamma > 0$ be sufficiently small constants and $q, t : \mathbb{N} \to \mathbb{N}$ be nondecreasing unbounded functions satisfying the following conditions.

1. The function $n \mapsto q(2n)$ is eventually surjective and $q(n)$ is computable in time $O(q(n) \log q(n))$.
2. $t(n)$ is computable in time $O(t(n))$, and $t^{-1}(n)$ is computable in time $O(n)$.
3. $q(n) \le \min\left\{\frac{r \cdot n}{2}, \frac{t(n)}{\log^{2.01} n}\right\}$, where r is the constant from Theorem 3.12.
4. $\log t(n) \log\log t(n)$ has slope $o(1)$ and $t(n) \le 2^{O\left(\frac{q(n)}{\log q(n)}\right)}$.

We then make the following definitions.

1. Let $\mathcal{Q}$ be the property from Definition 3.16.
2. Let L be the language $L^{(\gamma)}$ from Definition 3.10.
3. Let $T_\ell, T_u : \mathbb{N} \to \mathbb{R}$ be given by $T_\ell(n) := 2^{\frac{(1-\gamma)n}{A_\gamma \log n}}$ and $T_u(n) := 2^{\frac{n}{A_\gamma \log n}}$, where $A_\gamma > 0$ is the constant from Lemma 3.11.
4. Let $q_*, t_* : \mathbb{N} \to \mathbb{N}$ be given by $q_*(n) := q(2n)$ and $t_*(n) := t(2n)$.
5. Define the auxiliary function $f : \mathbb{N} \to \mathbb{R}$ by $f(m) := \frac{A_\gamma}{1-3\gamma} \log m \log\log m$. Then $k_* : \mathbb{N} \to \mathbb{N}$ is an explicit computable function, specified in Lemma 3.38 below, and $|k_*(n) - \lfloor f(t_*(n)) \rfloor| \le 1$.
6. Let $F : \mathbb{N} \to \mathbb{R}$ be given by $F(n) := n^{\frac{1}{1-4\gamma}}$.

Lemma 3.38. *The choices above of $\mathcal{Q}, L, T_\ell, T_u, k_*, q_*, t_*, F$ satisfy the conditions of Definition 3.33.*

Proof. **Conditions on $\mathcal{Q}$** have already been established in Lemma 3.36.

Conditions on L. The first condition is for $T_\ell(n)$ and $T_u(n)$ to be nondecreasing and $\Omega(n \log n)$, with $T_\ell \le T_u$; this is true by construction. Definition 3.33 also requires the language L to be a.e. nonempty, which we satisfy by choosing an appropriate encoding scheme for formulas into binary strings in Definition 3.10. Finally, L must be decidable by an $O(T_u(n))$-time randomized algorithm but not by any $O(T_\ell(n))$-time randomized algorithm, which is true by Lemma 3.11.

Conditions on q_*, t_*. The conditions that q_*, t_* be nondecreasing, that q_* be eventually surjective and satisfy $q_*(n) \le n$ and $q_*(n) \log q_*(n) = O(t_*(n)/\log^{1.01} n)$, and that $q_*(n)$ be computable in $O(q_*(n) \log q_*(n))$ time hold by the assumptions on q, t and the definition of q_*, t_*.

Attainment. Definition 3.33 requires that the pair (T_ℓ, T_u) attain t_* up to gap F with length function k_*. We first show that $T_\ell(k_*(n)) \ge t_*(n)$ for all sufficiently large n. Since t_* is unbounded, for all sufficiently large n, we have

$$k_*(n) \ge \left\lfloor \frac{A_\gamma}{1-3\gamma} \log t_*(n) \log\log t_*(n) \right\rfloor - 1 \ge \frac{(1+0.1\gamma)A_\gamma}{1-\gamma} \log t_*(n) \log\log t_*(n)$$

and hence, as desired,

$$T_\ell(k_*(n)) = 2^{\frac{(1-\gamma)k_*(n)}{A_\gamma \log k_*(n)}} \ge 2^{\frac{(1+0.1\gamma)\log t_*(n)\log\log t_*(n)}{\log\left(\frac{A_\gamma}{1-3\gamma}\cdot \log t_*(n)\log\log t_*(n)\right)}} \ge 2^{\frac{(1+0.1\gamma)\log t_*(n)\log\log t_*(n)}{\log((\log t_*(n))^{1+0.1\gamma})}} = 2^{\log t_*(n)} = t_*(n)\,.$$

We now show that $T_u(k_*(n)) \le F(t_*(n))$ for all sufficiently large n. As above, for all sufficiently large n, we have

$$k_*(n) \le \left\lfloor \frac{A_\gamma}{1-3\gamma} \log t_*(n) \log\log t_*(n) \right\rfloor + 1 \le \frac{A_\gamma}{1-4\gamma} \log t_*(n) \log\log t_*(n)$$

and hence, as desired,

$$T_u(k_*(n)) = 2^{\frac{k_*(n)}{A_\gamma \log k_*(n)}} \le 2^{\frac{\log t_*(n)\log\log t_*(n)}{(1-4\gamma)\log\left(\frac{A_\gamma}{1-3\gamma}\log t_*(n)\log\log t_*(n)\right)}} \le \left(2^{\frac{\log t_*(n)\log\log t_*(n)}{\log\log t_*(n)}}\right)^{\frac{1}{1-4\gamma}} = F(t_*(n))\,.$$

Conditions on k_*. We now complete the definition of k_* and show that it is eventually surjective with $k_*(n) = O(q_*(n))$ and $k_*(n) \le r \cdot n$, and moreover that $k_*(n)$ is computable in time $O(T_u(k_*(n)))$ and ${k_*}^\dagger(n)$ is computable in time $O(T_\ell(n))$.

We use the following fact from numerical analysis, see e.g., [Bre76]: given a number x with ℓ bits of precision, it is possible to compute $\log x$ up to ℓ bits of precision in time $O(\ell \operatorname{polylog} \ell)$. It follows that the function $f(m) = \frac{A_\gamma}{1-3\gamma} \log m \log\log m$ may be computed up to arbitrarily small constant additive error in time $O(\log m \operatorname{poly}\log\log m)$.

We first complete the definition of $k_*(n)$ by specifying the algorithm computing it in time $O(T_u(k_*(n)))$. Note that the desired upper bound is

$$O(T_u(k_*(n))) \ge O(T_\ell(k_*(n))) \ge O(t_*(n))\,,$$

where the last step is valid by attainment. Recall that $k_*(n)$ must satisfy $|k_*(n) - \lfloor f(t_*(n))\rfloor| \le 1$. Now, $t_*(n)$ is computable in time $O(t_*(n))$ by the assumption on t, and given $t_*(n)$, using the

observation above we may estimate $f(t_*(n))$ up to additive error at most $\frac{1}{8}$ – call this estimate $\widetilde{f}$ – in time $O(\log t_*(n) \operatorname{poly}\log\log t_*(n)) = O(t_*(n))$. We then define $k'(n) := \left\lfloor \widetilde{f} \right\rfloor$, and observe that it satisfies $|k'(n) - f(t_*(n))| \le 1$. Thus the computability claim holds.

Second, we claim that k_* is eventually surjective with $k'(n) = O(q_*(n))$ and $k_*(n) \le r \cdot n$. The first claim holds because the function $n \mapsto f(t_*(n))$ has slope $o(1)$ by the hypothesis that $\log t(n) \log\log t(n)$ has slope $o(1)$, along with the definition of $k_*(n)$ above. Since $t(n) \le 2^{O\left(\frac{q(n)}{\log q(n)}\right)}$, which implies that $t_*(n) \le 2^{O\left(\frac{q_*(n)}{\log q_*(n)}\right)}$, we get $k_*(n) = O\left(\log t_*(n) \log\log t_*(n)\right) = O(q_*(n))$, as desired. Finally, since $\log t_*(n) \log\log t_*(n)$ has slope $o(1)$, we get $k_*(n) = O\left(\log t_*(n) \log\log t_*(n)\right) = o(n)$, so $k_*(n) \le r \cdot n$ for all sufficiently large n.

Finally, we claim that ${k_*}^\dagger(n)$ is computable in $O(T_\ell(n))$ time. That is, there is an algorithm which, on input n, outputs some $n' \in \mathbb{N}$ satisfying $k_*(n') = n$ in the announced time. The desired upper bound is

$$O\left(T_\ell(n)\right) = O\left(T_\ell(k_*(n'))\right) \ge O(t_*(n')),$$

where the last step is valid by attainment. The algorithm works as follows: on input n, it first performs exponential search to find an integer m satisfying $f(m) \in \left[n + \frac{1}{4}, n + \frac{3}{4}\right]$, which exists because $s \mapsto f(t_*(s))$ has slope $o(1)$. Specifically, there exists $n' \in \mathbb{N}$ satisfying $\left|f(t_*(n')) - \left(n + \frac{1}{2}\right)\right| \le \frac{1}{16}$, so by estimating $f(m_i)$ up to additive error $\frac{1}{16}$ in time $O(\log m_i \operatorname{poly}\log\log m_i)$ in each step i of the exponential search and stopping when this estimate is within distance $\frac{1}{8}$ of $n + \frac{1}{2}$, we can find integer m satisfying $\left|f(m) - \left(n + \frac{1}{2}\right)\right| \le \frac{1}{4}$, as desired. Since m satisfies $m \le O(2^n)$ and $n \le O(\log^2 m)$, the exponential search takes $O(n)$ steps, and hence m can be found in time $O(\operatorname{polylog} m)$.

Then, the algorithm uses the invertibility of t to compute in $O(m)$ time the number n' satisfying $t_*(n'-1) \le m \le t_*(n')$, and outputs n'. We claim that $k_*(n') = n$. Note that $f(t_*(n'-1)) \le f(m) \le n + \frac{3}{4}$, which implies that $f(t_*(n')) \le n + \frac{3}{4} + \frac{1}{16} = n + \frac{13}{16}$ since $s \mapsto f(t_*(s))$ has slope $o(1)$. It follows that $k_*(n') = \left\lfloor \widetilde{f} \right\rfloor \le \left\lfloor n + \frac{13}{16} + \frac{1}{8} \right\rfloor = n$. Similarly, $f(t_*(n')) \ge f(m) \ge n + \frac{1}{4}$, and hence $k_*(n') = \left\lfloor \widetilde{f} \right\rfloor \ge \left\lfloor n + \frac{1}{4} - \frac{1}{8} \right\rfloor = n$. Thus $k_*(n') = n$, as claimed. The total running time is $O(\operatorname{polylog} m) + O(m) = O(m) \le O(t_*(n'))$, as needed. □

Combining Lemmas 3.34 and 3.38 yields the following corollary, which implies Theorem 3.4.

Corollary 3.39. *Let $\mathcal{P}$ be the hard property obtained by applying Definition 3.33 with the choices above. Then under Assumption 3.8, ε-testing $\mathcal{P}_{2n}$ requires $\Omega\left(\frac{q_*(n)}{\log q_*(n)}\right) = \Omega\left(\frac{q(2n)}{\log q(2n)}\right)$ queries and $\Omega\left(\frac{t_*(n)}{\log^{2.01} n}\right) = \Omega\left(\frac{t(2n)}{\log^{2.01}(2n)}\right)$ time. Moreover, there exists an ε-tester for $\mathcal{P}_{2n}$ using $O(q_*(n)) = O(q(2n))$ queries and $O\left(t_*(n)^{\frac{1}{1-4\gamma}} \log n\right) = O\left(t(2n)^{\frac{1}{1-4\gamma}} \log(2n)\right)$ time.*

4 Distribution-free distance approximation for halfspaces

In this section, we prove a fine-grained hardness result for distribution-free distance approximation to halfspaces in the low-dimensional setting. We first formally define distribution-free distance approximation and the halfspace property over $\mathbb{Z}^d$.

Definition 4.1 (Distribution-free distance approximation)**.** *Let $\mathcal{X}$ be a universe set and $\mathcal{P} \subset \{0,1\}^{\mathcal{X}}$ be a property of Boolean functions over $\mathcal{X}$. Define the* distance from a function $f : \mathcal{X} \to$

$\{0,1\}$ to property $\mathcal{P}$ with respect to a distribution $\mathcal{D}$ over $\mathcal{X}$ *as*

$$\mathsf{dist}_{\mathcal{D}}(f,\mathcal{P}) := \inf_{g\in\mathcal{P}} \mathsf{dist}_{\mathcal{D}}(f,g)\,, \quad \textit{where} \quad \mathsf{dist}_{\mathcal{D}}(f,g) := \mathop{\mathbb{P}}_{x\sim\mathcal{D}}[f(x)\neq g(x)]\,.$$

A randomized algorithm $\mathcal{A}$ is a distribution-free distance approximation *algorithm for property $\mathcal{P}$ if, for each probability distribution $\mathcal{D}$ over $\mathcal{X}$ and function $f:\mathcal{X}\to\{0,1\}$, given input parameters $\varepsilon,\delta\in(0,1)$, algorithm $\mathcal{A}$ uses* labeled samples *of the form $(x\sim\mathcal{D}, f(x))$ and queries to f to output a number $\widehat{\alpha}$ which, with probability at least $1-\delta$, satisfies $|\mathsf{dist}_{\mathcal{D}}(f,\mathcal{P})-\widehat{\alpha}|\le\varepsilon$. By default, $\delta=\frac{1}{3}$.*

Definition 4.2 (Hyperplanes and halfspaces)**.** *A $(d-1)$-dimensional* hyperplane *H in $\mathbb{R}^d$ is the set of points $x\in\mathbb{R}^d$ satisfying $\langle w,x\rangle+\theta=0$, where $w\in\mathbb{R}^d$ and $\theta\in\mathbb{R}$ are the parameters defining H. A function $f:\mathbb{Z}^d\to\{0,1\}$ is a* halfspace *if there exists a hyperplane H with parameters w,θ such that $f(x)=\mathbb{1}[\langle w,x\rangle+\theta\ge 0]$ for all $x\in\mathbb{Z}^d$. We denote the class of halfspaces over $\mathbb{Z}^d$ by $\mathcal{H}$.*

Our result is the following.

Theorem 4.3. *Under the integer k-SUM conjecture, for all constants $d\in\mathbb{N}$ and $\gamma>0$, there is no distribution-free distance approximation algorithm for halfspaces over $\mathbb{Z}^d$ running in time $(1/\varepsilon)^{\lceil(d+1)/2\rceil-\gamma}$ (as a function of ε). This lower bounds holds[17] even if the input distribution $\mathcal{D}$ is promised to be supported on points with absolute coordinate values at most $(1/\varepsilon)^{O_d(1)}$.*

We use the same RAM model as in Section 3, with modifications to match the setting of Theorem 4.3. The input tape is intially empty. The dimension $d\in\mathbb{N}$ is part of the problem description, and the error parameter ε is given on the parameter tape, encoded in $O(\log(1/\varepsilon))$ bits.[18] To model access to a distribution $\mathcal{D}$ and a function $f:\mathbb{Z}^d\to\{0,1\}$, we add two unit time operations:

DrawSamp: Clears the input tape and resets its head, then samples $(x,f(x))$ with $x\sim\mathcal{D}$, and writes $x_1,\ldots,x_d,f(x)$ on the input tape (recall that $x\in\mathbb{Z}^d$ and $f(x)$ is a bit).

QueryFunc(X_j): Clears the input tape and resets its head, sets $x:=(X_j,\ldots,X_{j+d-1})$ using d machine registers (see Table 1), and writes $f(x)$ on the input tape.

We prove Theorem 4.3 via a reduction from the integer k-SUM problem, stated next.

Problem 4.4 (k-SUM)**.** *Given k lists $A_1,\ldots,A_k$ of n distinct integers each, where each integer lies in the range $[-n^{2k},n^{2k}]$, is there set of choices $a_i\in A_i$ for each $i\in[k]$ such that $a_1+\cdots+a_{k-1}=a_k$?*

Our hardness result is based on the following standard conjecture on the time complexity of the k-SUM problem (see the lecture notes [WW20] for further background on this conjecture).

Conjecture 4.5 (k-SUM conjecture [AL13])**.** *For each constant $k\ge 2$ and $\gamma>0$, no randomized algorithm can solve Problem 4.4 in $O(n^{\lceil k/2\rceil-\gamma})$ time with error probability at most $\frac{1}{3}$.*

Remark 4.6. To align with our RAM model in Section 3 and avoid complications with real-number representation, we study distance approximation over $\mathbb{Z}^d$ rather than $\mathbb{R}^d$. Our arguments extend

[17] The significance of stating the lower bound for distributions supported on points with bounded coordinates is that, in our RAM model with logarithmic cost, one could obtain trivial lower bounds by constructing inputs that are arbitrarily expensive to read. The stated bound implies only a polylogarithmic overhead for reading the input, which keeps the bound meaningful.

[18] If $\langle\varepsilon\rangle$ is too long, we can reduce ε by at most a factor of 2 to get the desired length without affecting asymptotics.

to the real setting under a suitable real RAM model and the k-SUM conjecture; see Remark 4.8.

4.1 Proof of Theorem 4.3

Distance approximation for halfspaces is closely related to geometric decision problems known to be k-SUM hard. For example, the GEOMBASE problem[19] is a classic 3-SUM hard problem [GO95], and deciding whether $d+1$ points in $\mathbb{R}^d$ lie on a hyperplane is $(d+1)$-SUM hard [Eri96]. We build upon this theme by reducing from $(d+1)$-SUM to distance approximation for halfspaces.

The starting point for our proof is a construction from [FHKS17] for the following high-dimensional version of the GEOMBASE problem. A $(d-1)$-dimensional hyperplane in $\mathbb{R}^d$ is *vertical* if it contains a vertical line, i.e., two distinct points $p, q \in \mathbb{R}^d$ satisfying $p_i = q_i$ for all $i \in [d-1]$; otherwise, it is *non-vertical*. The problem is: given n points in $\mathbb{R}^d$, do any $d+1$ of them lie on a non-vertical $(d-1)$-dimensional hyperplane? This problem is $(d+1)$-SUM hard and was used by [FHKS17] to prove lower bounds for linear separability of probabilistic point sets. We state the $(d+1)$-SUM hardness result in the real-valued setting, which is the norm in computational geometry.

Theorem 4.7 (Implicit in [FHKS17, Theorem 5.1])**.** *Consider determining whether any $d+1$ of the given n points in $\mathbb{R}^d$ lie on a non-vertical $(d-1)$-dimensional hyperplane. Under the (real-valued) k-SUM conjecture, no randomized algorithm solves this problem in time $n^{\lceil (d+1)/2 \rceil - \gamma}$ for any $\gamma > 0$.*

Their reduction is as follows. Let $P = (p^{(1)}, \ldots, p^{(d)})$ be the vertices of a convex polygon embedded in the hyperplane $\{x_d = 0\} \subset \mathbb{R}^d$. Let $c = \frac{1}{d}\sum_{i \in [d]} p^{(i)}$ be the center of mass of the vertices. For $j \in [d]$, let e_j denote the the j^{th} standard basis vector. Given a $(d+1)$-SUM instance $A_1, \ldots, A_{d+1}$, map each value $a_i \in A_i$ for $i \in [d]$ to the point $a_i^* := p^{(i)} + a_i \cdot e_d \in \mathbb{R}^d$. Then map each value $a_{d+1} \in A_{d+1}$ of the last input set to the point $a_{d+1}^* := c + \frac{a_{d+1}}{d} \cdot e_d \in \mathbb{R}^d$. Let S be the set of $(d+1)n$ points given by this reduction. It follows that the $(d+1)$-SUM instance is a YES instance iff there exist $d+1$ points in S that lie on a non-vertical $(d-1)$-dimensional hyperplane.

To prove our distribution approximation lower bound, we modify the above construction as follows:

1. Multiply all points by d to ensure integer coordinates.
2. Replace each point a_i^* with two nearby points: one slightly above, labeled 1 by f; one slightly below, labeled 0.
3. Let $\mathcal{D}$ be the uniform distribution over these points; we provide access to labeled samples from $\mathcal{D}$ and f and query access to f.

We will show that if we start with a YES instance of $(d+1)$-SUM, then some halfspace correctly labels $(d+1)(n+1)$ points. If we start with a NO instance, then every halfspace labels at most $\frac{2(d+1)n}{2} + d = (d+1)(n+1) - 1$ points correctly. Thus, approximating $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{H})$ to additive error $O_d(1/n)$ solves the $(d+1)$-SUM problem. Setting $\varepsilon = \Theta(\frac{1}{n})$ yields Theorem 4.3.

We now formalize this argument.

Proof of Theorem 4.3. Given a $(d+1)$-SUM instance with input lists $A_1, \ldots, A_{d+1}$, each of length n, we reduce to distribution-free distance approximation for halfspaces over $\mathbb{Z}^d$ with parameter $\varepsilon = \Theta(\frac{1}{n})$ using the following construction.

[19] Given n points $(x_1, y_1), \ldots, (x_n, y_n)$ on the plane, with $y_i \in \{0, 1, 2\}$ for all $i \in [n]$, do any three of these points lie on a non-horizontal line?

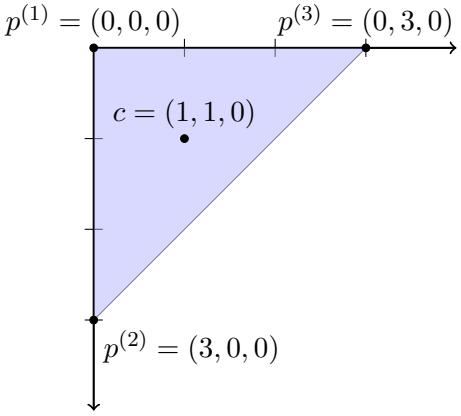

Figure 1: The polygon in the reduction for $d = 3$.

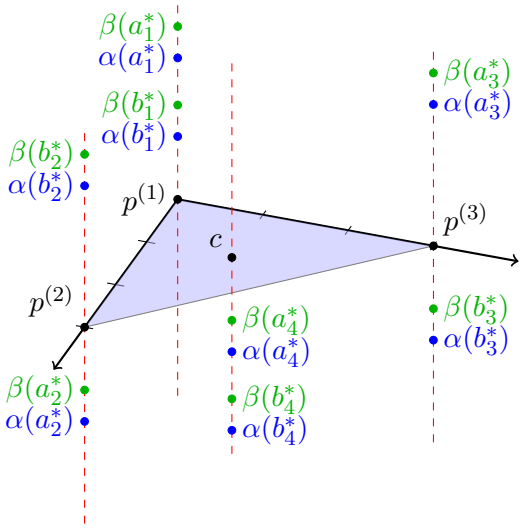

Figure 2: An illustration of our reduction for $d = 3$ and $n = 2$.

Let $P = (p^{(1)}, \ldots, p^{(d)})$ be the polygon in the hyperplane $\{x_d = 0\} \subset \mathbb{R}^d$, where $p^{(1)}$ is the zero vector and $p^{(i)} := d \cdot e_{i-1}$ for all $i = 2, \ldots, d$. Let c be the center of mass of the vertices of P, i.e., $c = \frac{1}{d}\sum_{i=1}^{d} p^{(i)} = (1, \ldots, 1, 0)$. Initialize a point set $S = \emptyset$ and a function $f : \mathbb{Z}^d \to \{0, 1\}$ to be zero on the entire domain. For each $i \in [d]$ and $a_i \in A_i$, define $a_i^* := p^{(i)} + 4da_i \cdot e_d$; for the last set A_{d+1}, for each $a_{d+1} \in A_{d+1}$, define $a_{d+1}^* := c + 4a_{d+1} \cdot e_d$. For each $i \in [d+1]$ and $a_i \in A_i$, add two points to the set S:

$$\alpha(a_i^*) := a_i^* - e_d, \quad \text{and} \quad \beta(a_i^*) := a_i^* + e_d,$$

and set the labels of these points to $f(\alpha(a_i^*)) := 0$ and $f(\beta(a_i^*)) := 1$. This completes the construction of the set S of $2(d+1)n$ points and of the Boolean function $f : \mathbb{Z}^d \to \{0, 1\}$.

To finish the reduction, let $\mathcal{D}$ be the uniform distribution over S and set $\varepsilon = \frac{1}{5(d+1)n}$. If the distance approximation algorithm executed on the constructed instance returns an estimate at most $\frac{|S|-(n+1)(d+1)+1/2}{|S|}$, accept; otherwise, reject.

It remains to prove the correctness of the reduction and analyze the efficiency of the simulation.

Correctness. For each list A_i in the k-SUM instance, all points created from the numbers in A_i using α and β functions are on the same line (see Figure 2); specifically, they only differ in the coordinate d. For all $i \in [d+1]$ and all $a_i, b_i \in A_i$ such that $a_i < b_i$, we have $a_i \le b_i - 1$, since a_i and b_i are integers. Consequently, the points a_i^* and b_i^* differ by at least 4 in coordinate d, which implies $\alpha(a_i^*) \prec \beta(a_i^*) \prec \alpha(b_i^*) \prec \beta(b_i^*)$.

Now suppose that $A_1, \ldots, A_{d+1}$ is a YES instance of the $(d+1)$-SUM problem. Then there exist $a_1 \in A_1, \ldots, a_{d+1} \in A_{d+1}$ such that $\sum_{i=1}^{d} a_i = a_{d+1}$, which implies $\frac{1}{d}\sum_{i=1}^{d} a_i^* = a_{d+1}^*$. Hence, the unique hyperplane H passing through points $a_1^*, \ldots, a_d^*$ also passes through point a_{d+1}^*. Therefore, there exists a halfspace h which agrees with f on at least all of the points $\alpha(a_i^*), \beta(a_i^*)$ for $i \in [d+1]$, plus at least half of the remaining $\alpha(\cdot), \beta(\cdot)$ points in S. Hence, f agrees with h on at least $\frac{2(d+1)n}{2} + (d+1) = (d+1)(n+1)$ points of S, and thus $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{H}) \le \frac{|S|-(n+1)(d+1)}{|S|}$.

Next, suppose that $A_1, \ldots, A_{d+1}$ is a NO instance of the $(d+1)$-SUM problem. We claim that

$$\mathsf{dist}_{\mathcal{D}}(f, \mathcal{H}) \geq \frac{|S| - (n+1)(d+1) + 1}{|S|}\,.$$

The claim follows if we show that, for every halfspace h, the function f agrees with h on at most $\frac{2(d+1)n}{2} + d = (d+1)(n+1) - 1$ points of S. Suppose for contradiction that f and h agree on at least $(d+1)(n+1)$ points of S. Then for each $i \in [d+1]$, there must be some $a_i \in A_i$ such that h correctly labels both $\alpha(a_i^*)$ and $\beta(a_i^*)$. This implies that the separating hyperplane H passes between these two points for each $i \in [d+1]$.

Since $A_1, \ldots, A_{d+1}$ are sets of integers and form a NO instance of the $(d+1)$-SUM problem,

$$\left|\sum_{i=1}^{d} a_i - a_{d+1}\right| \geq 1\,, \quad \text{and thus} \quad \left|\frac{1}{d}\sum_{i=1}^{d} 4da_i - 4a_{d+1}\right| \geq 4\,.$$

Therefore, for every choice of $\lambda_i \in [-1, 1]$ for each $i \in [d+1]$, the triangle inequality implies that

$$\left|\frac{1}{d}\sum_{i=1}^{d}(4da_i + \lambda_i) - (4a_{d+1} + \lambda_{d+1})\right| \geq 2 > 0\,.$$

Consequently, no hyperplane can pass within vertical distance 1 of every point a_i^* for all $i \in [d+1]$, which contradicts the previous conclusion that H passes between points $\alpha(a_i^*)$ and $\beta(a_i^*)$ for each $i \in [d+1]$, completing the proof of the claim.

We conclude that, if the distance approximation algorithm is run with parameter $\varepsilon = \frac{1}{5(d+1)n} < \frac{1}{2|S|}$, the reduction algorithm correctly answers for both YES and NO instance with probability at least $\frac{2}{3}$.

Efficiency. To simulate a distance approximation algorithm $\mathcal{A}$, we first read the entire input $A_1, \ldots A_{d+1}$ and sort each list A_i in increasing order. This takes $O(n\,\mathrm{polylog}(n))$ time. To answer a query $f(x)$ from $\mathcal{A}$, we spend $O(\mathrm{polylog}(n))$ time to perform binary search and check whether $x = \alpha(a_i^*)$ or $x = \beta(a_i^*)$ for some $a_i \in A_i$; if so, we return the corresponding label, and otherwise return 0. When $\mathcal{A}$ requests a labeled sample $(x \sim \mathcal{D}, f(x))$, we use random bits to take a sample from the point set S and compute its label as discussed. This also takes $O(\mathrm{polylog}(n))$ time.

Suppose $\mathcal{A}$ runs in time $O\left((1/\varepsilon)^{\lceil (d+1)/2 \rceil - \gamma}\right)$. Then we can solve our $(d+1)$-SUM instance in time $O\left((1/\varepsilon)^{\lceil (d+1)/2 \rceil - \gamma}\,\mathrm{polylog}(1/\varepsilon)\right)$, a contradiction. This completes the proof. □

Remark 4.8 (On the use of integrality)**.** Aside from our choice of RAM model, the proof of Theorem 4.3 also relies on the integrality of the input to ensure that a NO instance of $(d+1)$-SUM maps to a point set that cannot be well-approximated by any halfspace. Specifically, any hyperplane that does not pass through the exact point a_i^* cannot lie sufficiently close it to still separate $\alpha(a_i^*)$ and $\beta(a_i^*)$, preventing "almost-satisfiable" instances.

If we replicated our construction in the real-valued setting, with an appropriately adapted real RAM model, the only caveat is that, under arbitrary real-valued inputs, we no longer obtain this robust separation property for free; although we could place the points $\alpha(a_i^*)$ and $\beta(a_i^*)$ arbitrarily close to a_i^*, it seems difficult to choose this distance a priori without solving a k-SUM-like problem.

However, this issue disappears if we assume that the real-valued k-SUM problem, in the real RAM model, is still difficult even if the input is promised to be integer. Indeed, not only does this

seem natural enough (the real RAM model is not "supposed" to distinguish between integers and non-integers), but the original work of [GO95], who systematized the notion of 3SUM-hardness, made precisely the assumption that 3SUM is difficult on integer inputs in the real RAM model. We therefore view the real-valued analogue of Theorem 4.3 as equally plausible.

5 Statistical query lower bounds for the Gaussian distribution

In this section, we present *evidence* for a time complexity lower bound for distance approximation of halfspaces under the standard Gaussian distribution over $\mathbb{R}^d$, in the low-dimensional regime where d is a constant, in the form of a sample complexity lower bound against Statistical Query (SQ) algorithms.

For a fixed $d \in \mathbb{N}$, let $\mathcal{H}$ denote the class of halfspaces over $\mathbb{R}^d$, each of which is an $\mathbb{R}^d \to \{\pm 1\}$ function, and $\mathcal{N}(0, I)$ denote the standard multivariate Gaussian distribution over $\mathbb{R}^d$.

Definition 5.1 (Distribution-specific distance approximation in the SQ model)**.** *Let $d \in \mathbb{N}$. A randomized SQ algorithm $\mathcal{A}$ is a* distance approximation *algorithm for halfspaces under $\mathcal{N}(0, I)$ if, given parameters $\varepsilon, \delta \in (0, 1)$ and access to input function $f : \mathcal{X} \to \{\pm 1\}$ via a* STAT *oracle (see Definition 5.3), algorithm $\mathcal{A}$ outputs a number $\widehat{\alpha}$ which, with probability at least $1 - \delta$, satisfies $|\mathsf{dist}_{\mathcal{N}(0,I)}(f, \mathcal{H}) - \widehat{\alpha}| \leq \varepsilon$.*

The main result of this section is the following lower bound, which qualitatively matches the fine-grained, distribution-free lower bound in Theorem 4.3.

Theorem 5.2. *Let $d \geq 2$ be a constant in $\mathbb{N}$. Then every randomized SQ distance approximation algorithm for halfspaces under $\mathcal{N}(0, I)$ with success probability at least 0.51 requires*[20] *$(1/\varepsilon)^{\Omega(d)}$ queries to* STAT$(\varepsilon^{\Omega(d)})$.

We prove Theorem 5.2 by extending an argument of [DKZ20], who showed an SQ lower bound of $d^{\mathrm{poly}(1/\varepsilon)}$ for agnostic learning of halfspaces under the Gaussian distribution in the high-dimensional regime where $d \gg \mathrm{poly}(1/\varepsilon)$. Our argument combines a packing number result for the unit sphere in the low-dimensional regime, the construction of a "pseudorandom" function that serves as the NO instance for the distance approximation task, and a nonuniform derandomization argument.

The rest of this section is organized as follows. Section 5.1 defines the SQ model and introduces the key notion of SQ dimension. Section 5.2 extracts the result we use from [DKZ20]. Section 5.3 establishes the packing result. Section 5.4 constructs the pseudorandom function for SQ algorithms. Section 5.5 combines these ingredients to give a set of functions with high SQ dimension, which implies a lower bound against deterministic SQ algorithms. Finally, Section 5.6 proves Theorem 5.2.

5.1 SQ algorithms and SQ dimension

We start by formally defining the SQ model.

Definition 5.3 (STAT oracle, SQ algorithm [Kea98])**.** *Let $\mathcal{X}$ be a domain, $f : \mathcal{X} \to [-1, 1]$ be a function, $\mathcal{D}$ be a probability distribution over $\mathcal{X}$, and $\tau > 0$ be a* tolerance parameter*. Given a* statistical

[20] By $\Omega(d)$, we mean cd, where c is an absolute constant.

query $q : \mathcal{X} \times [-1,1] \to [-1,1]$, *the oracle* $\textsc{Stat}(\tau)$ *outputs* $v \in \mathbb{R}$ *satisfying* $\left| \mathop{\mathbb{E}}_{x \sim \mathcal{D}} [q(x, f(x))] - v \right| \leq \tau$. *A* Statistical Query (SQ) algorithm *accesses its input* f *only via calls to a* $\textsc{Stat}$ *oracle.*

Given a domain $\mathcal{X}$, distribution $\mathcal{D}$ over $\mathcal{X}$, and functions[21] $f, g : \mathcal{X} \to \mathbb{R}$, we call $\mathop{\mathbb{E}}_{x \sim \mathcal{D}} [f(x)g(x)]$ the *correlation* between f and g. In our setting, $\mathcal{D}$ is the standard multivariate Gaussian.

Definition 5.4 (SQ dimension [BFJ+94])**.** *Let* $\mathcal{X}$ *be a domain,* $\mathcal{F}$ *be a class of* $\mathcal{X} \to [-1,1]$ *functions, and* $\mathcal{D}$ *be a probability distribution over* $\mathcal{X}$*. The* SQ dimension of $\mathcal{F}$ under $\mathcal{D}$*, denoted* $\mathrm{SQdim}(\mathcal{F}, \mathcal{D})$*, is the largest* $s \in \mathbb{N}$ *such that there exist distinct functions* $f_1, \ldots, f_s \in \mathcal{F}$ *satisfying* $\left| \mathop{\mathbb{E}}_{x \in \mathcal{D}} [f_i(x) f_j(x)] \right| \leq \frac{1}{s}$ *for all distinct* $i, j \in [s]$.

Blum et al. [BFJ+94] showed that a lower bound on $\mathrm{SQdim}(\mathcal{F}, \mathcal{D})$ implies a lower bound for SQ algorithms that weakly learn $\mathcal{F}$. Concretely, we have the following result, whose simplified proof by [Szö09] we adapt to obtain a lower bound for distance approximation.

Theorem 5.5 (SQ dimension bound for weak learning)**.** *Let* $\mathcal{X}$ *be a domain,* $\mathcal{F}$ *be a class of* $\mathcal{X} \to [-1,1]$ *functions, and* $\mathcal{D}$ *be a probability distribution over* $\mathcal{X}$ *with* $\mathrm{SQdim}(\mathcal{F}, \mathcal{D}) = s$*. Then every (deterministic) SQ algorithm that, on input function* $f \in \mathcal{F}$*, outputs a function* $g : \mathcal{X} \to [-1,1]$ *whose correlation with* f *is* $\mathop{\mathbb{E}}_{x \sim \mathcal{D}} [f(x)g(x)] \geq s^{-1/3}$ *requires at least* $\frac{s^{1/3}}{2} - 1$ *queries to* $\textsc{Stat}(s^{-1/3})$.

Correlation queries. As observed by [BF02], in the distribution-specific setting, it suffices to consider statistical queries of the form $q(x, f(x)) = g(x)f(x)$ with $g : \mathcal{X} \to [-1,1]$. Given g, the oracle returns a value v satisfying $\left| \mathop{\mathbb{E}}_{x \sim \mathcal{D}} [g(x)f(x)] - v \right| \leq \tau$. Any general query can be simulated with two such queries, so we restrict attention to this form.

5.2 Prior work: high SQ dimension from packing numbers on the sphere

Diakonikolas, Kane and Zarifis [DKZ20] proved an SQ lower bound of $d^{\mathrm{poly}(1/\varepsilon)}$ for weakly learning halfspaces over $\mathbb{R}^d$ under the Gaussian distribution in high dimensions. Their proof constructs a set of Boolean functions with large SQ dimension, each correlated with a halfspace. This is done by selecting a large set of nearly orthogonal unit vectors in $\mathbb{R}^d$ and mapping each vector u to a Boolean function by projecting along u a one-dimensional k-alternating function satisfying a certain moment-matching condition.

Our proof builds upon the construction of [DKZ20], but we use a different packing bound suited to the low-dimensional setting. We start by extracting the following result from [DKZ20].

Theorem 5.6 (Implicit in [DKZ20])**.** *Let* $d, n, k \in \mathbb{N}$ *and* $\rho \in (0,1)$*. Suppose there exist vectors* $u_1, \ldots, u_n \in \mathbb{S}^{d-1}$ *satisfying* $|\langle u_i, u_j \rangle| \leq \rho$ *for all distinct* $i, j \in [n]$*. Then there exist functions* $f_1, \ldots, f_n : \mathbb{R}^d \to \{\pm 1\}$ *satisfying the following conditions.*

1. *For each* $i \in [n]$*, there exists a halfspace* $h_i : \mathbb{R}^d \to \{\pm 1\}$ *such that* $\mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [f_i(x) h_i(x)] \geq \frac{1}{2k}$.
2. $\left| \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [f_i(x) f_j(x)] \right| \leq 2\rho^{k+1}$ *for all distinct* $i, j \in [n]$.

[21] For simplicity, we omit from our statements tedious remarks about measurability of functions.

5.3 Packing slightly uncorrelated vectors on the low-dimensional sphere

We begin with a result from [CFJ13] on the asymptotic distribution of the minimum and maximum angles among n vectors drawn independently and uniformly from the unit sphere $\mathbb{S}^{d-1}$, for constant dimension d. In contrast, [DKZ20] (via a lemma from [DKS17]) relies on a different result from [CFJ13] applicable to the high-dimensional setting.

Theorem 5.7 (Theorem 2 of [CFJ13])**.** *Let $d \geq 2$ be an integer. Let $u_1 \ldots, u_n$ be sampled independently and uniformly at random from $\mathbb{S}^{d-1}$, and let $\theta_{\min}, \theta_{\max}$ denote the minimum and maximum angles, respectively, between any pairs of vectors u_i, u_j for distinct $i, j \in [n]$ Then as $n \to \infty$, the random variables $n^{2/(d-1)}\theta_{\min}$ and $n^{2/(d-1)}(\pi - \theta_{\max})$ converge weakly to the distribution with cumulative distribution function (CDF)*

$$F(x) = \begin{cases} 1 - e^{-K_d x^{d-1}} & \text{if } x \geq 0\,, \\ 0 & \text{if } x < 0\,, \end{cases}$$

where $K_d > 0$ is a constant that depends only on d.

Theorem 5.7 establishes weak convergence of distributions, i.e., pointwise convergence of the CDFs F_n to F. The following standard fact shows that, since F is continuous, convergence is uniform: for each $\delta > 0$, we can choose large enough n so that $|F_n(x) - F(x)| \leq \delta$ for all x.

Fact 5.8. *Suppose $(X_n)_{n\in\mathbb{N}}$ is a sequence of real-valued random variables converging weakly to X. Let F_n and F be the CDFs of X_n and X, respectively. If F is continuous, then $F_n \to F$ uniformly.*

We now prove our packing lemma.

Lemma 5.9 (Packing slightly uncorrelated vectors on the sphere)**.** *Let $d \geq 2$ be a fixed integer. Then for all constant $a > 0$ and $b \in \left(0, \frac{d-1}{4d}\right)$, and all sufficiently small $\varepsilon > 0$, there exists a set S of at least $(1/\varepsilon)^{bd}$ vectors on $\mathbb{S}^{d-1}$ such that, for all distinct $u, v \in S$, it holds that $|\langle u, v\rangle| \leq \varepsilon^{a\varepsilon}$.*

Proof. Let $n := \left\lceil (1/\varepsilon)^{bd} \right\rceil \leq 2(1/\varepsilon)^{bd}$ and $S = (u^1, \ldots, u^n)$ be a sequence of points sampled uniformly and independently from $\mathbb{S}^{d-1}$. We show that with positive probability, every pair $u^i \neq u^j$ in S with $i \neq j$ satisfies $\left|\left\langle u^i, u^j\right\rangle\right| \leq \varepsilon^{a\varepsilon}$, thus establishing the existence of such a set S.

Let $u, v \in \mathbb{S}^{d-1}$ be distinct and $\theta \in [0, \pi]$ be the angle between them. Let $\alpha := \varepsilon^{a\varepsilon}$. Since u, v are unit vectors, $|\langle u, v\rangle| = |\cos\theta|$. Thus,

$$|\langle u, v\rangle| \leq \alpha \iff |\cos\theta| \leq \alpha \iff \theta \in [\arccos(\alpha), \arccos(-\alpha)]\,. \tag{1}$$

We lower bound α by $\alpha = \varepsilon^{a\varepsilon} = e^{-a\varepsilon\ln(1/\varepsilon)} \geq 1 - a\varepsilon\ln\left(\frac{1}{\varepsilon}\right)$. Since arccos is a decreasing function, a sufficient condition for (1) to hold is

$$\theta \in [\arccos(1 - a\varepsilon\ln(1/\varepsilon)), \arccos(-1 + a\varepsilon\ln(1/\varepsilon))]\,.$$

By the series expansions for $x \to 0^+$,

$$\begin{aligned} \arccos(1 - x) &= \sqrt{2x} + O(x^{3/2}) \leq 2\sqrt{x}\,, \\ \arccos(-1 + x) &= \pi - \sqrt{2x} - O(x^{3/2}) \geq \pi - 2\sqrt{x}. \end{aligned}$$

Thus, for sufficiently small ε, it suffices if $\theta \in [C, \pi - C]$ where $C := 2\sqrt{a\varepsilon \ln(1/\varepsilon)}$. Equivalently,

$$\theta \geq C \text{ and } \pi - \theta \geq C\,. \tag{2}$$

By Theorem 5.7, all $u^i, u^j \in S$ with $i \neq j$ satisfy (2) and hence (1), except with probability at most $\mathbb{P}[\theta_{\min} < C] + \mathbb{P}[\pi - \theta_{\max} < C]$

$$= \mathbb{P}\left[n^{2/(d-1)}\theta_{\min} < n^{2/(d-1)}C\right] + \mathbb{P}\left[n^{2/(d-1)}(\pi - \theta_{\max}) < n^{2/(d-1)}C\right]. \tag{3}$$

For sufficiently small $\varepsilon > 0$, the RHS in the inequalities above is

$$\begin{aligned} n^{2/(d-1)}C &\leq 2\left(2(1/\varepsilon)^{bd}\right)^{2/(d-1)}\sqrt{a\varepsilon\ln(1/\varepsilon)} \\ &= 2\sqrt{a}\cdot 4^{1/(d-1)}\cdot\varepsilon^{\frac{1}{2}-\frac{2bd}{d-1}}\sqrt{\ln(1/\varepsilon)} \leq O(1)\cdot\varepsilon^{\Omega(1)}\,, \end{aligned}$$

where the first step uses the bound $n \leq 2(1/\varepsilon)^{bd}$ and the value of C, and the last step uses the assumption that $b < (d-1)/4d$.

By Theorem 5.7 and Fact 5.8, for all sufficiently small ε (i.e., all sufficiently large n), each of the probabilities from (3) is approximated by the limit CDF F from Theorem 5.7 up to an additive error of (say) 0.1. Thus, each of the probabilities from (3) is at most

$$0.1 + F(O(1)\cdot\varepsilon^{\Omega(1)}) = 0.1 + 1 - e^{-K_d(O(1)\cdot\varepsilon^{\Omega(1)})^{d-1}} = 0.1 + 1 - e^{-O(1)\cdot\varepsilon^{\Omega(1)}} < 0.11\,.$$

Thus, S satisfies the desired property with positive probability. □

5.4 A pseudorandom function for SQ algorithms

Combining Theorem 5.6 and Lemma 5.9, we obtain a large set $\mathcal{F}$ of Boolean functions with low pairwise correlations, where each $f \in \mathcal{F}$ is correlated with a halfspace. To derive an SQ lower bound for distance approximation (as in the proof of Theorem 5.5 for weak learning by [Szö09]), we would like to argue that an SQ algorithm cannot distinguish such an input f from "random noise", because an adversarial STAT oracle can consistently answer 0 until many queries are made.

However, we wish to obtain a lower bound for distance approximation of *functions*, not of randomized labelings (i.e., joint point-label distributions). Intuitively, a random function $f : \mathbb{R}^d \to \{\pm 1\}$ is indistinguishable from noise, but formally, such a function may not even be measurable, so it would be unsuitable as an input in the SQ model. To handle this, we construct a "pseudorandom" function f_0 that is sufficiently uncorrelated with every halfspace and every query of a target deterministic algorithm $\mathcal{A}$. To construct such f_0, we argue that the set of queries algorithm $\mathcal{A}$ makes on all inputs is finite and use this fact together with the connection between between VC dimension and covering numbers (for the class of halfspaces) to discretize the space $\mathbb{R}^d$ into sufficiently small cells, and then make f_0 balanced within each cell.

Our argument requires the queries of $\mathcal{A}$ to come from a finite set, regardless of the oracle's answers. To enforce this, we define a discretized oracle as follows. For all $x \in \mathbb{R}$ and $\tau \in (0, 1)$, let $\mathsf{round}_\tau(x)$ denote the integer multiple of τ that is closest to x, with rounding towards zero; that is, $\mathsf{round}_\tau(x) := \mathrm{sign}(x)\cdot\tau\cdot\lfloor|x|/\tau\rfloor$. A *$\tau$-rounding oracle* answers query $g : \mathcal{X} \to [-1, 1]$ with the quantity $\mathsf{round}_\tau\left(\mathop{\mathbb{E}}_{x\sim\mathcal{D}}[g(x)f(x)]\right)$. Such an oracle is a valid STAT(τ) oracle. If $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}[g(x)f(x)] \in (-\tau, \tau)$, then the τ-rounding oracle outputs 0, implementing the desired adversarial behavior.

Lemma 5.10. *Let $d \in \mathbb{N}$ and $\tau, \varepsilon > 0$. Let $\mathcal{A}$ be a deterministic SQ algorithm that takes a bounded number of bits of advice*[22] *and makes a bounded number of queries to a τ-rounding oracle. Then there exists a function $f_0 : \mathbb{R}^d \to \{\pm 1\}$ such that:*

1. $\mathsf{dist}_{\mathcal{N}(0,I)}(f_0, \mathcal{H}) \geq \frac{1}{2} - \frac{\varepsilon}{100}$.
2. *For each function $f : \mathbb{R}^d \to \{\pm 1\}$, every query $g : \mathbb{R}^d \to [-1, 1]$ made by $\mathcal{A}$ on input f satisfies* $\left| \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [g(x) f_0(x)] \right| \leq \frac{\tau}{2}$.

Proof. Since each output of the τ-rounding oracle comes from the finite set of integer multiples of τ in $[-1, 1]$, and $\mathcal{A}$ takes a finite number of bits of advice and terminates after finitely many queries, there are finitely many sequences of oracle outputs seen by $\mathcal{A}$ regardless of its input. Since $\mathcal{A}$ is deterministic, there exists a finite set $\mathcal{G}$ such that every query g made by $\mathcal{A}$ comes from $\mathcal{G}$.

Now, since the class $\mathcal{H}$ of halfspaces over $\mathbb{R}^d$ has finite VC dimension, a standard result in learning theory [Hau95, Corollary 1] says that $\mathcal{H}$ has finite covering number with respect to the Gaussian distribution, that is, there exists a finite set $\mathcal{R}$ of reference halfspaces such that every halfspace h in $\mathbb{R}^d$ satisfies $\mathsf{dist}_{\mathcal{N}(0,I)}(h, r) \leq \varepsilon/100$ for some $r \in \mathcal{R}$.

For each $x \in \mathbb{R}^d$, let its *color* $C(x)$ be the tuple containing all labels of x by the (rounded) query functions from $\mathcal{G}$ and the reference halfspaces from $\mathcal{R}$:

$$C(x) := \left(\left(\mathsf{round}_{\tau/2}(g(x)) \right)_{g \in \mathcal{G}}, (r(x))_{r \in \mathcal{R}} \right) .$$

The number of possible colors is finite. For each such color c, definite set $P_c = \{x \in \mathbb{R}^d : C(x) = c\}$. Then sets P_c partition $\mathbb{R}^d$. For each part P_c, let $L_c \cup R_c$ be a partition of P_c of balanced Gaussian measure, i.e., a partition satisfying $\mathop{\mathbb{P}}_{x \sim \mathcal{N}(0,I)} [x \in L_c] = \mathop{\mathbb{P}}_{x \sim \mathcal{N}(0,I)} [x \in R_c]$. Define the function f_0 as follows: for all colors c and $x \in P_c$, let $f_0(x) = +1$ if $x \in L_c$ and $f_0(x) = -1$ if $x \in R_c$.

Now we prove the two items of Lemma 5.10. For the first item, let h be a halfspace in $\mathbb{R}^d$. By construction, f_0 is $\frac{1}{2}$-far from every reference halfspace $r \in \mathcal{R}$ under the Gaussian distribution, since for each part P_c, halfspace r gives all of P_c the same label, whereas f_0 gives label $+1$ or -1 with equal conditional probability. Using the covering property, let $r \in \mathcal{R}$ be a reference halfspace satisfying $\mathsf{dist}_{\mathcal{N}(0,I)}(h, r) \leq \varepsilon/100$. Then, by the triangle inequality,

$$\mathsf{dist}_{\mathcal{N}(0,I)}(f_0, h) \geq \mathsf{dist}_{\mathcal{N}(0,I)}(f_0, r) - \mathsf{dist}_{\mathcal{N}(0,I)}(h, r) \geq \frac{1}{2} - \frac{\varepsilon}{100} .$$

Hence $\mathsf{dist}_{\mathcal{N}(0,I)}(f_0, \mathcal{H}) \geq \frac{1}{2} - \frac{\varepsilon}{100}$, as claimed.

For the second item, let function $g \in \mathcal{G}$ be a query of $\mathcal{A}$. By the construction of f_0,

$$\mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} \left[\mathsf{round}_{\tau/2}(g(x)) f_0(x) \right] = 0 , \tag{4}$$

because for each part P_c, every $x \in P_c$ has the same value of $\mathsf{round}_{\tau/2}(g(x))$, whereas f_0 gives label $+1$ or -1 with equal conditional probability. By applying (4) and then Jensen's inequality together

[22] We allow advice because in the final stage of the proof of Theorem 5.2, we convert a randomized algorithm to a deterministic algorithm with advice.

with the fact that $|f_0(x)| = 1$ for all $x \in \mathbb{R}^d$, we get

$$\left| \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [g(x) f_0(x)] \right| = \left| \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} \left[\left(g(x) - \mathsf{round}_{\tau/2}(g(x)) \right) f_0(x) \right] \right|$$
$$\leq \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} \left[\left| g(x) - \mathsf{round}_{\tau/2}(g(x)) \right| \right] \leq \frac{\tau}{2} . \qquad \square$$

5.5 Lower bound for deterministic algorithms

We combine the ingredients above to prove a lower bound for deterministic SQ algorithms for distance approximation of halfspaces. Our proof builds on the simplified argument of [Szö09] for weak learning (Theorem 5.5), which uses the correlation bound in the definition of SQ dimension to show that each statistical query, if answered adversarially, can rule out only a limited number of inputs.

Theorem 5.11. *Let $d \geq 2$ be a constant. Then every deterministic SQ distance approximation algorithm for halfspaces under $\mathcal{N}(0, I)$ with a bounded number of bits of advice requires $(1/\varepsilon)^{\Omega(d)}$ queries to $\mathrm{STAT}(\varepsilon^{\Omega(d)})$.*

Proof. Let $\gamma \in (0, 1)$ be a constant. First, we apply Lemma 5.9 with parameters $a = d$ and $b = \frac{\gamma(d-1)}{4d}$. This yields a set of $s = (1/\varepsilon)^{\frac{\gamma}{4}(d-1)}$ unit vectors with pairwise correlation at most $\rho = \varepsilon^{d\varepsilon}$. We apply Theorem 5.6 with parameter $k = \frac{1}{\varepsilon} - 1$ to this set of vectors to get a set $\mathcal{F}$ of s functions of the form $\mathbb{R}^d \to \{\pm 1\}$ satisfying the following conditions.

1. For each $f \in \mathcal{F}$, there is a halfspace $h : \mathbb{R}^d \to \{\pm 1\}$ satisfying $\mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [f(x)h(x)] \geq \frac{1}{2k} \geq \frac{\varepsilon}{2}$, and consequently, $\mathsf{dist}_{\mathcal{N}(0,I)}(f, \mathcal{H})$ is at most

$$\mathsf{dist}_{\mathcal{N}(0,I)}(f, h) = \mathop{\mathbb{P}}_{x \sim \mathcal{N}(0,I)} [f(x) \neq h(x)] = \frac{1}{2}\Big(1 - \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [f(x)h(x)]\Big) \leq \frac{1}{2} - \frac{\varepsilon}{4} .$$

2. For all distinct $f, f' \in \mathcal{F}$, we have $\left| \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [f(x)f'(x)] \right| \leq 2\rho^{k+1} = 2\varepsilon^d \leq \varepsilon^{\frac{\gamma}{4}(d-1)} = \frac{1}{s}$, where the second inequality holds for small enough $\varepsilon > 0$.

Let $\mathcal{A}$ be a deterministic SQ algorithm that makes less than $\frac{s^{1/3}-1}{2}$ queries to a τ-rounding oracle with $\tau = s^{-1/3}$. We apply Lemma 5.10 to get a function f_0 that has correlation at most $\frac{\tau}{2}$ with all queries made by $\mathcal{A}$ and satisfies $\mathsf{dist}_{\mathcal{N}(0,I)}(f_0, \mathcal{H}) \geq \frac{1}{2} - \frac{\varepsilon}{100}$. We define our "adversarial" oracle to always respond with 0 to queries made by $\mathcal{A}$. Then f_0 is consistent with all query answers. We will show that some $f_* \in \mathcal{F}$ is also consistent with all query answers. Then $\mathcal{A}$ cannot distinguish f_0 from f_*, thereby failing to approximate the distance to $\mathcal{H}$ well.

We use the inner product notation $\langle f, g \rangle := \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [f(x)g(x)]$. Fix a query $g : \mathbb{R}^d \to [0, 1]$. By definition, it has norm $\|g\| \leq 1$. Let the *bad set* $B := \{f \in \mathcal{F} : \langle g, f \rangle \geq \tau\}$. Then $\left\langle g, \sum_{f \in B} f \right\rangle \geq |B|\tau$. On the other hand, by the Cauchy-Schwarz inequality and condition 2 on the set $\mathcal{F}$,

$$\left\langle g, \sum_{f \in B} f \right\rangle^2 \leq \left\| \sum_{f \in B} f \right\|^2 = \sum_{f, f' \in B} \langle f, f' \rangle \leq \sum_{f \in B} \left(1 + \frac{|B| - 1}{s} \right) \leq |B| + \frac{|B|^2}{s} .$$

Combining the two inequalities and then substituting $\tau = s^{-1/3}$, we obtain $|B| \le \frac{s}{s\tau^2-1} = \frac{s}{s^{1/3}-1}$. Similarly, at most $\frac{s}{s^{1/3}-1}$ functions in $\mathcal{F}$ have correlation at most $-\tau$ with g. Hence, at most $\frac{2s}{s^{1/3}-1}$ functions in $\mathcal{F}$ are inconsistent with the answer 0 to query g. Since $\mathcal{A}$ makes less than $\frac{s^{1/3}-1}{2}$ queries, some $f_* \in \mathcal{F}$ is consistent with every query answer being 0. By construction, f_0 is also consistent with every query answer being 0. However, $\mathsf{dist}_{\mathcal{N}(0,I)}(f_0, \mathcal{H}) \ge \frac{1}{2} - \frac{\varepsilon}{100}$, whereas $\mathsf{dist}_{\mathcal{N}(0,I)}(f_*, \mathcal{H}) \le \frac{1}{2} - \frac{\varepsilon}{4}$. Thus $\mathcal{A}$ cannot approximate distance to $\mathcal{H}$ with parameter $\frac{\varepsilon}{16}$. Because $s^{1/3} = (1/\varepsilon)^{\frac{\gamma}{12}(d-1)}$, this concludes our proof. □

5.6 Lower bound for randomized algorithms

We now extend the lower bound in Theorem 5.11 to randomized SQ algorithms via a nonuniform derandomization argument. The key observation is that the lower bound for deterministic algorithms from Theorem 5.11 applies to *nonuniform* algorithms. Since a randomized algorithm with sufficiently high success probability implies, via a probabilistic argument, a deterministic algorithm with advice, we obtain a lower bound for randomized algorithms. A similar observation for weak learning algorithms appears in [BF02].

Proof of Theorem 5.2. Any algorithm with success probability at least 0.51 can be boosted to succeed with probability $1 - \delta$ by running the algorithm $O(\log(1/\delta))$ times and outputting the median of the estimates. Thus, fixing any constant $\gamma \in (0, 1)$, it suffices to show that every algorithm with failure probability $< \varepsilon^{\frac{\gamma}{2}(d-1)}$ requires $(1/\varepsilon)^{\Omega(d)}$ SQ queries. Let $\mathcal{B}$ be a randomized SQ distance approximation algorithm for halfspaces under $\mathcal{N}(0, I)$ with failure probability $< \varepsilon^{\frac{\gamma}{2}(d-1)}$. By a union bound over the set $\mathcal{F} \cup \{f_0\}$ from the proof of Theorem 5.11, which consists of $(1/\varepsilon)^{\frac{\gamma}{4}(d-1)} + 1 < (1/\varepsilon)^{\frac{\gamma}{2}(d-1)}$ functions, there is a random seed r such that algorithm $\mathcal{B}$ run with random seed r outputs an accurate estimate on all of these functions. Let $\mathcal{A}$ be a deterministic algorithm obtained from $\mathcal{B}$ by using r as advice instead of the random seed. Then $\mathcal{A}$ succeeds on $\mathcal{F} \cup \{f_0\}$ and has the same query complexity as $\mathcal{B}$. By Theorem 5.11, we conclude that $\mathcal{A}$ (and hence $\mathcal{B}$) requires $(1/\varepsilon)^{\Omega(d)}$ queries to $\textsc{Stat}(\varepsilon^{\Omega(d)})$. □

References

[AL13] Amir Abboud and Kevin Lewi. Exact weight subgraphs and the k-SUM conjecture. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, volume 7965 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2013.

[BBBY12] Maria-Florina Balcan, Eric Blais, Avrim Blum, and Liu Yang. Active property testing. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 21–30. IEEE Computer Society, 2012.

[BF02] Nader H. Bshouty and Vitaly Feldman. On using extended statistical queries to avoid membership queries. *J. Mach. Learn. Res.*, 2:359–395, 2002.

[BFH21] Eric Blais, Renato Ferreira Pinto Jr, and Nathaniel Harms. VC dimension and distribution-free sample-based testing. In Samir Khuller and Virginia Vassilevska

Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 504–517. ACM, 2021.

[BFJ+94] Avrim Blum, Merrick L. Furst, Jeffrey C. Jackson, Michael J. Kearns, Yishay Mansour, and Steven Rudich. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 253–262. ACM, 1994.

[BHK25] Lorenzo Beretta, Nathaniel Harms, and Caleb Koch. Testing juntas optimally with samples. *CoRR*, abs/2505.04604, 2025.

[BHR05] Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF properties are hard to test. *SIAM J. Comput.*, 35(1):1–21, 2005.

[BKST23] Guy Blanc, Caleb Koch, Carmen Strassle, and Li-Yang Tan. A strong composition theorem for junta complexity and the boosting of property testers. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1757–1777. IEEE, 2023.

[BMR16] Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant testers of image properties. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 90:1–90:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[BMR22] Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant testers of image properties. *ACM Trans. Algorithms*, 18(4):37:1–37:39, 2022.

[BN13] Zvika Brakerski and Moni Naor. Fast algorithms for interactive coding. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 443–456. SIAM, 2013.

[Bre76] Richard P Brent. Multiple-precision zero-finding methods and the complexity of elementary function evaluation. In *Analytic Computational Complexity*, pages 151–176. Academic Press, 1976.

[BS23] Eric Blais and Cameron Seth. Testing graph properties with the container method. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1787–1795. IEEE, 2023.

[BY22] Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing - Problems and Techniques*. Springer, 2022.

[CFJ13] T Tony Cai, Jianqing Fan, and Tiefeng Jiang. Distributions of angles in random packing on spheres. *Journal of Machine Learning Research*, 14(136):1837–1864, 2013.

[CG18] Clément L. Canonne and Tom Gur. An adaptivity hierarchy theorem for property testing. *Comput. Complex.*, 27(4):671–716, 2018.

[CGRZ24] Eshan Chattopadhyay, Mohit Gurumukhani, Noam Ringach, and Yunya Zhao. Two-sided lossless expanders in the unbalanced setting. *CoRR*, abs/2409.04549, 2024.

[Che25] Yeyuan Chen. Unique-neighbor expanders with better expansion for polynomial-sized sets. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 3335–3362. SIAM, 2025.

[CP22] Xi Chen and Shyamal Patel. Distribution-free testing for halfspaces (almost) requires PAC learning. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 1715–1743. SIAM, 2022.

[CR73] Stephen A. Cook and Robert A. Reckhow. Time bounded random access machines. *J. Comput. Syst. Sci.*, 7(4):354–375, 1973.

[CRVW02] Michael R. Capalbo, Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 659–668. ACM, 2002.

[DEM96] David P. Dobkin, David Eppstein, and Don P. Mitchell. Computing the discrepancy with applications to supersampling patterns. *ACM Trans. Graph.*, 15(4):354–376, 1996.

[DKK+21] Ilias Diakonikolas, Daniel M. Kane, Vasilis Kontonis, Christos Tzamos, and Nikos Zarifis. Agnostic proper learning of halfspaces under Gaussian marginals. In Mikhail Belkin and Samory Kpotufe, editors, *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*, volume 134 of *Proceedings of Machine Learning Research*, pages 1522–1551. PMLR, 2021.

[DKR23] Ilias Diakonikolas, Daniel Kane, and Lisheng Ren. Near-optimal cryptographic hardness of agnostically learning halfspaces and ReLU regression under Gaussian marginals. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 7922–7938. PMLR, 2023.

[DKS17] Ilias Diakonikolas, Daniel M. Kane, and Alistair Stewart. Statistical query lower bounds for robust estimation of high-dimensional Gaussians and Gaussian mixtures. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 73–84. IEEE Computer Society, 2017.

[DKZ20] Ilias Diakonikolas, Daniel Kane, and Nikos Zarifis. Near-optimal SQ lower bounds for agnostically learning halfspaces and ReLUs under Gaussian marginals. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[EGR25] Talya Eden, Ludmila Glinskih, and Sofya Raskhodnikova. Fast agnostic learners in the plane. *CoRR*, abs/2510.18057, 2025.

[Eri96] Jeff Erickson. New lower bounds for convex hull problems in odd dimensions. In Sue Whitesides, editor, *Proceedings of the Twelfth Annual Symposium on Computational Geometry, Philadelphia, PA, USA, May 24-26, 1996*, pages 1–9. ACM, 1996.

[FGKP09] Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM J. Comput.*, 39(2):606–645, 2009.

[FHKS17] Martin Fink, John Hershberger, Nirman Kumar, and Subhash Suri. Hyperplane separability and convexity of probabilistic point sets. *J. Comput. Geom.*, 8(2):32–57, 2017.

[FLN+02] Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 474–483. ACM, 2002.

[FR21] Nimrod Fiat and Dana Ron. On efficient distance approximation for graph properties. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1618–1637. SIAM, 2021.

[GGR98] Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.

[GJ22] Etienne Grandjean and Louis Jachiet. Which arithmetic operations can be performed in constant time in the RAM model with addition? *CoRR*, abs/2206.13851, 2022.

[GKNR12] Oded Goldreich, Michael Krivelevich, Ilan Newman, and Eyal Rozenberg. Hierarchy theorems for property testing. *Comput. Complex.*, 21(1):129–192, 2012.

[GKSV24] Aravind Gollakota, Adam R. Klivans, Konstantinos Stavropoulos, and Arsen Vasilyan. An efficient tester-learner for halfspaces. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[GO95] Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom.*, 5:165–185, 1995.

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

[Gol19] Oded Goldreich. Hierarchy theorems for testing properties in size-oblivious query complexity. *Comput. Complex.*, 28(4):709–747, 2019.

[GR09] Venkatesan Guruswami and Prasad Raghavendra. Hardness of learning halfspaces with noise. *SIAM J. Comput.*, 39(2):742–765, 2009.

[Har19] Nathaniel Harms. Testing halfspaces over rotation-invariant distributions. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 694–713. SIAM, 2019.

[Hau95] David Haussler. Sphere packing numbers for subsets of the Boolean n-cube with bounded Vapnik-Chervonenkis dimension. *J. Comb. Theory A*, 69(2):217–232, 1995.

[HLM+25] Jun-Ting Hsieh, Ting-Chun Lin, Sidhanth Mohanty, Ryan O'Donnell, and Rachel Yun Zhang. Explicit two-sided vertex expanders beyond the spectral barrier. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 833–842. ACM, 2025.

[HMMP24] Jun-Ting Hsieh, Theo McKenzie, Sidhanth Mohanty, and Pedro Paredes. Explicit two-sided unique-neighbor expanders. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 788–799. ACM, 2024.

[HvdH21] David Harvey and Joris van der Hoeven. Integer multiplication in time $O(n \log n)$. *Annals of Mathematics*, 193(2):563–617, 2021.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[Kea98] Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998.

[KKMS08] Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio. Agnostically learning halfspaces. *SIAM J. Comput.*, 37(6):1777–1805, 2008.

[KLR25] Esty Kelman, Ephraim Linder, and Sofya Raskhodnikova. Online versus offline adversaries in property testing. In Raghu Meka, editor, *16th Innovations in Theoretical Computer Science Conference, ITCS 2025, January 7-10, 2025, Columbia University, New York, NY, USA*, volume 325 of *LIPIcs*, pages 65:1–65:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025.

[KST23] Caleb Koch, Carmen Strassle, and Li-Yang Tan. Properly learning decision trees with queries is NP-hard. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 2383–2407. IEEE, 2023.

[LN11] Oded Lachish and Ilan Newman. Testing periodicity. *Algorithmica*, 60(2):401–420, 2011.

[MORS09] Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing ± 1-weight halfspace. 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 646–657. Springer, 2009.

[MORS10a] Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing halfspaces. *SIAM J. Comput.*, 39(5):2004–2047, 2010.

[MORS10b] Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing (subclasses of) halfspaces. In *Property Testing - Current Research and Surveys*, volume 6390, pages 334–340. Springer, 2010.

[MP21] Michael Matheny and Jeff M. Phillips. Approximate maximum halfspace discrepancy. In Hee-Kap Ahn and Kunihiko Sadakane, editors, *32nd International Symposium on*

Algorithms and Computation, ISAAC 2021, December 6-8, 2021, Fukuoka, Japan, volume 212 of *LIPIcs*, pages 4:1–4:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[NS11] Ilan Newman and Christian Sohler. Every property of hyperfinite graphs is testable. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 675–684. ACM, 2011.

[Ona12] Krzysztof Onak. On the complexity of learning and testing hyperfinite graphs. Available from the author's website, 2012.

[PPSZ05] Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *J. ACM*, 52(3):337–364, 2005.

[PRR06] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.

[PRW22] Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of Boolean functions. *Random Struct. Algorithms*, 60(2):233–260, 2022.

[Ras03] Sofya Raskhodnikova. *Property testing: theory and applications.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2003.

[Rey20] Lev Reyzin. Statistical queries and statistical algorithms: Foundations and applications. *CoRR*, abs/2004.00557, 2020.

[RS96] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

[RV23] Ronitt Rubinfeld and Arsen Vasilyan. Testing distributional assumptions of learning algorithms. 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 1643–1656. ACM, 2023.

[RVW00] Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, Redondo Beach, California, USA, November 12-14, 2000*, pages 3–13. IEEE Computer Society, 2000.

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press, 2014.

[Spi96] Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Trans. Inf. Theory*, 42(6):1723–1731, 1996.

[SS24] Asaf Shapira and Henrique Stagni. A tight bound for testing partition properties. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 4305–4320. SIAM, 2024.

[Szö09] Balázs Szörényi. Characterizing statistical query learning: Simplified notions and proofs. In Ricard Gavaldà, Gábor Lugosi, Thomas Zeugmann, and Sandra Zilles, edi-

tors, *Algorithmic Learning Theory, 20th International Conference, ALT 2009, Porto, Portugal, October 3-5, 2009. Proceedings*, volume 5809 of *Lecture Notes in Computer Science*, pages 186–200. Springer, 2009.

[Val84] Leslie G. Valiant. A theory of the learnable. In Richard A. DeMillo, editor, *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 436–445. ACM, 1984.

[Vas15] Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk). In Thore Husfeldt and Iyad A. Kanj, editors, *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, volume 43 of *LIPIcs*, pages 17–29. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[VC71] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

[WW20] Virginia Vassilevska Williams and Ryan Williams. Lecture 9: Algorithms for k-SUM. Lecture notes for 6.S078: Fine-Grained Algorithms and Complexity, MIT CSAIL, October 2020.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il