

MIT Open Access Articles

Property Testing with Online Adversaries

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Omri Ben Eliezer, Esty Kelman, Uri Meir, and Sofya Raskhodnikova. 2025. Property Testing with Online Adversaries. ACM Trans. Comput. Theory (December 2025).

As Published: <http://dx.doi.org/10.1145/3778165>

Publisher: ACM

Persistent URL: <https://hdl.handle.net/1721.1/164543>

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of use: Creative Commons Attribution



Property Testing with Online Adversaries*

OMRI BEN ELIEZER[†], Massachusetts Institute of Technology, Cambridge, United States and Technion – Israel Institute of Technology, Haifa, Israel

ESTY KELMAN[‡], CS and CDS, Boston University, Boston, United States and Massachusetts Institute of Technology, Cambridge, United States

URI MEIR, Tel Aviv University, Tel Aviv, Israel

SOFYA RASKHODNIKOVA, Boston University, Boston, United States

The online manipulation-resilient testing model, proposed by Kalemaj, Raskhodnikova and Varma (Theory of Computing 2023), studies property testing in situations where access to the input degrades continuously and adversarially. Our main contributions are as follows:

- An extension of the model, introducing *batch queries* where multiple queries are made and answered between each round of manipulation, and *fractional manipulation rate*, where the adversary makes less than one manipulation per round.
- New optimal testers for linearity testing of Boolean functions in the original online and offline models.
- A new lower-bound for testing low-degree of Boolean functions in the original model which can be overcome by an algorithm using batch queries.
- Efficient testers for local properties of sequences when the manipulation rate is fractional. Specifically, for sortedness, we show a sharp transition from optimal query complexity to the impossibility of testability, depending on the manipulation rate.

CCS Concepts: • **Theory of computation** → **Property testing and sublinear algorithms**.

Additional Key Words and Phrases: Linearity testing, low-degree testing, Reed-Muller codes, testing properties of sequences, erasure-resilience.

1 Introduction

The online manipulation-resilient testing model, proposed by Kalemaj, Raskhodnikova and Varma [40], studies property testing in contexts where access to the input is controlled by an adversary and degrades over time. Their motivation includes situations where access to data is restricted because of privacy or is misrepresented by someone trying to cover fraud while having to release some data in response to subpoenas. Another motivation is testing properties of a massive ever-changing object, where probing the object affects it. For example, a navigation

*A preliminary version of this work appeared in the proceedings of ITCS 2024 [16].

[†]Supported by an Alon Scholarship and by a Taub Family Foundation “Leaders in Science & Technology” Fellowship.

[‡]Supported by the Computer Science and the Computing and Data Science Departments, Boston University. Supported in part by an Amazon Faculty Research Award to Arnab Bhattacharyya, in part by ERC grant 834735, and in part by NSF TRIPODS program (award DMS-2022448).

Authors’ Contact Information: Omri Ben Eliezer, Massachusetts Institute of Technology, Cambridge, Massachusetts, United States and Technion – Israel Institute of Technology, Haifa, Israel; e-mail: omribene@cs.technion.ac.il; Esty Kelman, CS and CDS, Boston University, Boston, Massachusetts, United States and Massachusetts Institute of Technology, Cambridge, Massachusetts, United States; e-mail: ekelman@mit.edu; Uri Meir, Tel Aviv University, Tel Aviv, Tel Aviv, Israel; e-mail: urimeir.cs@gmail.com; Sofya Raskhodnikova, Boston University, Boston, Massachusetts, United States; e-mail: sofya.raskhodnikova@gmail.com.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 1942-3462/2025/12-ART

<https://doi.org/10.1145/3778165>

app might adjust its estimate of which routes are congested in response to queries. Such situations are plentiful in data analysis of modern social and transportation networks.

Modeling access to data as degrading adversarially allows the algorithm designer to ensure that algorithms work in situations where the degradation to access is too complicated to model accurately or where it is desirable to avoid relying on distributional assumptions in the algorithm analyses. The general goal is to understand what properties of the input can be gleaned without any distributional assumptions, and even in extremely adversarial regimes. Specifically, we are interested in the regime where access to the data or the data itself could be manipulated by an adversary in response to the algorithm’s actions during its execution. From the theoretical point of view, this investigation sheds light on the structure of witnesses for different properties.

In the online manipulation-resilient testing model of [40], after each query to the input object is answered, the adversary can manipulate (i.e., erase or corrupt) a fixed number of input values. The input is represented by a function f on an arbitrary finite domain. The algorithm accesses it by specifying a query point x in the domain and receiving the answer to the query from the oracle that represents the current state of the manipulated input.

In this work, we investigate a more nuanced version of the online manipulation-resilient testing model. We allow the adversary to either make changes at a fixed rate (as in the model of [40]) or accumulate the number of allowed manipulations in order to utilize them at any subsequent step. We also study arbitrary rates of erasures in addition to integer rates investigated by [40], e.g., allowing the adversary to manipulate one point after every other query. Finally, we study *batch queries* that are grouped together, with no manipulations allowed until all queries in a batch have been answered. In addition to allowing us to design fast testers that overcome old and new lower bounds, studying batch queries is motivated by a connection to Maker-Breaker games, discussed later (in Section 1.3).

We give online testers for several well studied properties. For Boolean functions on the domain $\{0, 1\}^n$, we investigate linearity and, more generally, the class of functions of low degree, that is, of degree at most d over \mathbb{F}_2 for specified d . (Note that low-degree testing is equivalent to testing Reed-Muller codes.) For linearity, our online tester has optimal query complexity. For the degree- d property, we present a lower bound that nearly matches the upper bound in the concurrent work of Minzer and Zheng [47] and then show how to overcome the lower bound by using batch queries. Finally, we consider local properties of functions $f : [n] \rightarrow \mathbb{R}$. (We use $[n]$ to represent $\{1, 2, \dots, n\}$). Kalemaj et al. proved that two important properties in that class—sortedness and the Lipschitz property—are not testable with any number of queries when the adversary erases one point per query. To overcome this impossibility result, we consider adversaries that make less than one erasure per query. For sortedness, we characterize the rate of erasures for which online testing can be performed, exhibiting a sharp transition from optimal query complexity to impossibility of testability. Our online tester works for general *local properties*, i.e., properties that are characterized by families of forbidden pairs, formally defined in Section 1.1.3. Moreover, we show that with batches of size 2, we can test all local properties with the same query complexity as with no manipulations even when the number of manipulations per query (or a batch of queries) is nearly-linear in the input size.

Our online testers avoid querying manipulated data by heavily relying on randomness: the marginal distribution of each query is nearly uniform on a large subset of the domain. This ensures that each query is unlikely to be a point previously manipulated by the adversary (for any adversarial strategy). Consequently, we can separately analyze the probabilities of two important events: selecting a set of queries that exhibit a violation of the property (i.e., a *witness*) and querying data modified by the adversary. When selecting a witness is significantly more likely than encountering a modification, the tester is likely to find a witness that has not been tampered with. This approach allows us to design online testers that are resilient to both erasures and corruptions. In the presence of erasures, our testers achieve one-sided error (i.e., they never err on an input satisfying the property), while corruptions might incur errors on any input, resulting in two-sided error testers, which seems unavoidable.

1.1 Our Results and Techniques

We study two types of functions: Boolean functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and real-valued functions $f : [n] \rightarrow \mathbb{R}$, i.e., sequences. The testing algorithm gets oracle access to the input function via one of the specified online adversarial oracles. Initially, the oracle is identical to the input function. The oracles (formally defined in Section 2.1) vary in how they are allowed to modify the input. The main distinctions are erasures vs. arbitrary corruptions (of input values) and when modifications can occur. In the model of [40], the adversary gets a parameter $t \in \mathbb{N}$ and is allowed to modify t function values after answering each query. We call such an adversary *t-online fixed-rate*. We extend the definition to $t \in \mathbb{R}^+$ and also consider *t-online budget-managing* adversaries that get allocated t modifications after answering each query, but can use their allocation at any subsequent time step in the computation (see Definition 2.1). Budget-managing adversaries are at least as powerful as fixed-rate adversaries. Nevertheless, for some of the properties we study, we are able to match hardness results for (weaker) fixed-rate adversaries with algorithms that work even in the presence of (stronger) budget-managing adversaries.

In addition to the corruption rate t , our algorithms get the proximity parameter $\epsilon \in (0, 1)$, as in the standard property testing model, and have to distinguish functions that have the specified property from functions that differ in at least an ϵ -fraction of the domain from any function with the property. See Section 2.1 and Definition 2.4 for formal definitions.

We start our investigation with Boolean functions on the Boolean cube. The properties we study are linearity and, more generally, of being a polynomial of degree at most d for a given $d \in \mathbb{N}$.

1.1.1 Linearity Testing. Introduced in the pioneering work of [21], linearity testing has been extensively investigated; see, e.g., [8–11, 17, 30, 36, 40, 41, 63–68] and the survey in [58]. A function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is called *linear* if $f(x_1) + f(x_2) = f(x_1 \oplus x_2)$ for all $x_1, x_2 \in \{0, 1\}^n$, where addition is mod 2, and \oplus denotes bitwise XOR. We present an optimal tester for linearity that is resilient to online erasures, as well as online corruptions, even when the adversary is budget-managing.

THEOREM 1.1 (OPTIMAL ONLINE LINEARITY TESTER). *For all $n \in \mathbb{N}$, $\epsilon \in (0, 1/2]$, and t satisfying $t \log^2 t \leq 2^{-21} \cdot \epsilon^{2.5} 2^{n/2}$, there exists an ϵ -tester for linearity of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ that works in the presence of a t -online erasure (or corruption) budget-managing adversary and makes $O(\max\{\frac{1}{\epsilon}, \log t\})$ queries. In the case of erasure adversary, the tester has 1-sided error.*

In comparison, the linearity tester from [40] works for a smaller range of t (specifically, $t \leq c_0 \cdot \epsilon^{5/4} 2^{n/4}$ for some constant c_0) and makes $O(\min\{\frac{1}{\epsilon} \log \frac{t}{\epsilon}, \frac{t}{\epsilon}\})$ queries.

The linearity tester in Theorem 1.1 has optimal query complexity both for fixed-rate and budget-managing adversaries, in the case of erasures (and, consequently, in the more challenging setting with corruptions). The optimality follows from the known query lower bounds of $\Omega(\frac{1}{\epsilon})$ with no erasures and $\Omega(\log t)$ for fixed-rate adversarial erasures [40, Theorem 1.4].

Our online linearity tester improves on the tester in [40] both in terms of query complexity and in terms of simplicity of the tester. The classical linearity test of [21] looks for witnesses of nonlinearity consisting of three points x_1, x_2 and $x_1 \oplus x_2$ that satisfy $f(x_1) + f(x_2) \neq f(x_1 \oplus x_2)$, where addition is mod 2. Kalemaj et al. generalized it to witnesses consisting of any even number of points and their XOR. Let XORTEST_k denote the test that picks k points uniformly at random and checks if these points and their XOR form a witness of nonlinearity. We improve upon the soundness analysis of this test and use it to get an optimal online-erasure-resilient linearity tester. Specifically, in Lemma 3.1, we show that XORTEST_k rejects every function that is ϵ -far from linearity with probability proportional to $k\epsilon$, as long as $k\epsilon$ is at most a small constant. It implies that XORTEST_k can be used in the standard linearity testing setting (without erasures) to obtain another optimal $O(\frac{1}{\epsilon})$ -query tester: XORTEST_k can be repeated $O(1/(k\epsilon))$ times to obtain constant probability of error. To the best of our knowledge, prior to

our work, only testers based on the BLR test (i.e., XORTEST_2) have been shown to be optimal in the standard linearity testing setting.

Another important ingredient in the analysis of our online linearity tester is bounding the probability of seeing an erasure. A good bound for this event ensures that our tester is resilient to corruptions (not just erasures). It also allows us to obtain a clean online tester that is based on multiple simulations of XORTEST_k , where each simulation initially samples $2k$ points to fool the adversary and then selects a random subset of size k for the XOR as the final query. In contrast, the online linearity tester of [40] is more complicated: it relies on work investment strategy.

1.1.2 Low-Degree Testing. Low-degree testing, equivalent to local testing of Reed-Muller codes, is a natural generalization of linearity testing. It has been investigated, e.g., in [1, 2, 5, 6, 20, 25, 30, 32, 33, 35, 39, 40, 42, 43, 48, 49, 60–63, 65]. For a $d \in \mathbb{N}$, let \mathcal{P}_d denote the set of all polynomials $p : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ of degree at most d , that is, functions $p(x)$ that can be represented as a sum of monomials¹ of the form $\prod_{i \in S} x[i]$ over \mathbb{F}_2 , where $x = (x[1], \dots, x[n])$ is a vector of n bits and $|S| \leq d$. In the standard property testing model, \mathcal{P}_d can be ϵ -tested with $O(\frac{1}{\epsilon} + 2^d)$ queries by repeating the following test of Alon et al. [1]: select $d + 1$ points in \mathbb{F}_2^n uniformly at random, query f on all of their linear combinations, and accept iff the sum of the returned values is 0. This tester was analyzed by Alon et al. [1], with an asymptotic improvement by Bhattacharyya et al. [20], and the matching lower bound of $\Omega(\frac{1}{\epsilon} + 2^d)$ on query complexity was proved in [1].

As d grows, the test of Alon et al. becomes more structured: the number of points queried is exponential in the number of points selected at random. This makes it hard to adapt to the online model, since the adversary can predict and erase the points needed by the tester. We show that there is an underlying reason for this difficulty: low-degree testing for $d > 1$ is strictly harder than testing linearity in terms of the dependence on t , the rate of erasures.

THEOREM 1.2 (LOWER BOUND FOR LOW-DEGREE TESTING). *Fix an integer $d > 1$ and $\epsilon \in (0, 1/2)$. Let \mathcal{P}_d be the set of polynomials of degree at most d over \mathbb{F}_2 . There exists $n_0 = n_0(d, \epsilon)$, such that for all $n \geq n_0$, every ϵ -tester of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ for property \mathcal{P}_d that works in the presence of a t -online fixed-rate erasure adversary must make $\Omega\left(\binom{\log t}{d}\right)$ queries.*

Our lower bound is nearly tight in terms of the dependence on t and d : in a concurrent work, Minzer and Zheng [47] show that \mathcal{P}_d can be tested with $O\left(\frac{1}{\epsilon} \log^{3d+3} \frac{t}{\epsilon}\right)$ queries with t -online fixed-rate erasure adversaries. Thus, the query complexity of \mathcal{P}_d is $\log^{\Theta(d)} t$ (for constant values of ϵ, d).

To prove our lower bound, we define an extended representation $\text{Ext}_d(x)$ for each vector $x \in \mathbb{F}_2^n$, where each entry of $\text{Ext}_d(x)$ is an evaluation of a monomial of degree at most d on input x . We consider the adversarial strategy of erasing function values on all points whose extended representations are in the span of the extended representations of previous queries. We use [13, Lemma 1.4] (or, equivalently, [45, Theorem 1.5]) to demonstrate that when the number of queries is, roughly, at most $\binom{\log t}{d}$, the adversary can successfully execute this strategy. Finally, we apply Yao’s minimax principle to show that in this case no online tester can distinguish random polynomials of degree d from random functions.

We overcome the lower bound by allowing batch queries, where the tester gets a group of b queries answered between manipulations of the data. The original model corresponds to $b = 1$. Since the test of Alon et al. makes 2^{d+1} queries, it can be used directly in an online tester with batch size $b = 2^{d+1}$, achieving query complexity $O(\frac{1}{\epsilon} + 2^d)$, the same as in the offline regime. For completeness, in Section A, we analyze the rate of erasures t

¹To be consistent with previous work, we allow $S = \emptyset$. The property \mathcal{P}_1 is *affinity*, and linear functions (discussed in Section 1.1.1) are affine with the additional requirement that the constant in the polynomial representation is 0.

achievable in this setting. It is natural to ask for which batch sizes we can achieve overhead polynomial in $d \log t$ over the offline query complexity. We show how to do it for $b = 2^{d-1}$, i.e., with the batch size equal to one quarter of the points needed for the smallest witness of not satisfying \mathcal{P}_d . In particular, for quadraticity, it corresponds to batches of size $b = 2$, a natural extension of testing linearity with batches of size 1.

THEOREM 1.3. *There exists a constant $c > 0$ such that for all $n, d \in \mathbb{N}$, $\epsilon \in (0, 1/2)$, and t satisfying $d < \frac{n}{11}$ and $t \log^7 t \leq c \cdot \epsilon^{2.5} d^{-7} 2^{(n-11d)/2}$, there exists an ϵ -tester for property \mathcal{P}_d of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ that works in the presence of a $(2^{d-1}, t)$ -online erasure budget-managing adversary and makes $O(\frac{1}{\epsilon} + 2^{3d} (d + \log t)^3)$ queries.*

Our online low-degree tester is a natural generalization of our linearity tester and of the tester of [1]. The witnesses we consider generalize the $(d + 1)$ -dimensional cubes formed by the points queried by the tester of [1] to *chains of cubes* (see Figure 1).

We show chains of cubes are viable witnesses as they are a special case of k -local characterizations defined in the seminal work of Kaufman and Sudan [44] on algebraic property testing (see exact definitions in Section 5). In addition, our *chains-of-cubes* test shares important features with the test of Alon et al. This allows us to show, by generalizing an argument from [20], that for small values of ϵ , the probability of seeing a violation increases linearly in the size of the witness.

To test with a chain of cubes, the tester first declares $d - 1$ directions that form a linear subspace A . Then, intuitively, it runs our online linearity tester, replacing each query x with a batch query $x + A$ (all points in this affine subspace). To analyze erasure resilience, whenever the adversary erases point x , we allow it to erase the entire space $x + A$ for free. This allows us to analyze the probability of seeing an erasure over the quotient group \mathbb{F}_2^n/A , which is isomorphic to \mathbb{F}_2^{n-d+1} , essentially reducing the analysis to that of the online linearity tester (because, over this space, each query and erasure are of a single point).

1.1.3 Testing Local Properties. Next we investigate properties of sequences, represented by functions $f : [n] \rightarrow \mathbb{R}$. Kalemaj et al. showed that two fundamental properties of sequences, *sortedness* and the *Lipschitz property*, are not testable with any number of queries in the presence of a fixed-rate adversary making $t = 1$ erasures per query. Are these properties testable in models with fewer erasures than queries?

We explore this question in depth. The answer is generally positive, but differs between the two adversarial manipulation models we study: fixed-rate and budget-managing. Our results are optimal, and the query complexity in many regimes matches that of the standard (offline) property testing model. Our online testers work in the general framework of local properties [14].

Before stating our results, we define sortedness, the Lipschitz property, and the general class of local properties. A sequence $f : [n] \rightarrow \mathbb{R}$ is sorted if $f(x) \leq f(y)$ for all $x < y$, where $x, y \in [n]$. Sortedness is one of the most investigated properties in the context of property testing (see [12, 19, 23, 28, 29, 31, 53, 55] and the survey in [56]). A sequence $f : [n] \rightarrow \mathbb{R}$ is Lipschitz if $|f(x + 1) - f(x)| \leq 1$ for all $x \in [n - 1]$. The Lipschitz property has been studied in [4, 18, 22, 23, 26, 38] and has applications to data privacy. Both sortedness and the Lipschitz property belong to the class of local properties, defined by [14]. A property \mathcal{P} of sequences $f : [n] \rightarrow \mathbb{R}$ is *local*² if there exists a family \mathcal{F} of forbidden pairs $(a, b) \in \mathbb{R}^2$ such that

$$f \in \mathcal{P} \Leftrightarrow \forall i \in [n - 1] \forall (a, b) \in \mathcal{F} : (f(i), f(i + 1)) \neq (a, b).$$

In such a case, we say that \mathcal{P} is characterized by the family \mathcal{F} . Sortedness is characterized by $\mathcal{F} = \{(a, b) : a > b\}$; that is, a sequence is sorted if and only if it does not contain a pair of *consecutive* elements with decreasing values. The Lipschitz property is characterized by $\mathcal{F} = \{(a, b) : |a - b| > 1\}$.

²This class of properties is called 2-local in [14], where more general k -local properties are defined as characterized by families of forbidden k -tuples. For simplicity and clarity of exposition, we focus on 2-local properties, but our results easily generalize to k -local properties (with increased batch sizes).

Results for batch size $b = 1$. We give tight bounds on the rate of erasures and corruptions online algorithms can handle in the presence of *budget-managing* adversaries.

THEOREM 1.4 (TESTING LOCAL PROPERTIES IN PRESENCE OF BUDGET-MANAGING ADVERSARY WITH BATCH SIZE 1). *There exist absolute constants $0 < c < C$ satisfying the following. For all $\epsilon \in (0, 1)$ and $n \geq C/\epsilon$:*

- (1) *For every local property \mathcal{P} of sequences $f: [n] \rightarrow \mathbb{R}$, and every $t \leq c\epsilon$, there exists an ϵ -tester for \mathcal{P} that works in the presence of a t -online erasure budget-managing adversary and makes $O\left(\frac{\log(\epsilon n)}{\epsilon}\right)$ queries.*
- (2) *For every $t \geq C\epsilon$, there is no ϵ -tester for sortedness of sequences $f: [n] \rightarrow \mathbb{R}$ that works in the presence of a t -online erasure budget-managing adversary (with any number of queries).*

Both of these results hold (with the same parameters) if erasures are replaced with corruptions. In the case of the erasure adversary, the tester has one-sided error.

The tester in Item 1 of Theorem 1.4 has the same query complexity as the optimal tester for local properties in the offline model from [14] and matches the lower bound on the query complexity of offline testing of sortedness from [24, 31]. Our tester from Item 1 of Theorem 1.4 also attains optimal resilience guarantees, i.e., the rate of erasures it can tolerate is optimal (up to a constant factor), as demonstrated by our matching hardness result for sortedness in Item 2 of Theorem 1.4.

Theorem 1.4 highlights a dramatic phase transition in the threshold regime where $t = \Theta(\epsilon)$: when $t = \omega(\epsilon)$, sortedness is not testable at all; whereas when $t = o(\epsilon)$, it is testable (and in fact, all local properties are testable) with offline-optimal query complexity!

For fixed-rate adversaries, the threshold rate is arbitrarily close to 1. The negative result is established in [40]; the positive result is stated in Proposition 6.5 and proved in Section 6.

Results for batch size $b = 2$. For batch size one, we have seen that the threshold rate for sortedness in the presence of a budget-managing adversary is $\Theta(\epsilon)$, i.e., less than one. Batches of size two result in a dramatically different picture: we can tolerate as many as $\tilde{\Omega}(n)$ erasures or corruptions between consecutive batches while maintaining optimal query complexity!

THEOREM 1.5. *There exists $c > 0$ satisfying the following. Let $\epsilon > 0$ and $n \in \mathbb{N}$. For every local property \mathcal{P} of sequences $f: [n] \rightarrow \mathbb{R}$, and every $t \leq \frac{c\epsilon^2 n}{\log^2 \epsilon n}$, there exists an ϵ -tester for \mathcal{P} with batch size $b = 2$ that works in the presence of a t -online erasure (or corruption) budget-managing adversary and makes $O\left(\frac{\log(\epsilon n)}{\epsilon}\right)$ queries. In the case of erasures, the tester has one-sided error.*

Technical overview: Pair tester for local properties. Our main technical contribution here shows that a simple and generic pair tester (see Algorithm 5 and Lemma 6.2), which queries f on pairs of the form $(x, x + 2^i) \in [n]^2$, works for all local properties of sequences in the *offline* property testing model, with query complexity of $O\left(\frac{\log(\epsilon n)}{\epsilon}\right)$. This matches known lower bounds on the query complexity of sortedness [12, 24]. Our upper bound is the same as that obtained in [14], but the new (pair) tester is simpler and has the additional feature of being resilient to online manipulations. Notably, there exist pair testers which obtain this query complexity for specific classes of local properties; see, e.g., the work of Chakrabarty, Dixit, Jha, and Seshadhri [22] on bounded derivative properties. Our work generalizes this result to all local properties of sequences. The proof proposes and analyzes a randomly-shifted variant of Ben-Eliezer’s generic tester for local properties [14], and shows that this more structured tester can be simulated by a pair tester.

We show that the pair tester is online erasure-resilient (and corruption-resilient) in a strong sense: with good probability it never queries previously manipulated elements. The analysis distinguishes between two types of erasures: ones that happen after the first element of a pair is queried, but before the second element; and all other erasures. Roughly speaking, the sharp distinction in the erasure thresholds between batch size $b = 1$ and $b = 2$

exists since the first type of erasures, which is only available to the adversary when $b = 1$, is substantially more effective than the second type in disrupting the tester.

1.2 Related Work on Erasures and Corruptions in Property Testing

Erasure-resilient testing was first investigated by Dixit et al. [27] in an offline model. In the model of Dixit et al., also studied in [15, 46, 51, 54, 57, 59], the adversary performs all erasures to the function before the execution of the algorithm.

As discussed, the online testing model was defined in [40]. In addition to the results already mentioned, [40] gave an online ϵ -tester for quadraticity of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ that has query complexity $O(\frac{1}{\epsilon})$ for constant erasure rate t . The dependence on t in the query complexity was doubly exponential, and it was left open to obtain a quadraticity tester that can deal with corruptions. In a concurrent work, Minzer and Zheng [47] improve the quadraticity tester and vastly generalize it to deal with all low-degree properties \mathcal{P}_d over general fields \mathbb{F}_q . Their tester works in the presence of t -online fixed-rate erasure adversaries and makes $O\left(\frac{1}{\epsilon} \log^{3d+3} \frac{t}{\epsilon}\right)$ queries when q is a prime and $q^{O(1)} \cdot O\left(\frac{1}{\epsilon} \log^{3d+3q} \frac{t}{\epsilon}\right)$ queries when q is a prime power.

Testing in dynamic environments. Another related line of work is on property testing in dynamic environments (see, e.g., [34, 50]), which considers settings where the tested object undergoes changes independent of the actions taken by the tester. The (adversarial) online testing model extends the study of dynamic settings to regimes where the data in the object, or the access to the data, may change continuously in response to the actions of the tester.

1.3 Connection to Maker-Breaker Games

Positional games are a central and widely investigated topic in the modern combinatorics literature; see, e.g., the standard textbooks on this topic [7, 37]. Property testing in the online erasure model is closely related to positional games, and in particular to the most prominent and well studied example of positional games: *Maker-Breaker games*. Even though Kalemaj et al. described their quadraticity tester as a game, they did not discuss the connection to the Maker-Breaker literature.

A Maker-Breaker game is defined by a finite set X of board elements and a family $\mathcal{W} \subseteq 2^X$ of winning sets. In an $(s : t)$ Maker-Breaker game, two players, called Maker and Breaker, take turns in claiming previously unclaimed elements of X . On each turn, Maker claims s board elements, whereas Breaker claims t elements. Maker wins the game if she manages to claim all elements of some winning set; otherwise, Breaker wins.

In online testing, the algorithm plays the role of Maker and the adversary is Breaker. The set X is the domain of the function, and the winning sets are witnesses, i.e., tuples of points that demonstrate that the function does not have the property. A big complication is that the tester does not know in advance which sets are in \mathcal{W} . A prerequisite for designing an online tester is being able to identify the general structure of the sets in \mathcal{W} and a winning strategy for Maker. For example, Kalemaj et al. build their tester for quadraticity by identifying a winning strategy for a game where \mathcal{W} consists of “cubes” of the form $(x, y, z, x + y, x + z, y + z, x + y + z)$. Our low-degree tester (for the special case of quadraticity) uses more intricate winning sets; see the discussion of patterns in Section 5.

Note that the original online model corresponds to $(1 : t)$ Maker-Breaker games, whereas the version with batches of size b corresponds to general $(b : t)$ Maker-Breaker games. This provides additional motivation to study batches. Going in the other direction, we hope that online testing inspires new research on Maker-Breaker games. A lot of the current literature on positional games focuses on the case where the board is a complete graph and the winning sets are graph-related (e.g., cliques). Examples from online property testing may motivate new research on Maker-Breaker games with emphasis on other types of boards, such as the hypercube.

2 Preliminaries

Notation. We use $[n]$ to represent $\{1, 2, \dots, n\}$ and \log to denote logarithm base 2. Let \mathcal{P}_d be the class of Boolean functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ of degree at most d over \mathbb{F}_2 .

2.1 Online Testing

We model access to the input with a sequence $\{O_i\}_{i \in \mathbb{N}}$ of oracles, where O_i is used to answer the i -th query (or, more generally, the i -th batch of queries). Oracle O_1 gives access to the original input (e.g., when the input is a function f , we have $O_1 \equiv f$). Subsequent oracles are objects of the same type as the input (e.g., functions with the same domain and range). Each such oracle is obtained by the adversary by modifying the previous oracle to include a growing number of erasures/corruptions as i increases. We use $\text{Dist}(O, O')$ for the Hamming distance between the two oracles (i.e., the number of queries for which they give different answers). We let $t \in \mathbb{R}_{\geq 0}$ denote the number of *erasures* (or *corruptions*) *per query* (or a batch of queries).

Definition 2.1 (Fixed-rate and budget-managing adversaries). Fix a parameter $t > 0$. A sequence³ of oracles $O = \{O_i\}_{i \in \mathbb{N}}$ is induced by a t -online fixed-rate adversary if O_1 is equal to the input and, for all $i \in \mathbb{N}$,

$$\text{Dist}(O_i, O_{i+1}) \leq \lfloor (i+1) \cdot t \rfloor - \lfloor i \cdot t \rfloor.$$

A sequence of oracles $O = \{O_i\}_{i \in \mathbb{N}}$ is induced by a t -online budget-managing adversary if O_1 is equal to the input and, for all $i \in \mathbb{N}$,

$$\text{Dist}(O_1, O_{i+1}) \leq i \cdot t.$$

Note that a budget-managing adversary has more power: an oracle sequence that can be induced by a t -online fixed-rate adversary can also be induced by a t -online budget managing adversary.

Definition 2.2 (Batch- b adversary). Fix $b \in \mathbb{N}$. A *batch- b adversary* uses oracle O_1 to answer the first b queries and, more generally, oracle O_i to answer the i -th batch of b queries. By default (if b is not specified), we assume $b = 1$.

Our complexity measure is always the total number of queries, regardless of the batch size b . Finally, we consider two types of manipulations to the input: erasures and corruptions.

Definition 2.3 (Erasure and corruption adversaries). Let \perp represent the erasure symbol. A sequence of oracles $O = \{O_i\}_{i \in \mathbb{N}}$ is induced by an *erasure adversary* if for all $i \in \mathbb{N}$ and data points x ,

$$O_{i+1}(x) \in \{O_i(x), \perp\}.$$

A *corruption adversary* can change answers to anything in the range, i.e., O_i can be any valid input for the computational task at hand.

A property \mathcal{P} denotes a set of objects (typically, a set of functions). Intuitively, it represents the set of positive instances for the testing problem. The (relative Hamming) distance between a function f and a property \mathcal{P} , denoted $\text{dist}(f, \mathcal{P})$, is the smallest fraction of function values of f that must be changed to obtain a function in \mathcal{P} . Given a proximity parameter $\epsilon \in (0, 1)$, we say that f is ϵ -far from \mathcal{P} if $\text{dist}(f, \mathcal{P}) \geq \epsilon$. An online tester is given a proximity parameter ϵ and, in addition, one or two parameters that characterize its adversary: the rate of erasures (or corruptions) t and (optionally) the batch size b .

Definition 2.4 (Online ϵ -tester). Fix $\epsilon \in (0, 1)$. An online ϵ -tester \mathcal{T} for a property \mathcal{P} that works in the presence of a specified adversary (e.g., t -online batch- b erasure budget-managing adversary) is given access to an input function f via a sequence of oracles $O = \{O_i\}_{i \in \mathbb{N}}$ induced by that type of adversary. For all adversarial strategies of the specified type,

³Our algorithms only access a finite subsequence of this sequence.

- (1) if $f \in \mathcal{P}$, then \mathcal{T} accepts with probability at least $2/3$, and
- (2) if f is ϵ -far from \mathcal{P} , then \mathcal{T} rejects with probability at least $2/3$,

where the probability is taken over the random coins of the tester. If \mathcal{T} works in the presence of an erasure (resp., corruption) adversary, we refer to it as an online-erasure-resilient (resp., online-corruption-resilient) tester.

If \mathcal{T} always accepts all functions $f \in \mathcal{P}$, then it has *1-sided error*. If \mathcal{T} chooses its queries in advance, before observing any outputs from the oracle, then it is *nonadaptive*.

Note that, in general, a tester can choose its queries adaptively (even within the same batch for a batch- b adversary), i.e., each query can depend on answers to the previous queries.

Next we state a lemma from [40] that shows that some online-erasure-resilient testers are resilient to online corruptions as well. We apply it to several of our testers.

Lemma 2.5 ([40, Lemma 1.8]). *Let \mathcal{T} be an algorithm that is given access to a function via a t -online-erasure oracle and performs a specified computational task. Suppose that for all adversarial strategies, with probability at least $\frac{2}{3}$, the algorithm \mathcal{T} outputs a correct answer and queries no erased values during its execution. Then, for the same computational task, when \mathcal{T} is given access to a function via a t -online-corruption oracle, it outputs a correct answer with probability at least $\frac{2}{3}$.*

To ease notation, we use $O(x)$ for the oracle's answer to query x (omitting the timestamp i). If x was queried multiple times, $O(x)$ denotes the first answer given by the oracle.

3 Linearity Testing

This section is dedicated to proving Theorem 1.1. We first analyze an *offline* tester which we later simulate as part of our main algorithm. Since the proof of correctness relies on Fourier analysis, we use the standard notation switch for the value returned by a Boolean function (see, e.g., [52]), where true is denoted by -1 and false is denoted by 1 . As a result, the input function is of the form $f: \{0, 1\}^n \rightarrow \{-1, 1\}$, and it is linear if $f(x_1) \cdot f(x_2) = f(x_1 \oplus x_2)$ for all $x_1, x_2 \in \{0, 1\}^n$. We sometime shorthand $y = x_1 \oplus x_2 \cdots \oplus x_k$ by $y = \bigoplus_{i \in [k]} x_i$.

3.1 An Offline Linearity Test

We first define XORTEST_k , introduced by [40], which naturally extends the BLR test.

Algorithm 1: XORTEST_k

- Input:** Even integer parameter $k \geq 2$ and query access to a function $f: \{0, 1\}^n \rightarrow \{-1, 1\}$
- 1 Query k points $x_1, \dots, x_k \in \{0, 1\}^n$ chosen uniformly at random (with replacement).
 - 2 Query point $y = \bigoplus_{i \in [k]} x_i$.
 - 3 **Reject** if $f(y) \neq \prod_{i \in [k]} f(x_i)$ (equivalently, if $f(y) \cdot \prod_{i \in [k]} f(x_i) = -1$); otherwise, **accept**.
-

Algorithm 1 always accepts all linear functions, as can be shown by induction on k . The next lemma demonstrates that XORTEST_k rejects functions that are ϵ -far from linear with sufficient probability. This is a strengthening of [40, Theorem 1.2], which bounded the rejection probability by ϵ .

Lemma 3.1. *For all $\epsilon \in (0, \frac{1}{2})$, if f is ϵ -far from linear, and $k \geq 2$ is even, then*

$$\Pr[\text{XORTEST}_k(f) \text{ rejects}] \geq \frac{1 - (1 - 2\epsilon)^{k-1}}{2} \geq \min\left\{\frac{1}{4}, \frac{k\epsilon}{2}\right\}.$$

Since $k \geq 2$ and $\epsilon \in (0, 1/2]$, we get $(1 - 2\epsilon)^{k-1} \leq 1 - 2\epsilon$. Thus, the first inequality in Lemma 3.1 implies the lower bound of ϵ on the rejection probability of XORTEST_k , the bound proved in [40]. However, as it is imperative

to use $k > 2$, the strengthened bound is crucial in obtaining an optimal online-erasure-resilient tester. In addition, the stronger guarantee we prove suffices to obtain new optimal offline linearity testers by repeatedly running Algorithm 1 and accepting iff all iterations accept. This yields an optimal tester for all values of k from 2 to $O(\frac{1}{\epsilon})$.

PROOF OF LEMMA 3.1. The key tool used in the proof is Fourier analysis (see, e.g., [52] for an overview of the technique and standard facts). We start by giving a couple of standard definitions.

The *character* functions $\chi_S: \{0, 1\}^n \rightarrow \{-1, 1\}$, defined as $\chi_S = (-1)^{\sum_{i \in S} x_i}$ for $S \subseteq [n]$, form an orthonormal basis for the space of all real-valued functions on $\{0, 1\}^n$ equipped with the inner-product $\langle g, h \rangle = \mathbb{E}_{x \sim \{0, 1\}^n} [g(x)h(x)]$, where $g, h: \{0, 1\}^n \rightarrow \mathbb{R}$. For $g: \{0, 1\}^n \rightarrow \mathbb{R}$ and $S \subseteq [n]$, the Fourier coefficient of g on S is $\widehat{g}(S) = \langle g, \chi_S \rangle = \mathbb{E}_{x \sim \{0, 1\}^n} [g(x)\chi_S(x)]$.

Now consider a function $f: \{0, 1\}^n \rightarrow \{-1, 1\}$ that is ϵ -far from linear. It is well known that the distance from f to linearity is $\frac{1}{2} - \frac{1}{2} \max_{S \subseteq [n]} \widehat{f}(S)$. Since the distance is at least ϵ , we get

$$\max_{S \subseteq [n]} \widehat{f}(S) \leq 1 - 2\epsilon. \quad (1)$$

The following equalities are shown in [40, Equation (2.4)]:

$$\begin{aligned} \Pr [\text{XORTEST}_k(f) \text{ rejects}] &= \mathbb{E}_{x_1, \dots, x_k \sim \{0, 1\}^n} \left[\frac{1}{2} - \frac{1}{2} \prod_{i \in [k]} f(x_i) \cdot f\left(\bigoplus_{i \in [k]} x_i\right) \right] \\ &= \frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} \widehat{f}(S)^{k+1}. \end{aligned} \quad (2)$$

Next, we bound the sum in (2) in terms of $\max_{S \subseteq [n]} \widehat{f}(S)$ and then apply (1):

$$\sum_{S \subseteq [n]} \widehat{f}(S)^{k+1} \leq \max_{S \subseteq [n]} \widehat{f}(S)^{k-1} \cdot \sum_{S \subseteq [n]} \widehat{f}(S)^2 = \max_{S \subseteq [n]} \widehat{f}(S)^{k-1} \leq (1 - 2\epsilon)^{k-1}.$$

Substituting this expression into (2), we obtain the first inequality in Lemma 3.1. To obtain the second inequality in Lemma 3.1, we show that $1 - (1 - 2\epsilon)^{k-1}$ is at least $1/2$ for large values of k and at least $k\epsilon$ for small values of $k \geq 2$. The bound holds trivially for $\epsilon = 1/2$, so from now on assume $\epsilon < 1/2$. Let $k_0 \in \mathbb{R}$ be such that $(1 - 2\epsilon)^{k_0-1} = 1/2$. If $k \geq k_0$, we have

$$1 - (1 - 2\epsilon)^{k-1} \geq 1 - (1 - 2\epsilon)^{k_0-1} = 1/2.$$

For the case $k \in [2, k_0)$, we prove by induction on k that $1 - (1 - 2\epsilon)^{k-1} \geq k\epsilon$. For $k = 2$, equality holds. For each $k \in [3, k_0)$, we have

$$1 - (1 - 2\epsilon)^{k-1} = (1 - (1 - 2\epsilon)^{k-2}) + 2\epsilon(1 - 2\epsilon)^{k-2} \geq (k-1)\epsilon + 2\epsilon \cdot (1 - 2\epsilon)^{k_0-1} = k\epsilon,$$

where for the inequality we bound the first term by the inductive hypothesis on $k-1$ and the second term using $k-2 \leq k_0-1$. Thus, in both cases, at least one of the expressions in the minimum is a lower bound. \square

3.2 Online-Erasure-Resilient Linearity Tester

Our algorithm (Algorithm 2) is based on multiple simulations of XORTEST_k . Each simulation starts by querying a reserve of $m = 2k$ initial points to keep many possibilities open for the final XORTEST_k query. Specifically, a reserve of $2k$ random points creates roughly 2^{2k} different options for the final XORTEST_k query.⁴ We set m to about $\log t$ to ensure that a significant fraction of options remains open before the final query of the XORTEST_k

⁴In [40], the reserve is used to simulate XORTEST_k with different values of k . Fixing the value of k to half of the reserve size eases our analysis (allowing us to use Lemma 3.1 with the same k) while providing many options.

simulation is made. There is also a small additional dependence on ϵ to overcome the fact that the overall number of iterations depends on ϵ and, as a result, for some settings of parameters, the probability of error in each iteration has to be proportional to ϵ . In principle, setting $m = \Theta(\log t + \frac{1}{\epsilon})$ and the number of iterations, r , to $\Theta(1)$ is enough for the desired query complexity, but it significantly restricts the range of parameters (e.g., it only works when $\epsilon \geq 2/n$). Our choice of parameters is more subtle: besides ensuring optimal query complexity, it enlarges the range of ϵ and t (in Theorem 1.1) for which Algorithm 2 is applicable. To do this, we set m to a smaller value: $\log t + o(\log t) + \Theta(\log(\frac{1}{\epsilon}))$, and the number of iterations, r , depends on ϵ and m . Crucially, in each

Algorithm 2: Online-Erasure-Resilient Linearity Tester

Input: Parameters $\epsilon \in (0, 1/2]$, $t \in \mathbb{N}$; query access to f via t -erasure oracle sequence \mathcal{O}

```

1  $t \leftarrow \max\{t, 2\}$ ; // If  $t < 2$ , replace it with  $t = 2$ .
2  $m \leftarrow 4 \lceil \frac{1}{4}(14 + \log t + \log \log^2 t + \log \frac{1}{\epsilon^2}) \rceil$ ,  $\alpha \leftarrow \min\{\frac{1}{4}, \frac{m\epsilon}{4}\}$  and,  $r \leftarrow \lfloor \frac{5}{4\alpha} \rfloor$ .
3 repeat  $r$  times
4   Sample  $X = (x_1, \dots, x_m) \in (\{0, 1\}^n)^m$  uniformly at random.
5   Query  $f$  at points  $x_1, \dots, x_m$ .
6   Query  $f$  at point  $y = \bigoplus_{j \in S} x_j$ , where  $S$  is a uniformly random subset of  $[m]$  of size  $\frac{m}{2}$ .
7   if  $\mathcal{O}(y) \cdot \prod_{j \in S} \mathcal{O}(x_j) = -1$  then
8     Reject; // Equality implies all points were non-erased when queried.
9 Accept

```

iteration, the distribution over the queries made by Algorithm 2 is identical to that in XORTEST_k test with $k = \frac{m}{2}$. Indeed, since X and S (sampled in Steps 4 and 6 of Algorithm 2, respectively) are independent, we can imagine sampling S before X . For each choice of S , the marginal distribution of $(x_j)_{j \in S}$ is uniform over $(\{0, 1\}^n)^k$. Finally, the last query is $y = \bigoplus_{j \in S} x_j$, resulting in the same distribution on the $k + 1$ queries as in XORTEST_k .

The second ingredient we need is the following lemma.

Lemma 3.2. *For all $m \geq 10$, the probability that one specific iteration of the loop in Line 3 of Algorithm 2 queries an erased point is at most $\frac{3\alpha}{25}$, where α is as defined in Step 2 of Algorithm 2.*

We next restate and prove the main theorem of this section, deferring the proof of Lemma 3.2 to the next section.

THEOREM 1.1 (OPTIMAL ONLINE LINEARITY TESTER). *For all $n \in \mathbb{N}$, $\epsilon \in (0, 1/2]$, and t satisfying $t \log^2 t \leq 2^{-21} \cdot \epsilon^{2.5} 2^{n/2}$, there exists an ϵ -tester for linearity of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ that works in the presence of a t -online erasure (or corruption) budget-managing adversary and makes $O(\max\{\frac{1}{\epsilon}, \log t\})$ queries. In the case of erasure adversary, the tester has 1-sided error.*

PROOF. We first focus on erasures, and show that Algorithm 2 satisfies the conditions of the theorem. Clearly, it always accepts all linear functions. Now, fix an adversarial (budget-managing) strategy and suppose that the input function is ϵ -far from linear. By Lemma 3.1, since Algorithm 2 is executed with even $k = \frac{m}{2}$, the probability that one iteration of the loop in Step 3 samples a witness of nonlinearity is at least $\alpha = \min\{1/4, m\epsilon/4\}$.

By Lemma 3.2, the probability that an erasure is seen in a specific iteration is at most $\frac{3\alpha}{25}$. By a union bound, the probability of a single iteration seeing an erasure or not selecting a witness of nonlinearity is at most $1 - \alpha + \frac{3\alpha}{25} = 1 - \frac{22\alpha}{25}$. Algorithm 2 errs only if this occurs in all iterations. By independence of random choices in

different iterations, the failure probability is at most

$$\left(1 - \frac{22\alpha}{25}\right)^r \leq \left(1 - \frac{22\alpha}{25}\right)^{\frac{5}{4\alpha}} \leq e^{-1.1} \leq \frac{1}{3},$$

where we used that $r = \lceil \frac{5}{4\alpha} \rceil \geq \frac{5}{4\alpha}$ and $1 - x \leq e^{-x}$ for all x . The query complexity is at most

$$r(m+1) \leq \left(\frac{5}{4\alpha} + 1\right)(m+1) \leq \frac{4m}{\alpha} = \max\left\{16m, \frac{16}{\epsilon}\right\} \leq \max\left\{640 \cdot \log t, \frac{640}{\epsilon}\right\},$$

where the last inequality is due to the fact that $m \leq 20(\log t + \frac{1}{\epsilon})$.

We next show that Algorithm 2 is also corruption-resilient. For the soundness, our analysis holds since it suffices to have one iteration that finds a witness without seeing manipulations. For completeness, note that the algorithm can only err if it has seen a manipulation, and the probability of seeing a manipulation at any iteration is at most $3\alpha/25$ by Lemma 3.2. Using a union bound, the overall probability of seeing any manipulated entry during the entire execution is at most

$$\frac{3\alpha}{25} \cdot r \leq \frac{3\alpha}{25} \left(\frac{5}{4\alpha} + 1\right) = \frac{3}{20} + \frac{3\alpha}{25} \leq \frac{1}{3}. \quad \square$$

Remark. Our tester is applicable for all $t \leq \text{poly}(\epsilon) \cdot 2^{n/2}$. For large t , i.e., $t = \Omega(\epsilon 2^n)$, it can be easily shown that online testing is impossible. This follows from the fact that there exists some constant c such that $\frac{c}{\epsilon}$ queries are not enough to distinguish between linear and nonlinear functions, so by the time the tester makes $\frac{c}{\epsilon}$ queries, the adversary can already erase or corrupt the rest of the input. Subsequent work [3] determined the values of t for which linearity can be tested in the online setting. Analogous questions for other properties are investigated in Section 6.

3.3 Probability of Seeing an Erasure

Recall that Lemma 3.1 shows that, assuming f is ϵ -far, the probability of spotting a witness in a single iteration is at least $\alpha = \min\{\frac{1}{4}, \frac{m\epsilon}{4}\}$. In this section, we prove the probability of querying an erasure is at most $3\alpha/25$ in every iteration, as stated in Lemma 3.2. We start with an auxiliary claim, similar to an argument that appears in the proof of [40, Lemma 2.8].

Claim 3.3. Fix an iteration of the loop in Step 3 of Algorithm 2. For all $T \subseteq [m]$, let $y_T = \bigoplus_{j \in T} x_j$. Then, the probability there exist two subsets $T_1 \neq T_2$ of the set $[m]$ with $y_{T_1} = y_{T_2}$ is small:

$$\Pr_X [\exists T_1 \neq T_2 \text{ such that } y_{T_1} = y_{T_2}] \leq \frac{\alpha}{25}.$$

PROOF. We start by noting that m defined in Step 2 of Algorithm 2 equals $4 \lceil v/4 \rceil$ for some value v . This simply rounds v upwards to the next multiple of four, yielding the bound $m < v + 4$. Using this inequality and then the premise $t \cdot \log^2 t \leq 2^{-21} \cdot \epsilon^{2.5} 2^{n/2}$ from Theorem 1.1, we get

$$m < 18 + \log t + 2 \log \log t + 2 \log \frac{1}{\epsilon} \quad (3)$$

$$= \log \left(\frac{2^{18} t \log^2 t}{\epsilon^2} \right) \leq \log \left(\frac{2^{-3} \epsilon^{2.5} 2^{n/2}}{\epsilon^2} \right) \leq \log \left(\sqrt{\epsilon \cdot 2^n / 50} \right) = \frac{\log(\epsilon \cdot 2^n / 50)}{2}. \quad (4)$$

Consider two distinct sets $T_1, T_2 \subset [m]$. W.l.o.g. there exists an element $\ell \in T_1 \setminus T_2$. Fix all entries in X besides x_ℓ . The value of y_{T_2} is now fixed, but over the random choice of $x_\ell \in \{0, 1\}^n$, the vector y_{T_1} is uniform over $\{0, 1\}^n$. Thus, $\Pr_{x_\ell} [y_{T_1} = y_{T_2}] = 2^{-n}$ and, consequently,

$$\Pr_X [y_{T_1} = y_{T_2}] = \mathbb{E} \left[\Pr_{x_\ell} [y_{T_1} = y_{T_2}] \right] = \mathbb{E} [2^{-n}] = 2^{-n},$$

where both expectations are over all entries in X besides x_ℓ , which are drawn independently from x_ℓ . We use a union bound over all pairs of subsets T_1 and T_2 , and then apply (4) to get

$$\Pr_X [\exists T_1 \neq T_2 \text{ such that } y_{T_1} = y_{T_2}] \leq \frac{2^{2m}}{2^n} \leq \frac{\epsilon}{50} \leq \frac{\alpha}{25}.$$

The last inequality holds since $\epsilon \leq \min\{1/2, m\epsilon/2\} = 2\alpha$ for all $m \geq 2$. \square

We now upper bound the probability of seeing an erasure in one iteration.

PROOF OF LEMMA 3.2. The total number of erasures performed during the execution of the algorithm is at most $tr(m+1)$. We define three bad events and give upper bounds on their probabilities.

An erasure while querying X . Let B_1 be the event that $O(x_j) = \perp$ for some point x_j sampled in this iteration, where $j \in [m]$. Each point x_j is sampled uniformly from $\{0, 1\}^n$, so the probability it is erased is at most $tr(m+1)/2^n$. By a union bound over all m points, recalling $n \geq 2m$, we get

$$\Pr_X[B_1] \leq \frac{tr(m+1)m}{2^n} \leq \frac{tr(m+1)m}{2^{2m}}. \quad (5)$$

X induces a bad distribution of y points. Let B_2 be the event that, in this iteration, there exist two different choices of S leading to the same final query $y \in \{0, 1\}^n$. By Claim 3.3, it holds that $\Pr[B_2] \leq \frac{\alpha}{25}$.

An erasure on query y . Let B_3 be the event that $O(y) = \perp$ for y queried in this iteration. The adversary knows X before y is queried, but there are plenty of choices for y . Conditioned on $\overline{B_2}$, the distribution of y is uniform over $\binom{m}{m/2}$ different choices. We use $\binom{m}{m/2} \geq \frac{2^m}{\sqrt{2m}}$ to obtain

$$\Pr[B_3|\overline{B_2}] \leq \frac{tr(m+1)}{\binom{m}{m/2}} \leq \frac{tr(m+1)\sqrt{2m}}{2^m}. \quad (6)$$

In terms of the bad events, our goal is to show that $\Pr[B_1 \cup B_3] \leq \frac{3\alpha}{25}$. By using a union bound over B_1 and B_3 and then the law of total probability to compute $\Pr[B_3]$, we get

$$\begin{aligned} \Pr[B_1 \cup B_3] &\leq \Pr[B_1] + \Pr[B_3] \\ &= \Pr[B_1] + \Pr[B_3|\overline{B_2}] \cdot \Pr[\overline{B_2}] + \Pr[B_3|B_2] \cdot \Pr[B_2] \\ &\leq \underbrace{\Pr[B_1] + \Pr[B_3|\overline{B_2}]}_{(\star)} + \Pr[B_2]. \end{aligned}$$

Since $\Pr[B_2] \leq \frac{\alpha}{25}$, to complete the proof of Lemma 3.2, it remains to show that the expression denoted by (\star) is at most $\frac{2\alpha}{25}$. To do this, we combine the bounds from (5) and (6):

$$\Pr[B_1] + \Pr[B_3|\overline{B_2}] \leq \frac{tr(m+1)m}{2^{2m}} + \frac{tr(m+1)\sqrt{2m}}{2^m} \leq \frac{trm^2}{2^{m+1}} \leq \frac{rm^2\epsilon^2}{2^{15}\log^2 t}, \quad (7)$$

where the second inequality holds because $\frac{(m+1)m}{2^m} + (m+1)\sqrt{2m} \leq \frac{m^2}{2}$ for all $m \geq 10$, and the last inequality holds because the value of m specified in Step 2 of Algorithm 2 satisfies $2^m \geq \frac{2^{14}t\log^2 t}{\epsilon^2}$. To further bound the final expression in (7), we split the analysis into two cases, depending on the value of $m\epsilon$. Recall $\alpha = \min\{\frac{1}{4}, \frac{m\epsilon}{4}\}$, as defined in Step 2 of Algorithm 2.

Case I ($m\epsilon \leq 1$): In this case $\alpha = \frac{m\epsilon}{4}$. By the setting of parameters in Step 2 of Algorithm 2, the number of iterations is $r = \lceil \frac{5}{m\epsilon} \rceil \leq \frac{5}{m\epsilon} + 1$, which means that $rm\epsilon \leq 5 + m\epsilon \leq 6$. Using $\log t \geq 1$, the expression denoted by (★) is at most

$$\frac{(rm\epsilon)m\epsilon}{2^{15} \log^2 t} \leq \frac{6m\epsilon}{2^{15} \log^2 t} \leq \frac{m\epsilon}{50} = \frac{2\alpha}{25}.$$

Case II ($m\epsilon > 1$): In this case $\alpha = \frac{1}{4}$. We start by using (3):

$$\begin{aligned} m &< 18 + \log t + 2 \log \log t + 2 \log \frac{1}{\epsilon} \\ &\leq 18 + 2.1 \log t + \frac{1.1}{\epsilon} \leq 11.2 \cdot \max \left\{ 2 \log t, \frac{1}{\epsilon} \right\}, \end{aligned}$$

where the second inequality applies $2 \log y \leq 1.1y$ for all $y > 0$ (first with $y = \frac{1}{\epsilon}$ and then with $y = \log t$), and the last inequality uses $2 \log t \geq 2$ and $\frac{1}{\epsilon} \geq 2$. It follows that

$$\frac{m\epsilon}{2 \log t} \leq 11.2 \frac{\max\{2 \log t, \frac{1}{\epsilon}\}}{2 \log t \cdot \frac{1}{\epsilon}} = \frac{11.2}{\min\{2 \log t, \frac{1}{\epsilon}\}} \leq 5.7.$$

Finally, we use $r = \lceil \frac{5}{4\alpha} \rceil = 5$ and (7) to obtain that the expression denoted by (★) is at most

$$\frac{r}{2^{13}} \left(\frac{m\epsilon}{2 \log t} \right)^2 \leq \frac{5 \cdot (5.7)^2}{2^{13}} \leq \frac{1}{50} = \frac{2\alpha}{25}.$$

This completes the proof of Lemma 3.2. □

4 The Lower Bound for Low-Degree Testing

In this section, we prove our lower bound for online low-degree testing.

THEOREM 1.2 (LOWER BOUND FOR LOW-DEGREE TESTING). *Fix an integer $d > 1$ and $\epsilon \in (0, 1/2)$. Let \mathcal{P}_d be the set of polynomials of degree at most d over \mathbb{F}_2 . There exists $n_0 = n_0(d, \epsilon)$, such that for all $n \geq n_0$, every ϵ -tester of functions $f: \{0, 1\}^n \rightarrow \{0, 1\}$ for property \mathcal{P}_d that works in the presence of a t -online fixed-rate erasure adversary must make $\Omega\left(\binom{\log t}{d}\right)$ queries.*

PROOF. Fix a degree $d > 1$. Let $\binom{n}{\leq d}$ denote $\sum_{i=0}^d \binom{n}{i}$. For a vector $x \in \mathbb{F}_2^n$, define its extension $\text{Ext}_d(x) \in \mathbb{F}_2^{\binom{n}{\leq d}}$ to be the vector where each entry is indexed by a set $S \subseteq [n]$ of at most d coordinates and $\text{Ext}_d(x)_S = \prod_{i \in S} x_i$. In other words, each entry of $\text{Ext}_d(x)$ is an evaluation of one monomial of degree at most d on input x . In particular, the extended vector $\text{Ext}_d(x)$ includes all entries x_i of x (as entries indexed by singletons $S = \{i\}$). We define projection π as the inverse of Ext_d , i.e., $\pi(\text{Ext}_d(x)) = x$.

Consider the following strategy \mathcal{S} for a fixed-rate erasure adversary. Suppose the tester has successfully obtained answers to k distinct queries y_1, \dots, y_k . Consider the linear space in $\mathbb{F}_2^{\binom{n}{\leq d}}$ spanned by $\text{Ext}_d(y_1), \dots, \text{Ext}_d(y_k)$, and let $Z \subseteq \mathbb{F}_2^{\binom{n}{\leq d}}$ denote set of projections of its vectors back to \mathbb{F}_2^n using π . Note that $y_i \in Z$ for $i \in [k]$. The adversary simply tries to erase all (non-erased, non-queried) points of Z . Observe that Z is determined by the queries made by the tester and does not depend on the input function f . We next show that $|Z|$ is small relative to k , and so unless k is large enough, the adversary can entirely erase Z .

Claim 4.1. *Let $r = |Z|$. Then $\binom{\lceil \log r \rceil}{d} \leq k$.*

PROOF. Consider the matrix A with rows indexed by polynomials of degree at most d and columns indexed by vectors $z \in Z$, where the entry indexed by a polynomial p and a point z contains the value $p(z) \in \mathbb{F}_2$. To prove the claim, we give two bounds on the rank of A .

Upper bound: $rk(A) \leq k$. Let A' be the submatrix of A with rows indexed by monomials. Since every degree- d polynomial is a linear combination of monomials, we have $rk(A) = rk(A')$. In A' , the column indexed by z is exactly the extended vector $\text{Ext}_d(z)$. By definition, every column in A' is spanned by the k columns indexed by y_1, \dots, y_k , showing that $rk(A') \leq k$.

Lower bound: $rk(A) \geq \binom{\lceil \log r \rceil}{d}$. Let B be an arbitrary submatrix of A with only $r' \leq r$ columns, where $r' = 2^a$ is the largest power of 2 not exceeding r . Trivially, $rk(A) \geq rk(B)$. We note the rows of B are of the form $(p(z_1), \dots, p(z_{r'}))$, with a row for every polynomial p of degree at most d , and r' entries for each of the r' inputs. We then apply [13, Lemma 1.4] (or [45, Theorem 1.5]) which directly lower bounds the dimension of the rows of B , and hence $rk(B)$, by

$$\binom{a}{\leq d} \geq \binom{a}{d} = \binom{\lceil \log r \rceil}{d}.$$

The two inequalities together yield the claim. \square

Now we apply Yao's minimax principle for the online model [40, Corollary 9.4]. It states that to prove a lower bound q on the worst-case query complexity of online-erasure-resilient property testing, it suffices to give an adversarial strategy \mathcal{S} , a distribution \mathcal{D}^+ on positive instances, and a distribution \mathcal{D}^- on instances that are negative with probability at least $\frac{\epsilon}{7}$, such that every deterministic q -query algorithm that accesses its input via an oracle using strategy \mathcal{S} sees the same distribution on the the query-answer histories under \mathcal{D}^+ and \mathcal{D}^- .

Let \mathcal{D}^+ be the uniform distribution over all degree d Boolean functions on $\{0, 1\}^n$ and \mathcal{D}^- be the uniform distribution over all Boolean functions on $\{0, 1\}^n$.

We show that a function $f \sim \mathcal{D}^-$ is ϵ -far from \mathcal{P}_d (set of functions of degree at most d) with probability at least $6/7$. Let $g \in \mathcal{P}_d, f \sim \mathcal{D}^-$, and $\text{dist}(f, g)$ be the fraction of domain points on which f and g differ. Then, $\mathbb{E}[\text{dist}(f, g)] = 1/2$. By the Hoeffding bound, $\Pr_{f \sim \mathcal{D}^-}[\text{dist}(f, g) \leq \epsilon] \leq 2^{-\Omega_\epsilon(2^n)}$. By a union bound over the $2^{\binom{n}{\leq d}}$ functions of degree at most d , we get $\Pr_{f \sim \mathcal{D}^-}[\text{dist}(f, \mathcal{P}_d) \geq \epsilon] \geq 1 - 2^{\binom{n}{\leq d}} \cdot 2^{-\Omega_\epsilon(2^n)}$. For large enough n , this probability is at least $6/7$.

Let A be a deterministic algorithm that makes $q \leq \binom{\lceil \log t \rceil - 1}{d}$ queries to the oracle \mathcal{O} with adversarial strategy \mathcal{S} . By Claim 4.1, we get $r \leq t$, i.e., after each query y_i for $i \in [q]$, the adversary has sufficient erasure budget to erase Z . We argue that the distributions on the histories of query answers are the same under the distributions \mathcal{D}^+ and \mathcal{D}^- . Under both distributions, the adversary erases the same query points. Consider a query y_k of A that is not erased and suppose it is made after A successfully obtained answers a_1, \dots, a_{k-1} to distinct queries y_1, \dots, y_{k-1} . When $f \sim \mathcal{D}^-$, we have $\Pr_{f \sim \mathcal{D}^-}[f(y_k) = a_k \mid f(y_1) = a_1 \wedge \dots \wedge f(y_{k-1}) = a_{k-1}] = 1/2$ for all $a_1, \dots, a_k \in \{0, 1\}$, since the value of $f(y_k)$ is set uniformly and independently of values at other points. Now consider $f \sim \mathcal{D}^+$, viewed as a uniformly random coefficient vector $\mathbb{1}_f \in \{0, 1\}^{\binom{n}{\leq d}}$, where each entry corresponds to a monomial of degree at most d . For any query it holds that $f(y_i) = \langle \mathbb{1}_f, \text{Ext}_d(y_i) \rangle$. By definition of \mathcal{S} , the extension $\text{Ext}_d(y_k)$ is linearly independent of $\text{Ext}_d(y_1), \dots, \text{Ext}_d(y_{k-1})$, which implies that for any fixed values b_1, \dots, b_{k-1} of $\langle \mathbb{1}_f, \text{Ext}_d(y_i) \rangle, \dots, \langle \mathbb{1}_f, \text{Ext}_d(y_{k-1}) \rangle$, the value of $\langle \mathbb{1}_f, \text{Ext}_d(y_k) \rangle$ is still uniformly distributed.

Thus, \mathcal{D}^+ generates degree- d polynomials, \mathcal{D}^- generates functions that are ϵ -far from \mathcal{P}_d with probability at least $\frac{\epsilon}{7}$, and the query-answer histories for any deterministic algorithm A that makes $q \leq \binom{\lceil \log t \rceil - 1}{d}$ queries and runs against the t -online fixed-rate erasure oracle employing strategy \mathcal{S} are identical under \mathcal{D}^+ and \mathcal{D}^- . Consequently, Yao's principle implies the desired lower bound. \square

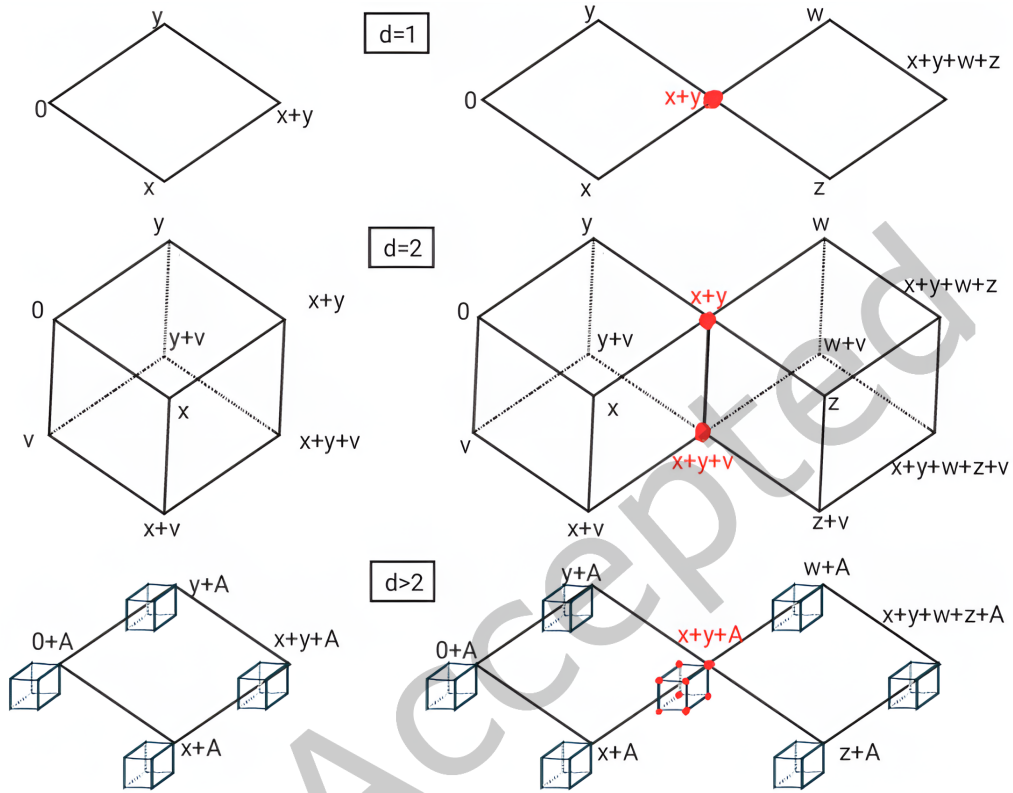


Fig. 1. Chains of cubes (with 0 as the first parameter). The left side shows linear subspaces (drawn as cubes); the right side shows chains formed by two cubes. Red bold points overlap and cancel each other. From top to bottom, we have witnesses for linearity (chains of squares), witnesses for quadraticity (each vertical edge represents a batch of $b = 2$ points), and witnesses for \mathcal{P}_d , where each batch queries an affine subspace with a fixed linear component A and different translations.

5 Low-Degree Testing

This section is dedicated to the proof of Theorem 1.3, which gives an online tester for the property \mathcal{P}_d (being a polynomial of degree at most d) with batches of size $b = 2^{d-1}$.

To generalize the strategy employed for linearity in Section 3.2, we visualize XORTEST_2 as a linear square $(0, x, y, x + y)$, with 0 denoting the origin, and view the extended test XORTEST_{2k} as a *chain of k squares* (see the top part of Figure 1). For quadraticity and all degrees $d > 2$, we generalize the visualization to a *chain of cubes* (see the bottom part of Figure 1).

5.1 Algebraic Testing Patterns

We define a *testing pattern* to be a mapping from a set of parameters to a structured set of points (e.g., mapping parameters x, y to the three points $x, y, x + y$). Fixing values of all parameters induces an *instance* of the testing pattern. To test a low-degree property \mathcal{P}_d , we sample a random instance of some “good” pattern and check the parity of the sum of function values on all points in this instance. This is a generalization of the tester of [1],

which uses the specific pattern of a linear subspace, mapping parameters x_1, \dots, x_{d+1} to the set of points spanned by them over \mathbb{F}_2 , namely the set $\{\sum_{i \in S} x_i\}_{S \subseteq [d+1]}$.

Our notion of a good testing pattern is a special case of the more abstract notion of *k-local formal characterizations* by [44], a fact that will be used later on. We give our definition below and soon after clarify this relation.

Definition 5.1 (Testing pattern, parameter, instance). A *testing pattern* is a matrix⁵

$$X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$$

with rows r_1, \dots, r_ℓ , such that $\ell \geq m$, each row is unique, and $\text{rank}(X) = m$.⁶ An *instance* of a pattern X is described by the product $XM \in \mathcal{M}_{\ell \times n}(\mathbb{F}_2)$ for some parameter matrix $M \in \mathcal{M}_{m \times n}(\mathbb{F}_2)$ with rows $a_1, \dots, a_m \in \mathbb{F}_2^n$. The rows of XM specify ℓ testing points y_1, \dots, y_ℓ in \mathbb{F}_2^n . We often write $X_{\ell \times m}$ or $M_{m \times n}$ to specify the dimensions of X or M as matrices.

Generalizing the notation from [1], for a function f and a pattern instance XM , we denote the sum (over \mathbb{F}_2) of the values of f on these points by

$$T_{X,f}(a_1, \dots, a_m) := \sum_{i \in [\ell]} f(y_i).$$

Each testing pattern induces a parity-checking tester, described in Algorithm 3.

Algorithm 3: An X -tester

Input: a pattern $X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ and query access to a function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$

- 1 Choose a parameter matrix $M \in \mathcal{M}_{m \times n}(\mathbb{F}_2)$ uniformly at random (alternatively, choose m row vectors $a_1, \dots, a_m \in \mathbb{F}_2^n$ independently and uniformly at random).
 - 2 Query f at testing points y_1, \dots, y_ℓ , the ℓ row vectors of the instance matrix XM .
 - 3 **Accept** if $T_{X,f}(a_1, \dots, a_m) = 0$; otherwise, **reject**.
-

Next, we define a good testing pattern for property \mathcal{P} to have two desired properties.

Definition 5.2 (Good patterns). A pattern $X_{\ell \times m}$ is *complete* for a property \mathcal{P} if for all functions $f \in \mathcal{P}$,

$$\Pr_{a_1, \dots, a_m} [T_{X,f}(a_1, \dots, a_m) = 1] = 0.$$

A pattern $X_{\ell \times m}$ is *minimally sound* for a property \mathcal{P} if for all functions $f \notin \mathcal{P}$,

$$\Pr_{a_1, \dots, a_m} [T_{X,f}(a_1, \dots, a_m) = 1] > 0.$$

If a pattern is both complete and minimally sound for \mathcal{P} , we call it *good* for \mathcal{P} .

In other words, a good pattern $X_{\ell \times m}$ for a property \mathcal{P} characterizes \mathcal{P} :

$$f \in \mathcal{P} \iff \forall a_1, \dots, a_m : T_{X,f}(a_1, \dots, a_m) = 0. \quad (8)$$

⁵We use $\mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ for the set of $\ell \times m$ matrices with entries in \mathbb{F}_2 .

⁶The rows of X correspond

Relation to ℓ -local characterizations. Our notion of good testing patterns is a special case of ℓ -local-characterizations, defined in the seminal work of [44]. Here we review their definitions and relate them to ours.

Definition 5.3 (*ℓ -local formal characterizations* [44, Definition 2.3]). Fix a field \mathbb{F} of cardinality q , where q is a prime power, and a field \mathbb{K} of cardinality q^c for $c \in \mathbb{N}$. A family of functions $\mathcal{F} \subseteq \{\mathbb{K}^n \rightarrow \mathbb{F}\}$ has an ℓ -local formal characterizations if there exists an integer m ; ℓ linear functions $L_1, \dots, L_\ell : \mathbb{K}^m \rightarrow \mathbb{K}$; and a linear subspace $V \subset \mathbb{F}^\ell$ such that $f \in \mathcal{F}$ if and only if for every $a_1, \dots, a_m \in \mathbb{K}^n$, we have $\langle f(y_1), \dots, f(y_\ell) \rangle \in V$, where $y_i = L_i(a_1, \dots, a_m)$. (Here, the linear functions L_i are interpreted as maps $(\mathbb{K}^n)^m \rightarrow \mathbb{K}^n$ in the natural way.)

Definition 5.4 (*2-ary independent characterization* [44, Definition 2.5]). An ℓ -local formal characterization $(L_1, \dots, L_\ell; V)$ is 2-ary independent if L_1 and L_j are linearly independent for every $j \in \{2, \dots, \ell\}$.

Observation 5.5. A good pattern $X_{\ell \times m}$ for property \mathcal{P} (Definition 5.2), is a special case of an ℓ -local characterization for the function family \mathcal{P} . Additionally, if no row of the pattern $X_{\ell \times m}$ is the zero vector, then the local characterization corresponding to it is 2-ary independent

Indeed, we use $\mathbb{K} = \mathbb{F} = \mathbb{F}_2$. The pattern matrix $X_{\ell \times m}$ describes ℓ linear functions, $L_i : \mathbb{F}_2^m \rightarrow \mathbb{F}_2$ where each row lays out the coefficients of a certain function, and the free coefficient is always 0. The instance matrix M describes m parameters $a_i \in \mathbb{F}_2^n$, and the ℓ testing points are $y_i = L_i(a_1, \dots, a_m)$. Finally, we choose the subspace $V \subseteq \mathbb{F}^\ell$ to be $V = \{(y_1, \dots, y_\ell) : \sum_{i \in [\ell]} y_i = 0\}$. Equation (8) now coincides with the characterization requirement of an ℓ -local characterization. The additional part holds since each row in $X_{\ell \times m}$ is unique (by definition), and since each two non-zero vectors over \mathbb{F}_2 are linearly independent. That is, each of the corresponding functions L_2, \dots, L_ℓ is linearly independent from L_1 , and the corresponding local characterization is indeed 2-ary independent.

5.2 Generalized Witnesses for \mathcal{P}_d

We first review the ideas used for linearity in Section 3, and interpret them in a way that can similarly generalize the standard test of [1, 20] for the property \mathcal{P}_d . To ease the discussion, we shall replace linearity with affinity (the property \mathcal{P}_1), for which the suitable test is XORT_{2k+1} , with an odd parameter (and an even number of points). The standard test samples 3 parameters s_0, x_1, x_2 and queries the 4 points $(s_0, x_1, x_2, s_0 + x_1 + x_2)$, which we view as a “square”, as illustrated (with $s_0 = 0$) in the top left part of Figure 1. Consider a second square, with new parameters x_3, x_4 and the parameter $s_1 = s_0 + x_1 + x_2$, so it can be “glued” to the first square at s_1 , which cancels out, as illustrated in the top right of Figure 1 (with s_1 in bold red). This leaves the points s_0, x_1, x_2, x_3, x_4 , and their sum $s_2 = s_0 + \sum_{j \in [4]} x_j$. To create a chain of k squares, use points x_1, \dots, x_{2k} and partial sums $s_i := s_0 + \sum_{j \in [2i]} x_j$ for $i \in [k]$. The square i (for $i \in [k]$) consists of the three points $s_{i-1}, x_{2i-1}, x_{2i}$ and their sum s_i , and throughout the entire chain the partial sums s_1, \dots, s_{k-1} cancel out, leaving the testing points $s_0, x_1, x_2, \dots, x_{2k}$ and their sum s_k . This is equivalent to XORT_{2k+1} , and fully captures the idea used for linearity in Section 3.

Remark. *When gluing two squares, we use one intersection point, as two would simply lead to a new square after cancellations. For example, gluing the squares $(x, y, z, x + y + z)$ and $(x, y, w, x + y + w)$ would leave the four points $(w, z, x + y + w, x + y + z)$. Writing $v = x + y + w$ yields $x + y + z = v + w + z$ (since we are working over \mathbb{F}_2^n), revealing the square $(v, w, z, v + w + z)$.*

For higher degrees, the well-known testers are a random linear subspace [1] and a random affine subspace [20]. A pattern for \mathcal{P}_d corresponding to the latter would have directional parameters a_1, \dots, a_{d+1} and an affine shift parameter a_{d+2} with the resulting testing points forming the affine subspace $\{a_{d+2} + \sum_{i \in S} a_i\}_{S \subseteq [d+1]}$ (and a random choice of parameters leads to a random affine subspace). As before, we can “glue” together two such subspaces that intersect on a $(d-1)$ -dimensional subspace, as shown in the lower right parts of Figure 1 (for \mathcal{P}_1 , the intersection was a single point). Repeating this process creates a pattern we visualize as a “chain of cubes”.

Generalizing to higher dimensions, it is easier to formally describe the pattern for degree $d + 1$ by replacing each point of XORTEST_{2k+1} (for affinity) with an entire d -dimensional cube (e.g., for affinity we change nothing as a 0-dimensional cube is simply a point). This is apparent in both Figure 1, and in the definition below.

Definition 5.6 (Chain of Cubes Pattern). For degree $d + 1$ and odd integer $s \geq 3$, we define the *chain of cubes pattern* $\chi_{d,s}$. It takes parameters a_1, \dots, a_d for the cube structure (when $d = 0$, there are no cube parameters), and a_{d+1}, \dots, a_{d+s} for the chain, and outputs $(s + 1) \cdot 2^d$ testing points, each is a combination of a subset of the first d parameters, and either all or exactly one of the last s parameters.

Formally, $\chi_{d,s}$ has “cube columns” $1, \dots, d$ and “chain columns” $d + 1, \dots, d + s$. We index each row with $i \in \{0, 1\}^d$ and $j \in [s + 1]$. In row (i, j) , the first d coordinates are exactly the vector i . The last s coordinates depend on j , where if $j \in [s]$ then only coordinate $d + j$ has value 1, and if $j = s + 1$ then all s chain coordinates have value 1.

To show that $\chi_{d,s}$ is a good pattern for the property \mathcal{P}_{d+1} , we show it satisfies an equivalent condition, stated next.

Claim 5.7. A pattern $X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ is good for \mathcal{P}_d if and only if the two conditions hold:

- (i) For all coordinate sets $S \subseteq [m]$ of size $|S| \leq d$, we have $\sum_{i \in [\ell]} \prod_{j \in S} X_{ij} = 0$.
- (ii) There exists a set $S \subseteq [m]$ of size $|S| = d + 1$ such that $\sum_{i \in [\ell]} \prod_{j \in S} X_{ij} = 1$.

The claim defines which patterns are good for \mathcal{P}_d , in terms of entries of the matrix X . The rough intuition here is that a function $f_{\text{YES}} \in \mathcal{P}_d$ is a summation of degree d monomials, and these cancel out due to the first bullet, but for any $f_{\text{NO}} \notin \mathcal{P}_d$, we can find an instance M^* for which the tester fails. The choice of M^* uses the existence of a monomial of degree at least $d + 1$ in f_{NO} . The formal proof generalizes that of [1, Lemma 1], where they show that a linear $(d + 1)$ -dimensional subspace is a good pattern for \mathcal{P}_d . It is rather technical and, therefore, is deferred to Section B.

Using Claim 5.7, we show that the chain of cubes pattern is good for our purposes:

Claim 5.8. The chain of cubes pattern $\chi_{d,s}$ is good for \mathcal{P}_{d+1} for all $d \in \mathbb{N}$ and all odd $s \geq 3$.

PROOF. We say a row is *full* on a set of coordinates $S \subseteq [d + s]$ if all coordinates in S are 1 in this row. We show the pattern $\chi_{d,s}$ satisfies the conditions of Claim 5.7. That is, each set S of at most $d + 1$ coordinates has an even number of full rows on S , and there exists a set S' of $d + 2$ coordinates for which the number of full rows on S' is odd.

Let S be a set of at most $d + 1$ coordinates. We break into cases according to the number of cube coordinates (the first d columns) in S . If $|S \cap [d]| < d$ then there exists a coordinate $c \in [d] \setminus S$ and each full row on S appears once with 1 in coordinate c and once with 0 in coordinate c , and the total number of full rows on S is even. If $|S \cap [d]| = d$ and $S = [d]$, then the full rows on S are exactly those where all cube coordinates are 1, those of index $(\vec{1}, i)$ where $i \in [s + 1]$, of which there are $s + 1$ rows, an even number. The last case is where $|S \cap [d]| = d$ but $|S| = d + 1$, so S contains exactly one chain coordinate $(d+c)$, where $c \in [s]$. Here, there are exactly two full rows on S , indexed with $(\vec{1}, c)$ and $(\vec{1}, s + 1)$.

To show the pattern is minimally sound, consider the set $S = [d + 2]$ (this is well defined since $s \geq 3$). A full row on S has 1 in all cube coordinates, and 1 in the first two chain coordinates. This occurs in exactly one row, indexed by $(\vec{1}, s + 1)$, completing the proof. \square

5.3 Soundness of Patterns

In this section, we show a soundness result for any good pattern of \mathcal{P}_d , and in particular for our chain of cubes pattern, $\chi_{d-1,s}$. We start by applying a theorem by [44] for 2-ary independent local characterizations (Definition 5.4), stated below. We then improve the soundness guarantee of good patterns by mimicking a specialized argument made in [20] for affine subspaces. This last argument helps when ϵ is small, so that most violating instances are such due to a single point. We note that the argument relies on two properties held by all good patterns. First, each two induced testing points, over a random instance, are independent. Second, the tester's decision is according to the parity of all answers to the queried points.

The following is a corollary of [44, Theorem 2.9] for the property \mathcal{P}_d over \mathbb{F}_2 :

Corollary 5.9 (Corollary of [44, Theorem 2.9]). *Fix a function f that is ϵ -far from \mathcal{P}_d , and a 2-ary independent ℓ -local characterization for \mathcal{P}_d . Then for the test corresponding to the characterization, it holds that*

$$\Pr [\text{test rejects when run on } f] \geq \min \{\epsilon/2, \gamma(\ell)\}, \quad (9)$$

where $\gamma(\ell) = 1/((2\ell + 1)(\ell - 1))$ only depends on ℓ .

We are now ready to prove the soundness of good patterns.

Lemma 5.10. *Fix any good pattern $X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ with $\ell \geq 3$ and where no row in X is the 0 vector. For any function f that is ϵ -far from \mathcal{P}_d , we have*

$$\Pr_{a_1, \dots, a_m} [T_{X,f}(a_1, \dots, a_m) = 1] \geq \min \left\{ \frac{\ell\epsilon}{2}, \frac{1}{2\ell^2} \right\}.$$

PROOF. Recall that by Observation 5.5, any good pattern $X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ for \mathcal{P}_d where no row is the 0 vector is a 2-ary independent ℓ -local characterization of \mathcal{P}_d . We can therefore apply Corollary 5.9 above for the corresponding test, and specifying the event of rejection, we have

$$\Pr_{a_1, \dots, a_m} [T_{X,f}(a_1, \dots, a_m) = 1] \geq \min \{\epsilon/2, \gamma(\ell)\}, \quad (10)$$

where $\gamma(\ell) = 1/((2\ell + 1)(\ell - 1))$ only depends on ℓ .

Next, we improve the soundness guarantee for small values of ϵ , the following argument generalizes the one in the proof of [20, Lemma 8]. Let g be a degree- d function that achieves minimal distance, ϵ , from f . Our probability space is the random choice of parameters a_1, \dots, a_m . We observe two types of events for each $i \in [\ell]$. The first event is E_i - the event that $f(y_i) \neq g(y_i)$. The second is F_i - the event that $f(y_i) \neq g(y_i)$ but $f(y_j) = g(y_j)$ for all $j \neq i$.

Since each y_i is uniform over \mathbb{F}_2^m , it is clear that $\Pr [E_i] = \epsilon$. Due to the fact that each two rows are different (and hence, over \mathbb{F}_2 , linearly independent), we get pairwise independence for the variables y_1, \dots, y_ℓ . I.e, for any $i \neq j$ we have $\Pr [E_i \wedge E_j] = \epsilon^2$. It is clear that for all $i \in [\ell]$:

$$\Pr [F_i] \geq \Pr [E_i] - \sum_{j \neq i} \Pr [E_i \wedge E_j] \geq \epsilon - \ell\epsilon^2.$$

Due to the completeness of the test, and since $g \in \mathcal{P}_d$, the parity-check passes on g for every instance. Note that if f differs from g on *exactly one* point in a specific instance, the parity-check on f fails on this instance. This event is exactly the disjoint union of all events F_i , and its probability is lower bounded as follows

$$\Pr \left[\bigsqcup_{i \in [\ell]} F_i \right] = \sum_{i \in [\ell]} \Pr [F_i] \geq \ell (\epsilon - \ell\epsilon^2) = \ell\epsilon(1 - \ell\epsilon).$$

Thus, for any ϵ -far function f , we get

$$\Pr_{a_1, \dots, a_m} [T_{X,f}(a_1, \dots, a_m) = 1] \geq \ell\epsilon(1 - \ell\epsilon).$$

We finish up by combining both arguments. If $\ell\epsilon \leq \frac{1}{2}$, we are guaranteed soundness of at least $\ell\epsilon(1 - \ell\epsilon) \geq \frac{\ell\epsilon}{2}$. Otherwise, $\ell\epsilon > \frac{1}{2} \geq \frac{1}{\ell}$, which implies $\frac{\epsilon}{2} > \frac{1}{2\ell^2}$. Combined with $\gamma(\ell) \geq \frac{1}{2\ell^2}$, the soundness guaranteed by (10) must be at least $\frac{1}{2\ell^2}$. \square

5.4 Algorithm with Batches of 2^{d-1} Queries

Our low-degree tester is based on multiple simulations of the parity-check tester with the pattern $\chi_{d-1,m}$, but each simulation uses additional queries to overcome the adversary, mirroring the simulation of XORTEST_k in Algorithm 2. The tester is described in Algorithm 4, where the number of iterations and the length of the chain, m , are chosen according to the soundness guarantee in Lemma 5.10 for the generalized patterns $\chi_{d-1,m}$.⁷

Algorithm 4: Online-Erasure-Resilient Degree- d Tester

Input: Parameters $\epsilon \in (0, 1/2]$ and $d, t \in \mathbb{N}$; query access to f via t -online erasure oracle sequence \mathcal{O} with batch size 2^{d-1}

- 1 $t \leftarrow \max\{t, 2\}$; // If $t < 2$, replace it with $t = 2$.
- 2 $m \leftarrow 2 \left\lceil \log \left(\frac{2^{20} 2^d t^{1/4} (d \log t)^{3/2}}{\epsilon^{1/2}} \right) \right\rceil + 1$, $\ell \leftarrow (m+1)2^{d-1}$, $\alpha \leftarrow \min\{\frac{\epsilon\ell}{2}, \frac{1}{2\ell^2}\}$, $r \leftarrow \lceil \frac{2}{\alpha} \rceil$.
- 3 **repeat** r **times**
- 4 Sample $V = (v_1, \dots, v_{d-1}) \in (\mathbb{F}_2^n)^{d-1}$ uniformly at random.
- 5 Sample $X = (x_1, \dots, x_{2m}) \in (\mathbb{F}_2^n)^{2m}$ uniformly at random.
- 6 **for each** $i \in [2m]$: batch query f at points $x_i + \sum_{j \in T} v_j$ for all $T \subseteq [d-1]$.
- 7 Sample uniformly at random a subset of $[2m]$ of size m , denoted $S = \{s_1, \dots, s_m\}$.
- 8 Set $y = \sum_{j \in S} x_j$ and batch query f at points $y + \sum_{j \in T} v_j$ for all $T \subseteq [d-1]$.
- 9 **if** $T_{\chi_{d-1,m}, \mathcal{O}}(v_1, \dots, v_{d-1}, x_{s_1}, \dots, x_{s_m}) = 1$ **then**
- 10 | **Reject**; // Equality implies all points were non-erased when queried.
- 11 **Accept**

5.5 Probability of Seeing an Erasure

To analyze the probability of seeing an erasure, we use the structured (“cube”) part of our patterns, which is evident in our tester as well. This is done, in essence, by considering the subspace spanned by v_1, \dots, v_{d-1} from 4, denoted by $A = \text{sp}\{v_1, \dots, v_{d-1}\}$, and observing the quotient group \mathbb{F}_2^n/A consisting of elements of the form $x + A$ (each corresponds to 2^{d-1} points in \mathbb{F}_2^n). Projecting all our queries and erasures to this group, results with a similar setting to the one used to prove the resilience of our linearity tester.

More formally, assume $\dim(A) = d - 1$ (if it is smaller, it only works to our benefit), and let $U = (u_d, \dots, u_n)$ complete the set of vectors V to a basis of the entire space. Define the space spanned by these vectors $B = \text{sp}\{u_d, \dots, u_n\}$, and let π_B be a projections of any vector $x \in \mathbb{F}_2^n$ to B using its unique representation over the basis $V \cup U$. The image of the projection is B , which is isomorphic to \mathbb{F}_2^{n-d+1} , and for each $u \in B$, its pre-image is exactly $\pi_B^{-1}(u) = u + A = \{u + v : v \in A\}$.

⁷Improving the soundness guarantee in Lemma 5.10 to $\Theta(\min\{\ell\epsilon, 1\})$ would allow us to choose smaller values for parameters m and r , resulting in a better query complexity.

We can therefore recast a single iteration of Algorithm 4 in terms of B alone. Projection of a uniformly random point $x_i \in \mathbb{F}_2^n$ is a uniformly random point $u_i \in B$, with $x_i + A = u_i + A$ (which defines the same batch of queries). Furthermore, after choosing the random subset of indices S , we get

$$\pi_B(y) = \pi_B\left(\sum_{j \in S} x_j\right) = \sum_{j \in S} \pi_B(x_j) \in B,$$

for which we again query $y + A = \pi_B(y) + A$. It is now clear that an iteration of Algorithm 4 reduced to its queries on B is equivalent to an iteration of Algorithm 2 on an entire space of dimension $n' = n - d + 1$ (which B is isomorphic to). Lastly, we strengthen our adversary, by saying each time it tries to erase a point $x \in \mathbb{F}_2^n$, the entire set $x + A$ is erased instead, and we think of it as an erasure of $\pi_B(x)$ in B . Thus, the probability of seeing an erasure, even with the strengthened adversary, is that of Algorithm 2 seeing an erasure.

Applying the same arguments as in Claim 3.3, with $n' = n - d + 1$, the adjusted number of batches, $2m$, and adjusted premise on t leads to the following claim.

Claim 5.11. *Fix an iteration of the loop in Step 3 of Algorithm 4, and denote $u_i = \pi_B(x_i)$. For all $T \subseteq [2m]$, let $y_T = \sum_{j \in T} u_j$. Then, the probability there exist two subsets $T_1 \neq T_2$ of the set $[2m]$ with $y_{T_1} = y_{T_2}$ is small:*

$$\Pr_X \left[\exists T_1 \neq T_2 \text{ such that } y_{T_1} = y_{T_2} \right] \leq \frac{2^{4m}}{2^{n'}} \leq \frac{\epsilon}{2^{20} 2^{2d} (d \log t)^2},$$

as long as $t \log^7 t \leq c \cdot \epsilon^{2.5} d^{-7} 2^{(n-11d)/2}$ for some constant c .

Finally, we can obtain the following lemma. We omit its proof which is identical to the proof of Lemma 3.2, using $n' \geq 4m$ (instead of $n \geq 2m$ there).

Lemma 5.12. *For all $m \geq 5$, and t as above, the probability that one specific iteration of the loop in Line 3 of Algorithm 4 queries an erased point is at most*

$$\frac{4trm^2}{2^{2m+1}} + \frac{\epsilon}{2^{20} 2^{2d} (d \log t)^2}.$$

5.6 Analysis of the Online Tester

Now we complete the analysis of Algorithm 4 and prove Theorem 1.3 for erasure adversaries. Similar to the case of linearity, our tester has a small constant probability of ever seeing a manipulated entry, and hence it is robust against corruptions as well.

THEOREM 1.3. *There exists a constant $c > 0$ such that for all $n, d \in \mathbb{N}$, $\epsilon \in (0, 1/2)$, and t satisfying $d < \frac{n}{11}$ and $t \log^7 t \leq c \cdot \epsilon^{2.5} d^{-7} 2^{(n-11d)/2}$, there exists an ϵ -tester for property \mathcal{P}_d of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ that works in the presence of a $(2^{d-1}, t)$ -online erasure budget-managing adversary and makes $O\left(\frac{1}{\epsilon} + 2^{3d} (d + \log t)^3\right)$ queries.*

PROOF FOR THE CASE OF ERASURES. We show that Algorithm 4 satisfies the conditions of the theorem. It always accepts all degree- d functions. Now, fix an adversarial (budget-managing) strategy and suppose that the input function is ϵ -far from any degree d function.

By Lemma 5.10 with $\ell = (m + 1)2^{d-1}$ points in the $\chi_{d-1, m}$ pattern, the probability that one iteration of the loop in Step 3 samples a witness for not being degree d is $\alpha = \min\left\{\frac{\ell\epsilon}{2}, \frac{1}{2\ell^2}\right\}$. We split the analysis into two cases, depending on which term achieves the minimum.

Case I: $\alpha = \ell\epsilon/2$, the number of iterations is $r = \left\lceil \frac{2}{\alpha} \right\rceil = \left\lceil \frac{4}{\ell\epsilon} \right\rceil$ which means that $r\ell\epsilon \leq \left(\frac{4}{\ell\epsilon} + 1\right)\ell\epsilon = 4 + \ell\epsilon \leq 5$. By Lemma 5.12, the probability that an erasure is seen within a single iteration is at most

$$\frac{4trm^2}{2^{2m+1}} + \frac{\epsilon}{2^{20} 2^{2d} (d \log t)^2}.$$

We bound the second term naively by $\frac{\epsilon}{100}$. For the first term, we use the setting of m in the algorithm which implies $2^{2m} \geq \frac{2^{20}t}{\epsilon^2}$, and the fact that $m \leq \ell$ to get

$$\frac{4rm^2\epsilon^2}{2^{20}} \leq \frac{4(rm\epsilon)m\epsilon}{2^{20}} \leq \frac{20m\epsilon}{2^{20}} \leq \frac{m\epsilon}{2^{15}}.$$

The probability that an erasure is seen in a specific iteration is therefore at most

$$\frac{m\epsilon}{2^{15}} + \frac{\epsilon}{100} < \frac{\ell\epsilon}{80}.$$

By a union bound, the probability of a single iteration seeing an erasure or not selecting a witness is at most $1 - \frac{\ell\epsilon}{2} + \frac{\ell\epsilon}{80} \leq 1 - \frac{\ell\epsilon}{3}$. Algorithm 4 errs only if this occurs in all iterations. By independence of random choices in different iterations, the failure probability is at most

$$\left(1 - \frac{\ell\epsilon}{3}\right)^r \leq \left(1 - \frac{\ell\epsilon}{3}\right)^{\frac{4}{\ell\epsilon}} \leq e^{-\frac{4}{3}} \leq \frac{1}{3},$$

where we used $1 - x \leq e^{-x}$ for all x .

The total number of queries used in this case is at most $r(2m+1)b \leq 2r\ell \leq 10/\epsilon$.

Case II: $\alpha = \frac{1}{2\ell^2}$, and $r = 4\ell^2$. We first bound m , set in Algorithm 4, in terms of d and $\log t$.

$$\begin{aligned} m &\leq 41 + 2d + \log t + 3 \log d + 3 \log \log t + \log\left(\frac{1}{\epsilon}\right) \\ &\leq 41 + 4 \log t + 8d + 3 \log m \\ &\leq 20(\log t + d) + 3 \log m, \end{aligned}$$

The second inequality follows from the fact that, in this case, $\log(\frac{1}{\epsilon}) \leq 3 \log \ell \leq 3(\log m + d)$. Since $m \geq 2^5$ we have $3 \log m \leq m/2$ which implies (for $t, d \geq 2$)

$$m \leq 40(\log t + d) \leq 2^6 d \log t.$$

To bound the probability that an erasure is seen in one iteration, we start, like in Case I, with Lemma 5.12. We bound the first term recalling that $r = 4\ell^2 \leq 4m^2 2^{2d}$,

$$\frac{4trm^2}{2^{2m+1}} \leq \frac{8tm^4 2^{2d}}{2^{2m}} \leq \frac{2^3 \cdot 2^{2d} m^4 \epsilon^2}{2^{80} 2^{4d} (d \log t)^6} \leq \frac{1}{2^{40} m^2 2^{2d}} \leq \frac{1}{20\ell^2},$$

where the second inequality plugs in the value of m set in the algorithm to bound $t/2^{2m}$, and the third uses $m \leq 40d \log t$ in the denominator. Adding the second term, we get at most

$$\frac{4trm^2}{2^{2m+1}} + \frac{\epsilon}{2^{20} 2^{2d} (d \log t)^2} \leq \frac{1}{20\ell^2} + \frac{1}{20\ell^2} \leq \frac{1}{10\ell^2}.$$

By a union bound, the probability of a single iteration seeing an erasure or not selecting a witness is at most $1 - \frac{1}{2\ell^2} + \frac{1}{10\ell^2} \leq 1 - \frac{1}{3\ell^2}$. As before, we use the independence of different iterations to conclude the algorithm errs with probability at most $(1 - \frac{1}{3\ell^2})^{4\ell^2} \leq \frac{1}{3}$.

The total number of queries used is $r(2m+1)b \leq 6m^3 2^{3d} \leq 240 \cdot 2^{3d} (d + \log t)^3$. \square

6 Online Erasure-Resilient Testing of Local Sequential Properties

In this section, we establish our results on testing local properties of sequences discussed in Section 1.1.3. That is, we prove the following two main results, for batch size one and two, respectively.

THEOREM 1.4 (TESTING LOCAL PROPERTIES IN PRESENCE OF BUDGET-MANAGING ADVERSARY WITH BATCH SIZE 1). *There exist absolute constants $0 < c < C$ satisfying the following. For all $\epsilon \in (0, 1)$ and $n \geq C/\epsilon$:*

- (1) *For every local property \mathcal{P} of sequences $f: [n] \rightarrow \mathbb{R}$, and every $t \leq c\epsilon$, there exists an ϵ -tester for \mathcal{P} that works in the presence of a t -online erasure budget-managing adversary and makes $O(\frac{\log(\epsilon n)}{\epsilon})$ queries.*
- (2) *For every $t \geq C\epsilon$, there is no ϵ -tester for sortedness of sequences $f: [n] \rightarrow \mathbb{R}$ that works in the presence of a t -online erasure budget-managing adversary (with any number of queries).*

Both of these results hold (with the same parameters) if erasures are replaced with corruptions. In the case of the erasure adversary, the tester has one-sided error.

THEOREM 1.5. *There exists $c > 0$ satisfying the following. Let $\epsilon > 0$ and $n \in \mathbb{N}$. For every local property \mathcal{P} of sequences $f: [n] \rightarrow \mathbb{R}$, and every $t \leq \frac{c\epsilon^2 n}{\log^2 \epsilon n}$, there exists an ϵ -tester for \mathcal{P} with batch size $b = 2$ that works in the presence of a t -online erasure (or corruption) budget-managing adversary and makes $O(\frac{\log(\epsilon n)}{\epsilon})$ queries. In the case of erasures, the tester has one-sided error.*

Our core technical result in this section shows that a simple pair tester (presented in Algorithm 5) can test all local properties of sequences.

For $d, n \in \mathbb{N}$, where $d < n$, we define the set of all distance- d pairs in $[n]$ by⁸

$$D_d(n) = \{(x, y) \in [n]^2 : y - x \equiv d \pmod{n}\}.$$

We also define the *interval* $[a : b]$ as the set of all integers $\{i \in \mathbb{Z} : a \leq i \leq b\}$. The *length* of the interval $[a : b]$ is $b - a + 1$. Our algorithm relies on a fundamental notion of *unrepairability* in a sequence f with respect to a property \mathcal{P} . The notion we use is a slightly generalized version of a notion of unrepairability for intervals originally proposed in [14].

Definition 6.1 (Unrepairability). Let \mathcal{P} be a local property of sequences $f: [n] \rightarrow \mathbb{R}$ characterized by the forbidden family \mathcal{F} . For a subset $S \subseteq [n]$, a sequence f is *unrepairable* on S with respect to \mathcal{P} if the following holds: every sequence $f': [n] \rightarrow \mathbb{R}$, where $f'(x) = f(x)$ for all $x \in S$, does not satisfy \mathcal{P} . (I.e., every f' that agrees with f on all entries in S contains a pattern from \mathcal{F} .)

Crucially, given \mathcal{P} and query access to f , in order to know whether f is unrepairable on S w.r.t. \mathcal{P} , one needs to query f only on the elements of S . For our purposes, $|S| \leq 2$ always holds, and so we only need to make two queries in order to check unrepairability.

Note that, up to some edge cases, an interval which satisfies the notion of unrepairability from [14] is called a *witness interval* in this paper (see Definition 6.12 below); the notion of a witness interval is used in the analysis, but not in the algorithm.

We first show that the pair tester works for every local property \mathcal{P} in the *offline* property testing model. The statement is given in the following lemma and proved in Section 6.2.

Lemma 6.2 (Offline correctness of pair tester). *Algorithm 5 is a nonadaptive ϵ -tester with one-sided error probability bounded by $\frac{1}{7}$ for every local property \mathcal{P} in the offline testing model (without erasures or corruptions).*

The pair tester not only works in the offline setting, but it is also unlikely to see erasures in the online setting when t is appropriately small. The next two lemmas state results of this type for batch sizes 1 and 2, respectively.

⁸Note that the definition is cyclic in the sense that it allows for pairs where x is close to n , and y is much smaller and close to 1. The cyclic nature of the definition slightly simplifies the algorithm and analysis.

Algorithm 5: Pair tester for local properties

Input: local property \mathcal{P} , $\epsilon \in (0, 1)$; query access to $f : [n] \rightarrow \mathbb{R}$ (in offline model)
or, for $t \in \mathbb{R}$ and $b \in \mathbb{N}$, via t -online erasure oracle sequence \mathcal{O} with batch size b

- 1 Set $\ell = \lfloor \log \left(\frac{\epsilon n}{4} \right) \rfloor$.
- 2 **repeat** $\frac{200 \log(\epsilon n)}{\epsilon}$ **times**
- 3 Sample $i \in [0 : \ell]$ uniformly at random,
- 4 Sample $(x, y) \in D_{2^i}(n)$ uniformly at random; **query** $f(x)$ and $f(y)$.
- 5 **if** f is unrepairable on $\{x, y\}$ **then**
- 6 **Reject**
- 7 **Accept**

Lemma 6.3 (Pair tester, batch size 1). *Fix a local property \mathcal{P} , $\epsilon > 0$, $n \in \mathbb{N}$, and $t \leq \epsilon/10^7$. The probability that Algorithm 5, when run in the online erasure model with parameters \mathcal{P} , ϵ , n , t and batch size $b = 1$, queries an erased element is at most $1/10$.*

Lemma 6.4 (Pair tester, batch size 2). *Fix a local property \mathcal{P} , $\epsilon > 0$, $n \in \mathbb{N}$, and $t \leq \frac{\epsilon^2 n}{2 \cdot 10^6 \log^2(\epsilon n)}$. The probability that Algorithm 5, when run in the online erasure model with parameters \mathcal{P} , ϵ , n , t and batch size $b = 2$, queries an erased element is at most $1/10$.*

Note the sharp contrast between the threshold rate for $b = 1$ ($t = \Theta(\epsilon)$) and $b = 2$ ($t = \tilde{\Theta}(n)$).

The positive results of Theorems 1.4 and 1.5 follow from Lemmas 6.2, 6.3, 6.4. The proof of the negative (impossibility) result of Theorem 1.4 is given in Section 6.3.

We start with the proof of Theorem 1.4, concerning batch size one.

PROOF OF THEOREM 1.4, PART 1. For an erasure adversary, we use Lemmas 6.2 and 6.3. Consider a local property \mathcal{P} and a sequence f that is ϵ -far from \mathcal{P} . Let A be the event that the tester rejects, given query access to f (not to the erasure oracle). Let B be the event that the tester encounters at least one erasure. The rejection probability of the tester in the online setting is at least

$$\Pr[A \setminus B] \geq \Pr[A] - \Pr[B] \geq \frac{6}{7} - \frac{1}{10} > \frac{2}{3},$$

where the second inequality follows from Lemmas 6.2 and 6.3.

For a corruption adversary, we use the same argument combined with Lemma 2.5. \square

Next we provide the corresponding proof for batch size two.

PROOF OF THEOREM 1.5. The proof (for both erasures and corruptions) is identical to the above proof of Theorem 1.4, Part 1, except that we use Lemma 6.4 instead of Lemma 6.3. \square

We next state our main result for *fixed-rate* adversaries, showing that the threshold rate is arbitrarily close to 1. The proof appears in Section 6.1. A matching negative result is established in [40], see Theorems 1.5 and 1.7 there.

Proposition 6.5. *Fix $\epsilon \in (0, 1)$, $t < 1$ and a local property \mathcal{P} of sequences $f : [n] \rightarrow \mathbb{R}$, where $n \geq \frac{C(\log \frac{1}{\epsilon} + \log \frac{1}{1-t})^2}{\epsilon^2(1-t)}$ for a large enough constant $C > 0$. There exists a (one-sided error) ϵ -tester for \mathcal{P} that works in the presence of a t -online erasure fixed-rate adversary and has query complexity $O\left(\frac{\log(\epsilon n)}{\epsilon(1-t)}\right)$. For a corruption adversary, the same is true without the one-sided error guarantee.*

6.1 Online Erasure-Resilience of Pair Tester

We next prove that the pair tester is unlikely to see an erased element during the course of the algorithm, completing the proofs of Lemma 6.3, Lemma 6.4, and Proposition 6.5. In the following definition, we separate the erasures into two types.

Definition 6.6 (Blind and relative erasures). Consider a single loop iteration of Algorithm 5. An erasure made before the first query in this iteration is called *blind* (with respect to this iteration). An erasure made after the first query but before the second query of the iteration is called *relative*.

Relative erasures can be much more effective than blind ones in making the tester query an erased element. For relative erasures, observe that once the first query, x , in a given iteration is made, the second query can take one of roughly $\log(\epsilon n)$ options, and its probability to encounter a relative erasure made during the iteration is of order $1/\log(\epsilon n)$. In contrast, Lemma 6.7 shows that the probability of any future query to see a specific blind erasure is much smaller: $O(1/n)$.

Lemma 6.7 (Pair tester: marginal probability). Consider a fixed iteration of the loop in Algorithm 5 with parameters $n, \mathcal{P}, \epsilon, t$. Fix $z \in [n]$. The probability that z is queried in this iteration is $\frac{2}{n}$.

PROOF. Fix the value of i sampled in Line 3 of Algorithm 5. Out of the n pairs in $D_{2^i}(n)$, exactly two contain z , so the probability of z to be queried is exactly $\frac{2}{n}$. \square

This sharp contrast between relative and blind erasures is the reason for the huge separation between the batch-1 case and the batch-2 case. In the former case, both relative and blind erasures are available to the adversary, whereas in the latter case, only blind erasures are available.

The following simple corollary bounds the probability that the pair tester queries at least one blind erasure in the course of the algorithm.

Corollary 6.8. Consider one run of Algorithm 5. Let q be the number of queries made by the tester, and let $T = tq$ denote the erasure budget for the run. The probability that the tester encounters a blind erasure is at most $\frac{2Tq}{n}$.

PROOF. Consider a query x made by the tester. Let S be the set of blind erasures (with respect to the current iteration) made so far. The randomness used by Algorithm 5 to choose x is independent of the set S (by the structure of the tester and since S consists only of blind erasures). By Lemma 6.7, for each query, $\Pr[x \in S] \leq \frac{2|S|}{n} \leq \frac{2T}{n}$. The corollary follows by a union bound over all queries. \square

We now prove that our testers are unlikely to encounter erasures for both batch sizes, starting with batch size one (Lemma 6.3) followed by the corresponding result for batch size two (Lemma 6.4).

PROOF OF LEMMA 6.3. The number of queries made by Algorithm 5 is $q = \frac{400 \log(\epsilon n)}{\epsilon}$. The total number of erasures is at most $T = tq \leq \frac{4}{10^5} \log(\epsilon n)$. By Corollary 6.8, the probability that the algorithm queries a blind erasure at least once during the course of its run is bounded by

$$\frac{2Tq}{n} \leq \frac{3200 \cdot \log^2 \epsilon n}{10^4 \cdot \epsilon n} \leq \frac{3200}{10^5} \cdot \frac{6}{5} < \frac{4}{100},$$

where the last inequality uses the bound $\frac{\log^2(z)}{z} \leq 6/5$ for $z \geq 1$ (plugging in $z = \epsilon n$ and assuming $\epsilon \geq \frac{1}{n}$, as otherwise, the problem becomes a decision problem and requires reading the entire input in the worst case.).

Next we bound the probability that Algorithm 5 encounters a relative erasure. The a priori probability that a viable pair (x, y) (that is, any pair $(x, y) \in D_{2^i}(n)$ for some choice of $i \in [0 : \ell]$) is queried in a given iteration is

exactly $\frac{1}{n^{(\ell+1)}}$, where ℓ is as in the algorithm. Note that this probability does not depend on i . By Bayes' rule, conditioned on the first element being x , the probability of each of the elements in the set

$$\{y \in [n] : \exists i \in [0 : \ell] \text{ such that } (x, y) \in D_{2^i}(n)\}$$

to be the second query of the iteration is exactly $\frac{1}{\ell+1} \leq \frac{2}{\log(\epsilon n)}$; all other values of y have probability zero to be queried. Therefore, for any specific relative erasure conducted by the adversary, the probability that the second query of the relevant iteration hits the erasure is at most $\frac{2}{\log(\epsilon n)}$. By a union bound (noting that, by definition, each erasure is relative with respect to at most one iteration of the algorithm, and is blind with respect to previous iterations), the probability that the algorithm queries a relative erasure throughout its run is at most $\frac{2}{\log(\epsilon n)} \cdot T \leq \frac{8}{10^5}$. The proof follows by combining the bounds for blind and relative erasures. \square

PROOF OF LEMMA 6.4. Set $C = 1/(2 \cdot 10^6)$. In the batch size 2 case there are no relative erasures, since each loop iteration in the algorithm is captured by a single batch. That is, all erasures are blind. We apply Corollary 6.8 with

$$q = \frac{200 \log(\epsilon n)}{\epsilon} \quad \text{and} \quad T = tq \leq \frac{C\epsilon^2 n}{\log^2(\epsilon n)} \cdot \frac{400 \log(\epsilon n)}{\epsilon} = \frac{400C\epsilon n}{\log(\epsilon n)}$$

to bound the probability to query at least one (blind) erasure during the run of the algorithm by

$$\frac{2Tq}{n} \leq \frac{400C\epsilon}{\log(\epsilon n)} \cdot \frac{400 \log(\epsilon n)}{\epsilon} \leq 16 \cdot 10^4 C < \frac{1}{10},$$

as desired. \square

We next prove the main proposition regarding fixed-rate adversaries.

PROOF OF PROPOSITION 6.5. Our tester applies the pair tester (Algorithm 5) as a black box, but only at pre-specified time stamps along the way (in which the fixed-rate adversary is known not to make erasures). Let $m = O(\epsilon^{-1} \log(\epsilon n))$ denote the number of iterations in Algorithm 5. By definition of the fixed-rate model (Definition 2.1), when the rate is $t < 1$, there exists a sequence of integers $i_1 < i_2 < \dots < i_m$, which satisfies the following properties.

- For each $j \in [m]$, the adversary cannot make an erasure after the query at time i_j is conducted.
- For each $j \in [m-1]$, we have $i_{j+1} - i_j \leq \frac{2}{1-t}$.

Our tester in this case simulates Algorithm 5: for each $j = 1, \dots, m$ we run iteration j of the latter at timestamps i_j and $i_j + 1$. Note that there is no erasure between them (so all erasures are blind). Let $q \leq 2m \cdot \frac{2}{1-t} = O\left(\frac{\log(\epsilon n)}{\epsilon(1-t)}\right)$ be the total number of queries in our case. By Lemma 6.7, the probability that the query at each time i_j or $i_j + 1$ encounters an erased element is at most $2q/n$. By a union bound, the probability that we observe at least one erasure along the way is bounded by

$$\frac{2q}{n} \cdot m \leq \frac{1}{100}, \tag{11}$$

provided that C is large enough. Now, disregarding erasures, by Lemma 6.2 if f is ϵ -far from \mathcal{P} then the probability that the tester queries a violation to \mathcal{P} is at least $6/7$. Combined with (11), the proof follows for the online erasures setting. The case of online corruptions immediately follows from Lemma 2.5. \square

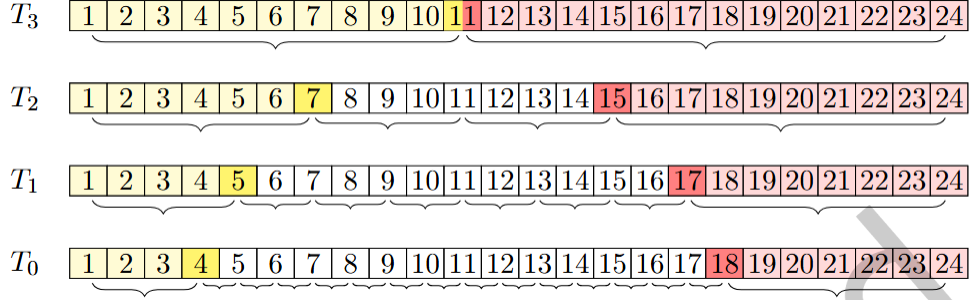


Fig. 2. An illustration of Definition 6.9 with parameters $n = 24, a = 3, w = 16, \ell = 3$. That is, the above is the 3-shifted $(3, 16)$ -hierarchical partition of the array $[1 : 24]$, where indices are written in each cell for convenience. The shaded intervals in yellow and red are respectively the left and right extremal intervals defined in Definition 6.10, with dark shade marking the endpoints.

6.2 Pair Tester for Local Properties via Shifted Interval Partitioning

The main purpose of this section is to prove Lemma 6.2, establishing the correctness of the pair tester in the offline setting. We first present and analyze Algorithm 6, a randomly shifted version of the structured tester from [14] for local properties. We then show that the pair tester can simulate the shifted structured tester.

The next two definitions present a hierarchical partition structure of intervals (with some overlaps between endpoints of intervals), used in Algorithm 6.

Definition 6.9 (Shifted hierarchical partition). Let $a, w \in [n]$ and let $\ell \geq 0$ be an integer, where w is divisible by 2^ℓ and $a + w \leq n$. An a -shifted hierarchical partition of width w with ℓ layers in $[n]$, or a -shifted (ℓ, w) -hierarchical partition in short, is a tuple (T_0, \dots, T_ℓ) where for each $i \in [0 : \ell]$, the i -th layer T_i is a set of intervals defined by

$$T_i = \{[a + j \cdot 2^i : a + (j + 1) \cdot 2^i] \mid j \in [w/2^i - 2]\} \cup \{[1 : a + 2^i], [a + w - 2^i : n]\}.$$

In other words, each layer T_i consists of intervals of length $2^i + 1$, except the first interval (starting at 1) and the last interval (ending at n), which may be longer: the first interval is of length $2^i + a$, and the last interval is of length $2^i + 1 + n - a - w$. The intervals in each layer T_i intersect only at their endpoints.

The first and last intervals in each layer T_i play a special role in our analysis.

Definition 6.10 (Extremal intervals). Let T be an a -shifted (ℓ, w) -hierarchical partition of $[n]$. An interval $I = [x : y] \in T_i$ is said to be *left-extremal* if $x = 1$ and *right-extremal* if $y = n$. In either of these cases, I is considered *extremal*.

The *query-pair* of an interval I with respect to T and i , denoted $Q_{T,i}(I)$, is defined as $\{x, y\}$ if I is not extremal, $\{a, a + 2^i\}$ if I is left-extremal, and $\{a + w - 2^i, a + w\}$ if I is right-extremal.

Equipped with a shifted hierarchical partition structure, we now present the algorithm.

The following lemma establishes that Algorithm 6 is an ϵ -tester for local properties in the standard *offline* model of property testing.

Lemma 6.11. Let $n \in \mathbb{N}$ and \mathcal{P} be a local property of sequences $f : [n] \rightarrow \mathbb{R}$. Algorithm 6 is an ϵ -tester for \mathcal{P} with one-sided error and success probability $6/7$ in the offline model (with no erasures/corruptions).

For the proof, we define the following notion of a witness interval. Roughly speaking, an interval of f within $[n]$ is a witness with respect to a property \mathcal{P} , if the value of f on the endpoints of the interval determine that f

Algorithm 6: Shifted hierarchical structured tester for local properties in offline model

Input: local property \mathcal{P} , $\epsilon \in (0, 1)$; query access to $f : [n] \rightarrow \mathbb{R}$ (in offline model)

- 1 Set $\ell = \lfloor \log(\frac{\epsilon n}{4}) \rfloor$.
- 2 Let $w \in \mathbb{N}$ be the maximum integer divisible by 2^ℓ and satisfying $w \leq n - \epsilon n/4$.
- 3 **repeat** $\frac{20 \log(\epsilon n)}{\epsilon}$ **times**
- 4 Sample $a \in [\lceil \epsilon n/4 \rceil]$ uniformly at random.
- 5 Let $T = (T_0, \dots, T_\ell)$ be the a -shifted (ℓ, w) -hierarchical partition of $[n]$.
- 6 Sample a uniformly random $i \in [0 : \ell]$ and a uniformly random interval $[x, y] \in T_i$.
- 7 **Query** $f(x')$ and $f(y')$, where $\{x', y'\} = Q_{T,i}([x, y])$.
- 8 **if** f *is unrepairable on* $\{x', y'\}$ **then**
- 9 **Reject**
- 10 **Accept**

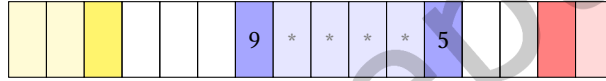


Fig. 3. Three types of witness intervals, with the endpoint or endpoints of each interval bolded. Standard witness intervals, such as the middle one in blue, have both their left and right endpoint strictly in the interior. Left and right extremal witnesses (in yellow and red, respectively) have only one endpoint. The value of f on the endpoints of the interval, and only on them, determines whether an interval is a witness or not. For example, for the property of sortedness, the interval in blue violates the property since $9 > 5$ (regardless of the values marked by $*$ in the middle), and so it is a witness interval.

does not satisfy \mathcal{P} (i.e., contains a forbidden pattern for \mathcal{P}), regardless of what the values in the *interior* of the interval are. This is a modification of a definition from [14], though the notions of a left and right extremal witness are new and were developed to accommodate the more randomized structure in this work. See also Figure 3.

Definition 6.12 (Witness interval). Let \mathcal{P} be a local property of length- n sequences, and \mathcal{F} be the family of forbidden patterns characterizing \mathcal{P} . The interval $I = [a : b]$ is said to be a *witness interval* for the sequence $f : [n] \rightarrow \mathbb{R}$ not satisfying \mathcal{P} , if one of the following holds:

- (Standard witness interval.) $a > 1$ and $b < n$, and all sequences $f' : [b - a + 1] \rightarrow \mathbb{R}$, where $f'(1) = f(a)$ and $f'(b - a + 1) = f(b)$, contain at least one copy of a pattern from \mathcal{F} .
- (Left extremal witness interval.) $a = 1$, and all sequences $f' : [b] \rightarrow \mathbb{R}$ with $f'(b) = f(b)$ contain a pattern from \mathcal{F} .
- (Right extremal witness interval.) $b = n$, and all $f' : [n - a + 1] \rightarrow \mathbb{R}$ with $f'(1) = f(a)$ contain a pattern from \mathcal{F} .

Let $T = (T_0, T_1, \dots, T_\ell)$ be an a -shifted (ℓ, w) -hierarchical partition of $[n]$. For all $i, j \in [0 : \ell]$, an interval $J \in T_j$ is an *ancestor* of an interval $I \in T_i$ if $i < j$ and $I \subset J$. For $i < \ell$, the *parent* of I is its unique ancestor in T_{i+1} . An interval I in $\bigcup_{i=0}^{\ell} T_i$ is a *maximal witness* (with respect to f, \mathcal{P}, T) if I is a witness interval for f with respect to \mathcal{P} , while all ancestors of I are not witnesses.

To know whether $[a : b]$ is a witness interval for f with respect to a given property, one only needs to query $f(a)$ and $f(b)$ in the case that $a > 1$ and $b < n$; only $f(b)$ when $a = 1$; and only $f(a)$ when $b = n$. The main structural statement connecting being far from a property \mathcal{P} and the structure of witness intervals for f in the hierarchical partition is as follows.

Lemma 6.13. *Let $f: [n] \rightarrow \mathbb{R}$ be ϵ -far from a local property \mathcal{P} . Let $T = (T_0, \dots, T_\ell)$ be an a -shifted (ℓ, w) -hierarchical partition of $[n]$, where $a \leq \lceil \epsilon n/4 \rceil$, and ℓ and w are as defined in Algorithm 6. The set \mathcal{U} of all maximal witness intervals in f with respect to T and \mathcal{P} satisfies:⁹*

$$\sum_{i=0}^{\ell} \frac{|T_i \cap \mathcal{U}|}{|T_i|} \geq \frac{\epsilon}{8}.$$

PROOF. If $T_\ell \cap \mathcal{U} \neq \emptyset$, the lemma follows since $|T_\ell| \leq \frac{n}{\epsilon n/8} = \frac{8}{\epsilon}$. Otherwise, each $I \in \mathcal{U}$ has a (non-witness) parent $P(I)$ in T . For convenience, for each interval I , define $g(I) := I \cap [a : a + w]$.

We first claim that $|g(P(I))| = 2|g(I)| - 1$. Indeed, for $I \in T_i$ for some $i \in [0 : \ell]$, we have

$$|g(I)| = 2^i + 1 \quad \text{and} \quad |g(P(I))| = 2^{i+1} + 1 = 2|g(I)| - 1. \quad (12)$$

Note that the above is trivial for non-extremal intervals, and follows from definition for the extremal ones. Consequently,

$$1 + \sum_{I \in \mathcal{U}} |g(P(I))| \leq 2 \sum_{I \in \mathcal{U}} |g(I)|. \quad (13)$$

We now bound the left-hand side from below, assuming that f is ϵ -far from \mathcal{P} . For any interval $I = [b : c]$, define its set of endpoints as $\{b, c\} \setminus \{1, n\}$, and define the interior $\text{Int}(I)$ of I as the set of all non-endpoint elements in I . Let

$$S = \bigcup_{I \in \mathcal{U}} \text{Int}(P(I)) \cup [1 : a] \cup [a + w : n].$$

Claim 6.14. *There exists $f' \in \mathcal{P}$ such that $f'(x) = f(x)$ for all $x \in [n] \setminus S$.*

PROOF. Let \mathcal{F} be the forbidden family characterizing \mathcal{P} .

First, for any pair of consecutive elements $j, j + 1$ which both belong to $[n] \setminus S$, consider the interval $A_j = [j : j + 1] \in T_0$. This interval is not contained in an interval from \mathcal{U} . By the maximality of \mathcal{U} , the interval A_j must be a non-witness, meaning that $(f(j), f(j + 1)) \notin \mathcal{F}$.

Let L denote the minimal interval in T which contains $[1 : a + 1]$ and is a non-witness. Note that either $L = [1 : a + 1]$ or $L = P(I)$ for some $I \in \mathcal{U}$. Define R similarly replacing $[1 : a + 1]$ with $[a + w - 1 : n]$. The intervals L, R and all intervals $P(I)$, for $I \in \mathcal{U}$, are non-witnesses (by definition of \mathcal{U}). Thus, one can remove all \mathcal{F} -copies from these intervals by only modifying their interiors.

Let f' be the sequence resulting from f by making these modifications (only in interiors of intervals from $\{L, R\} \cup \{P(I) : I \in \mathcal{U}\}$; for all other elements x , we set $f'(x) = f(x)$). Observe that $f'(x) = f(x)$ for $x \notin S$. Indeed, since $\text{Int}(L) \subseteq [1 : a] \cup \bigcup_{I \in \mathcal{U}} \text{Int}(P(I))$ and $\text{Int}(R) \subseteq [a + w : n] \cup \bigcup_{I \in \mathcal{U}} \text{Int}(P(I))$, the modified elements all lie in S .

Our choice of f' implies that $(f'(j), f'(j + 1)) \notin \mathcal{F}$ for each pair of elements $\{j, j + 1\}$ with nonempty intersection with S . For pairs $\{j, j + 1\} \subseteq [n] \setminus S$, we have seen in the first paragraph that $(f(j), f(j + 1)) \notin \mathcal{F}$. Since $j, j + 1 \notin S$, we have $f'(j) = f(j)$ and $f'(j + 1) = f(j + 1)$. Hence, $(f'(j), f'(j + 1)) \notin \mathcal{F}$. Thus, f' does not contain \mathcal{F} -copies, completing the proof of Claim 6.14. \square

We now complete the proof of Lemma 6.13. Since f is ϵ -far from \mathcal{P} , we have that $|S| \geq \epsilon n$. On the other hand, $|[1 : a]| + |[a + w : n]| = n - w + 1 \leq \frac{\epsilon n}{4} + 2^\ell + 1 \leq \frac{\epsilon n}{2} + 1$ and so $\bigcup_{I \in \mathcal{U}} |g(P(I))| \geq \epsilon n - \frac{\epsilon n}{2} - 1 = \frac{\epsilon n}{2} - 1$. From (13), we conclude that $\sum_{I \in \mathcal{U}} |g(I)| \geq \frac{\epsilon n}{4}$.

⁹Note that T_i and $T_i \cap \mathcal{U}$ are collections of sets; for example, $|T_i|$ is the *number of intervals* in T_i (and not, say, the total number of elements in all intervals in T_i).

Since $|T_i| \leq n/2^i$ for all $i \in [0 : \ell]$, we derive the statement of the lemma:

$$\sum_{i=0}^{\ell} \frac{|T_i \cap \mathcal{U}|}{|T_i|} \geq \sum_{i=0}^{\ell} \sum_{I \in T_i \cap \mathcal{U}} \frac{2^i}{n} \geq \frac{1}{n} \sum_{i=0}^{\ell} \sum_{I \in T_i \cap \mathcal{U}} \frac{|g(I)|}{2} \geq \frac{\epsilon}{8},$$

where in the second inequality we used the fact that $|g(I)| = 2^i + 1 \leq 2^{i+1}$, established above. \square

PROOF OF LEMMA 6.11. The tester always accepts when f satisfies \mathcal{P} . Suppose now that f is ϵ -far from \mathcal{P} . Let T and \mathcal{U} be as guaranteed in the statement of Lemma 6.13. Consider a specific iteration of the loop in Algorithm 6. For a specific value of $i \in [0 : \ell]$, the probability that the tester rejects is at least $\frac{|T_i \cap \mathcal{U}|}{|T_i|}$. Thus, the rejection probability for i sampled uniformly from that range is at least

$$\frac{1}{\ell + 1} \sum_{i=0}^{\ell} \frac{|T_i \cap \mathcal{U}|}{|T_i|} \geq \frac{1}{\log(\epsilon n)} \cdot \frac{\epsilon}{8} = \frac{\epsilon}{8 \log(\epsilon n)},$$

where the inequality follows from Lemma 6.13. That is, the rejection probability in a single iteration is at least $\alpha = \frac{\epsilon}{8 \log(\epsilon n)}$, independently of other rounds. After $\frac{5}{2\alpha} = 20\epsilon^{-1} \log(\epsilon n)$ iterations, the probability that the tester accepts is at most $(1 - \alpha)^{5/(2\alpha)} < \frac{1}{7}$, and it correctly rejects otherwise. \square

Lemma 6.15 (Marginal probability for pairs in structured tester). *Let $\epsilon > 0$ and $n \geq 16/\epsilon$. Let ℓ be as in Algorithm 6. For $x, y \in [n]$, where $x < y$, the probability that x and y are queried in Line 7 of Algorithm 6 is at most $\frac{8}{n \log(\epsilon n)}$ if $y - x = 2^i$ for some $i \in [0 : \ell]$, and equals zero otherwise.*

PROOF. The definition of a query-pair $Q_{T,i}(\cdot)$ immediately implies the probability zero statement.

Fix $i \in [0 : \ell]$. Denote by E_i the event that i was sampled in Line 6 of Algorithm 6. Note that $\Pr[E_i] = \frac{1}{\ell+1} < \frac{2}{\log(\epsilon n)}$. For x to be queried in a certain iteration (conditioned on E_i), the following two events must take place:

- *Event A:* $x - a$ is an integer multiple of 2^i , for a as defined in Line 4 of Algorithm 6.
- *Event B:* An interval sampled in Line 6 leads to x being queried.

We start by bounding $\Pr[A|E_i]$. The number of multiples of 2^i in an interval of length $m = \lceil \epsilon n/4 \rceil$ is at most $\frac{\epsilon n}{4 \cdot 2^i} + 1 \leq \frac{\epsilon n}{2^{i+1}}$. Since $a \in [m]$ is picked uniformly at random, $\Pr[A|E_i] \leq \frac{\epsilon n}{2^{i+1}} \cdot \frac{4}{\epsilon n} \leq \frac{1}{2^{i-1}}$. We next bound $\Pr[B|A, E_i]$. There are at least $n/2^{i+1}$ intervals in T_i , out of which at most one leads to x being queried. Thus $\Pr[B|A, E_i] \leq \frac{2^{i+1}}{n}$. Combining all of our bounds, we have

$$\Pr[B \wedge A \wedge E_i] = \Pr[B|A, E_i] \cdot \Pr[A|E_i] \cdot \Pr[E_i] \leq \frac{2^{i+1}}{n} \cdot \frac{1}{2^{i-1}} \cdot \frac{2}{\log(\epsilon n)} = \frac{8}{n \log(\epsilon n)}. \quad \square$$

Lemma 6.15 allows us to show that Algorithm 5 can in a strong sense “simulate” the structured tester in Algorithm 6. In the following lemma, we use such a simulation argument to prove that Algorithm 5 is indeed a valid pair tester (in an *offline* setting) for \mathcal{P} .

PROOF OF LEMMA 6.2. By Lemma 6.11, Algorithm 6 is an ϵ -tester for \mathcal{P} with one-sided error in the offline model. We couple a single run of Algorithm 6 with parameters n, \mathcal{P}, ϵ with a single run of Algorithm 5 with the same parameters. Specifically, we couple (i) each iteration of the loop in Algorithm 6 with (ii) ten iterations of the loop in Algorithm 5.

For each pair $(x, y) \in D_d(n)$, if d is not of the form $d = 2^i$ for $i \in [0 : \ell]$ (for ℓ as defined in both algorithms), then the probability that x and y are queried in both setups (i) and (ii) is zero. Suppose now that $d = 2^i$ for $i \in [0 : \ell]$. By Lemma 6.15, the probability that x and y are queried in setting (i) is at most $\frac{8}{n \log(\epsilon n)}$. In Algorithm 5, the probability that x and y are queried in a single iteration of the loop is at least $\frac{1}{n} \cdot \frac{1}{\log(\epsilon n)} = \frac{1}{n \log(\epsilon n)}$. Each of

the ten iterations in setting (ii) is independent, and so the probability that (x, y) is *not* queried in any of these iterations is bounded by $\left(1 - \frac{1}{n \log(\epsilon n)}\right)^{10} < 1 - \frac{8}{n \log(\epsilon n)}$. That is, (x, y) is queried in at least one iteration with probability at least $\frac{8}{n \log(\epsilon n)}$. The lemma follows since loop iterations in both algorithms are independent. \square

6.3 Impossibility Result: No Tester for $\omega(\epsilon)$ Erasure Rate

In this section we complete the proof of Theorem 1.4, via a suitable lower bound construction.

PROOF OF THEOREM 1.4, ITEM 2 (IMPOSSIBILITY). Let $\epsilon \in (0, \frac{1}{3}]$. Let $n \in \mathbb{N}$ be even. Using the same notation as in [40], we define the following distributions \mathcal{D}^+ and \mathcal{D}^- over functions $f : [n] \rightarrow [n]$. This is the construction proposed and analyzed in [40], but we replaced $\frac{1}{3}$ with ϵ .

\mathcal{D}^+ *distribution*: independently for all $i \in [n/2]$:

- $f(2i - 1) = f(2i) = 2i - 1$ with probability ϵ . (In this case, we call $(2i - 1, 2i)$ a “low pair”.)
- $f(2i - 1) = f(2i) = 2i$ with probability ϵ . (Here we call $(2i - 1, 2i)$ a “high pair.”)
- $f(2i - 1) = 2i - 1$ and $f(2i) = 2i$ with probability $1 - 2\epsilon$. (“Increasing pair.”)

\mathcal{D}^- *distribution*: independently for all $i \in [n/2]$:

- $f(2i - 1) = 2i$ and $f(2i) = 2i - 1$ with probability ϵ . (“Violating pair.”)
- $f(2i - 1) = 2i - 1$ and $f(2i) = 2i$ with probability $1 - \epsilon$. (Increasing pair.)

All functions $f \in \mathcal{D}^+$ are monotone, whereas the expected distance of a function $f \in \mathcal{D}^-$ from monotonicity is ϵ . It is straightforward to verify using Chernoff bound that the distance of $f \in \mathcal{D}^-$ to monotonicity is bigger than $\frac{\epsilon}{2}$ with probability $\frac{99}{100}$ as long as, say, $n > \frac{20}{\epsilon}$; see the discussion of Yao’s minimax principle after Claim 4.1 and [40, Corollary 9.4] for more details.

Consider the following budget-managing adversary in the erasure model.¹⁰

- When $f \in \mathcal{D}^-$: If the tester queries an element of a violating pair, then the adversary erases the other element of this pair (i.e., the adversary erases $2i$ if $2i - 1$ was queried, and vice versa). If the tester queries an element of an increasing pair, then the adversary erases the other element with probability $\frac{\epsilon}{1 - \epsilon}$.
- When $f \in \mathcal{D}^+$: If the tester queries an element of a low pair or a high pair, the adversary erases the other element. Otherwise (the increasing pair case), the adversary does not erase.

Indistinguishability of \mathcal{D}^+ and \mathcal{D}^- under adversarial erasures. We claim the tester cannot distinguish these two distributions given the above adversarial strategy. Indeed, first observe that if an element x was queried and its paired element was not erased, then they necessarily constitute an increasing pair. Thus, the only information that the adversary can extract by querying the element x' paired to x is the probability that x' is subsequently erased after x is queried, conditioned on the value of $f(x)$.

Suppose that, say, $x = 2i - 1$ (the case $x = 2i$ is symmetric). First, consider distribution \mathcal{D}^+ . If $f(x) = x = 2i - 1$, then we are either at the “low pair” or “increasing pair” case, and the other element x' is erased only in the low pair case. This happens with probability

$$\frac{\epsilon}{\epsilon + (1 - 2\epsilon)} = \frac{\epsilon}{1 - \epsilon}.$$

¹⁰Our description of the adversarial strategy assumes that the adversary knows whether f was generated from \mathcal{D}^+ or \mathcal{D}^- . However, this assumption is not really needed: the only function that can be generated with nonzero probability from both of these distributions is the identity function ($f(x) = x$ for all x), and since the generation probability is exponentially small in n , this does not materially affect the analysis.

If $f(x) = 2i = x + 1$ in this case, then we are in the “high pair” category, and x' is erased with probability 1. For the \mathcal{D}^- distribution, by definition of the adversary, it erases x' if $f(x) = x$ with probability $\epsilon/(1 - \epsilon)$, and with probability 1 if $f(x) = x + 1$. Thus, the probabilities are equal for both distributions.

Budget analysis. It remains to verify that for an adversarial erasure rate of $C \cdot \epsilon$ for some large enough absolute constant C , the adversary’s budget suffices to run the above strategy at all times. We do so using a combination of Markov inequality (for the earlier parts of the process) and Chernoff inequality (for the later parts).

For any round $i \in \mathbb{N}$ of the process, let X_i denote the random variable indicating whether the tester queried an element from a previous unqueried non-increasing pair of f . Note that the probability of each X_i , even conditioned on all previous outcomes, is at most 2ϵ . Denote $Y_\ell = \sum_{i=1}^\ell X_i$ and note that $\mathbb{E}[Y_\ell] \leq 2\epsilon \cdot \ell$. Let $\ell_0 = 2/(\epsilon \cdot C)$ be a crude upper bound for the time it takes the adversary to add an erasure to its budget. By time $\ell_0 m$ the adversary has a budget of at least m erasures. Note that if $Y_{\ell_0 m} < m$ for all $m \in \mathbb{N}$ then the adversary has sufficient erasure budget at all times, as time steps in between follow automatically. Thus, in any execution where the adversary fails, there must exist some $m \in \mathbb{N}$ for which $Y_{\ell_0 m} \geq m$. We denote these bad events by Z_m (for any $m \in \mathbb{N}$), and bound the probability of any of these events ever happening. We use Chernoff inequality with the upper bound on the expectation:

$$\Pr[Z_m] = \Pr[Y_{\ell_0 m} \geq m] = \Pr[Y_{\ell_0 m} \geq (1 + \delta) \cdot 2\epsilon \ell_0 m] \leq \exp\left(-\frac{\delta^2 \cdot 2\epsilon \ell_0 m}{2 + \delta}\right),$$

where we set $C = 60$, and $\delta = 14$, so that $(1 + \delta) \cdot 2\epsilon \ell_0 = 1$. This implies $\epsilon \ell_0 = \frac{2}{C} = \frac{1}{30}$, and $\frac{2\delta^2}{2+\delta} = 24.5$, leading to

$$\Pr[Z_m] \leq e^{-49m/30} < \frac{1}{5^m}.$$

Finally, by a union bound, the erasure budget is insufficient with probability at most

$$\Pr[\cup_{m \in \mathbb{N}} Z_m] \leq \sum_{m=1}^{\infty} \Pr[Z_m] < \sum_{m=1}^{\infty} \frac{1}{5^m} = \frac{1}{4}.$$

Thus, with probability more than $\frac{3}{4}$, the adversary can maintain its strategy indefinitely. \square

Acknowledgments

We thank Shachar Lovett for referring us to [13, 45], which led to the result in Section 4. This research was partly conducted while EK and UM were visiting the Simons Institute for the Theory of Computing.

References

- [1] Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. 2005. Testing Reed-Muller codes. *IEEE Transactions on Information Theory* 51, 11 (2005), 4032–4039. doi:10.1109/TIT.2005.856958
- [2] Sanjeev Arora and Madhu Sudan. 2003. Improved Low-Degree Testing and its Applications. *Combinatorica* 23, 3 (2003), 365–426. doi:10.1007/s00493-003-0025-0
- [3] Vipul Arora, Esty Kelman, and Uri Meir. 2025. On optimal testing of linearity. In *2025 Symposium on Simplicity in Algorithms (SOSA)*. SIAM, 65–76.
- [4] Pranjal Awasthi, Madhav Jha, Marco Molinaro, and Sofya Raskhodnikova. 2016. Testing Lipschitz Functions on Hypergrid Domains. *Algorithmica* 74, 3 (2016), 1055–1081. doi:10.1007/s00453-015-9984-y
- [5] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. 1991. Checking Computations in Polylogarithmic Time. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 21–31. doi:10.1145/103418.103428
- [6] László Babai, Lance Fortnow, and Carsten Lund. 1991. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Computational Complexity* 1 (1991), 3–40. doi:10.1007/BF01200056
- [7] József Beck. 2008. *Combinatorial Games: Tic-Tac-Toe Theory*. Cambridge: Cambridge University Press.
- [8] Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. 1996. Linearity testing in characteristic two. *IEEE Transactions on Information Theory* 42, 6 (1996), 1781–1795. doi:10.1109/18.556674

- [9] Mihir Bellare, Oded Goldreich, and Madhu Sudan. 1998. Free Bits, PCPs, and Non-approximability - Towards Tight Results. *SIAM Journal on Computing (SICOMP)* 27, 3 (1998), 804–915. doi:10.1137/S0097539796302531
- [10] Mihir Bellare, Shafi Goldwasser, Carsten Lund, and Alexander Russell. 1993. Efficient probabilistically checkable proofs and applications to approximations. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*. 294–304. doi:10.1145/167088.167174
- [11] Mihir Bellare and Madhu Sudan. 1994. Improved non-approximability results. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*. 184–193. doi:10.1145/195058.195129
- [12] Aleksandrs Belovs. 2018. Adaptive Lower Bound for Testing Monotonicity on the Line. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*. 31:1–31:10. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.31
- [13] Ido Ben-Eliezer, Rani Hod, and Shachar Lovett. 2012. Random low-degree polynomials are hard to approximate. *Computational Complexity* 21, 1 (2012), 63–81.
- [14] Omri Ben-Eliezer. 2019. Testing Local Properties of Arrays. In *10th Innovations in Theoretical Computer Science Conference (ITCS)*. 11:1–11:20.
- [15] Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Ron D. Rothblum. 2020. Hard Properties with (Very) Short PCPPs and Their Applications. In *Proceedings, Innovations in Theoretical Computer Science (ITCS)*. 9:1–9:27. doi:10.4230/LIPIcs.ITCS.2020.9
- [16] Omri Ben-Eliezer, Esty Kelman, Uri Meir, and Sofya Raskhodnikova. 2024. Property Testing with Online Adversaries. In *15th Innovations in Theoretical Computer Science Conference (ITCS)*. 11:1–11:25.
- [17] Eli Ben-Sasson, Madhu Sudan, Salil P. Vadhan, and Avi Wigderson. 2003. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*. 612–621. doi:10.1145/780542.780631
- [18] Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. 2014. L_p -testing. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*. 164–173. doi:10.1145/2591796.2591887
- [19] Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. 2012. Transitive-Closure Spanners. *SIAM Journal on Computing (SICOMP)* 41, 6 (2012), 1380–1425. doi:10.1137/110826655
- [20] Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. 2010. Optimal Testing of Reed-Muller Codes. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*. 488–497. doi:10.1109/FOCS.2010.54
- [21] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. 1993. Self-Testing/Correcting with Applications to Numerical Problems. *J. Comput. System Sci.* 47, 3 (1993), 549–595. doi:10.1016/0022-0000(93)90044-W
- [22] Deeparnab Chakrabarty, Kashyap Dixit, Madhav Jha, and C. Seshadhri. 2017. Property Testing on Product Distributions: Optimal Testers for Bounded Derivative Properties. *ACM Transactions on Algorithms (TALG)* 13, 2 (2017), 20:1–20:30. doi:10.1145/3039241
- [23] Deeparnab Chakrabarty and C. Seshadhri. 2013. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*. 419–428. doi:10.1145/2488608.2488661
- [24] Deeparnab Chakrabarty and C. Seshadhri. 2014. An Optimal Lower Bound for Monotonicity Testing over Hypergrids. *Theory of Computing* 10 (2014), 453–464. doi:10.4086/toc.2014.v010a017
- [25] Irit Dinur and Venkatesan Guruswami. 2013. PCPs via Low-Degree Long Code and Hardness for Constrained Hypergraph Coloring. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*. 340–349. doi:10.1109/FOCS.2013.44
- [26] Kashyap Dixit, Madhav Jha, Sofya Raskhodnikova, and Abhradeep Thakurta. 2013. Testing the Lipschitz Property over Product Distributions with Applications to Data Privacy. In *Theory of Cryptography Conference (TCC)*. 418–436. doi:10.1007/978-3-642-36594-2_24
- [27] Kashyap Dixit, Sofya Raskhodnikova, Abhradeep Thakurta, and Nithin Varma. 2018. Erasure-Resilient Property Testing. *SIAM Journal on Computing (SICOMP)* 47, 2 (2018), 295–329. doi:10.1137/16M1075661
- [28] Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. 1999. Improved Testing Algorithms for Monotonicity. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*. 97–108. doi:10.1007/978-3-540-48413-4_10
- [29] Funda Ergün, Sampath Kannan, Ravi Kumar, Ronitt Rubinfeld, and Mahesh Viswanathan. 2000. Spot-checkers. *J. Comput. System Sci.* 60, 3 (2000), 717–751. doi:10.1006/jcss.1999.1692
- [30] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. 1996. Interactive Proofs and the Hardness of Approximating Cliques. *J. ACM* 43, 2 (1996), 268–292. doi:10.1145/226643.226652
- [31] Eldar Fischer. 2004. On the strength of comparisons in property testing. *Information and Computation* 189, 1 (2004), 107–116. doi:10.1016/j.ic.2003.09.003
- [32] Katalin Friedl and Madhu Sudan. 1995. Some Improvements to Total Degree Tests. In *Third Israel Symposium on Theory of Computing and Systems (ISTCS)*. 190–198. doi:10.1109/ISTCS.1995.377032
- [33] Peter Gemmel, Richard J. Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. 1991. Self-Testing/Correcting for Polynomials and for Approximate Functions. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*. 32–42. doi:10.1145/103418.103429
- [34] Oded Goldreich and Dana Ron. 2017. On learning and testing dynamic environments. *Journal of the ACM (JACM)* 64, 3 (2017), 1–90.
- [35] Elad Haramaty, Amir Shpilka, and Madhu Sudan. 2013. Optimal Testing of Multivariate Polynomials over Small Prime Fields. *SIAM Journal on Computing (SICOMP)* 42, 2 (2013), 536–562. doi:10.1137/120879257

- [36] Johan Håstad and Avi Wigderson. 2003. Simple analysis of graph tests for linearity and PCP. *Random Structures and Algorithms* 22, 2 (2003), 139–160. doi:10.1002/rsa.10068
- [37] Dan Hefetz, Michael Krivelevich, Miloš Stojaković, and Tibor Szabó. 2014. *Positional Games*. Oberwolfach Seminars. Vol. 44.
- [38] Madhav Jha and Sofya Raskhodnikova. 2013. Testing and Reconstruction of Lipschitz Functions with Applications to Data Privacy. *SIAM Journal on Computing (SICOMP)* 42, 2 (2013), 700–731. doi:10.1137/110840741
- [39] Charanjit S. Jutla, Anindya C. Patthak, Atri Rudra, and David Zuckerman. 2009. Testing low-degree polynomials over prime fields. *Random Structures and Algorithms* 35, 2 (2009), 163–193. doi:10.1002/rsa.20262
- [40] Iden Kalemaj, Sofya Raskhodnikova, and Nithin Varma. 2023. Sublinear-Time Computation in the Presence of Online Erasures. *Theory of Computing* 19(1) (2023), 1–48. <http://theoryofcomputing.org/articles/v019a001/>
- [41] Tali Kaufman, Simon Litsyn, and Ning Xie. 2010. Breaking the Epsilon-Soundness Bound of the Linearity Test over GF(2). *SIAM Journal on Computing (SICOMP)* 39, 5 (2010), 1988–2003. doi:10.1137/080715548
- [42] Tali Kaufman and Dor Minzer. 2022. Improved optimal testing results from global hypercontractivity. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, 98–109.
- [43] Tali Kaufman and Dana Ron. 2006. Testing Polynomials over General Fields. *SIAM Journal on Computing (SICOMP)* 36, 3 (2006), 779–802. doi:10.1137/S0097539704445615
- [44] Tali Kaufman and Madhu Sudan. 2008. Algebraic property testing: the role of invariance. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, 403–412.
- [45] Peter Keevash and Benny Sudakov. 2005. Set systems with restricted cross-intersections and the minimum rank of inclusion matrices. *SIAM Journal on Discrete Mathematics* 18, 4 (2005), 713–727.
- [46] Amit Levi, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. 2021. Erasure-Resilient Sublinear-Time Graph Algorithms. In *Proceedings, Innovations in Theoretical Computer Science (ITCS)*, 80:1–80:20. doi:10.4230/LIPIcs.ITCS.2021.80
- [47] Dor Minzer and Kai Zhe Zheng. 2024. Adversarial Low Degree Testing. In *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 4395–4409.
- [48] Dana Moshkovitz. 2017. Low-degree test with polynomially small error. *Computational Complexity* 26, 3 (2017), 531–582. doi:10.1007/s00037-016-0149-4
- [49] Dana Moshkovitz and Ran Raz. 2008. Sub-Constant Error Low Degree Test of Almost-Linear Size. *SIAM Journal on Computing (SICOMP)* 38, 1 (2008), 140–180. doi:10.1137/060656838
- [50] Yonatan Nakar and Dana Ron. 2021. Testing Dynamic Environments: Back to Basics. In *48th International Colloquium on Automata, Languages, and Programming (ICALP)*, 98:1–98:20.
- [51] Ilan Newman and Nithin Varma. 2021. New Sublinear Algorithms and Lower Bounds for LIS Estimation. In *Proceedings, International Colloquium on Automata, Languages and Programming (ICALP)*, 100:1–100:20. doi:10.4230/LIPIcs.ICALP.2021.100
- [52] Ryan O’Donnell. 2014. *Analysis of Boolean Functions*. Cambridge University Press.
- [53] Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. 2018. Parameterized Property Testing of Functions. *ACM Transactions on Computation Theory* 9, 4 (2018), 17:1–17:19. doi:10.1145/3155296
- [54] Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. 2022. Approximating the distance to monotonicity of Boolean functions. *Random Structures and Algorithms* 60, 2 (2022), 233–260. doi:10.1002/rsa.21029
- [55] Sofya Raskhodnikova. 1999. Monotonicity Testing. *Masters Thesis, MIT* (1999).
- [56] Sofya Raskhodnikova. 2016. Testing if an array is sorted. *Encyclopedia of Algorithms* (2016), 2219–2222. doi:10.1007/978-1-4939-2864-4_700
- [57] Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma. 2021. Erasures versus errors in local decoding and property testing. *Random Structures and Algorithms* 59, 4 (2021), 640–670. doi:10.1002/rsa.21031
- [58] Sofya Raskhodnikova and Ronitt Rubinfeld. 2016. Linearity Testing/Testing Hadamard Codes. In *Encyclopedia of Algorithms*. Springer, 1107–1110. doi:10.1007/978-1-4939-2864-4_202
- [59] Sofya Raskhodnikova and Nithin Varma. 2018. Brief Announcement: Erasure-Resilience Versus Tolerance to Errors. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*, 111:1–111:3.
- [60] Ran Raz and Shmuel Safra. 1997. A Sub-Constant Error-Probability Low-Degree Test, and a Sub-Constant Error-Probability PCP Characterization of NP. In *Proceedings of the 29th ACM Symposium on Theory of Computing (STOC)*, 475–484.
- [61] Noga Ron-Zewi and Madhu Sudan. 2013. A New Upper Bound on the Query Complexity of Testing Generalized Reed-Muller Codes. *Theory of Computing* 9 (2013), 783–807. doi:10.4086/toc.2013.v009a025
- [62] Ronitt Rubinfeld and Madhu Sudan. 1996. Robust Characterizations of Polynomials with Applications to Program Testing. *SIAM Journal on Computing (SICOMP)* 25, 2 (1996), 252–271. doi:10.1137/S0097539793255151
- [63] Alex Samorodnitsky. 2007. Low-degree tests at large distances. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 506–515. doi:10.1145/1250790.1250864
- [64] Alex Samorodnitsky and Luca Trevisan. 2000. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 191–199. doi:10.1145/335305.335329

- [65] Alex Samorodnitsky and Luca Trevisan. 2009. Gowers Uniformity, Influence of Variables, and PCPs. *SIAM Journal on Computing (SICOMP)* 39, 1 (2009), 323–360. doi:10.1137/070681612
- [66] Amir Shpilka and Avi Wigderson. 2006. Derandomizing Homomorphism Testing in General Groups. *SIAM Journal on Computing (SICOMP)* 36, 4 (2006), 1215–1230. doi:10.1137/S009753970444658X
- [67] Madhu Sudan and Luca Trevisan. 1998. Probabilistically Checkable Proofs with Low Amortized Query Complexity. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, 18–27. doi:10.1109/SFCS.1998.743425
- [68] Luca Trevisan. 1998. Recycling Queries in PCPs and in Linearity Tests (Extended Abstract). In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 299–308. doi:10.1145/276698.276769

Just Accepted

A Observation about Low-Degree Testing with Large Batches

This section is devoted to the proof of the following observation on online testing of property \mathcal{P}_d (being a polynomial of degree at most d) with batches of size $b = 2^{d+1}$.

Observation A.1. Fix $d, n \in \mathbb{N}$, $\epsilon \in (0, 1/2)$, and t such that $t \leq \min \{\epsilon^2 2^{2d}, \eta\} \cdot 2^{n-d}/10^5$ for some constant η , and $d < n/3$. Then there exists an ϵ -tester for property \mathcal{P}_d of functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ that works in the presence of a $(2^{d+1}, t)$ -online erasure budget-managing adversary and makes $O(\frac{1}{\epsilon} + 2^d)$ queries.

PROOF. With batch size $b = 2^{d+1}$, one could use a single batch to run the standard degree- d test: choose a $(d+1)$ -dim affine subspace uniformly at random, and reject if the restricted function is not of degree d – that is, the XOR of all returned values is 1 (and in particular, none of them is erased). By [20, Theorem 1], there exists a constant ζ , such that any ϵ -far function is rejected by the test with probability at least $s := \min \{2^{d+1}\epsilon, \zeta\}$, while any degree- d function is always accepted. We next consider $r = \frac{10}{s}$ iterations of this atomic test, and analyze the probability of seeing an erasure.

For completeness, note that erasures cannot cause the tester to reject, so a degree- d function is always accepted.

For soundness, fix an ϵ -far function f , and consider the iteration $i+1 \in [r]$. At most $i \cdot t$ points were erased so far. Each queried point has a marginal distribution of a uniformly random point, with probability at most $i \cdot t/2^n$ to be a previously erased point. Using a union bound over all 2^{d+1} points, the probability of seeing *any* erasure during iteration i is at most

$$\frac{it2^{d+1}}{2^n} \leq \frac{10t2^{d+1}}{s \cdot 2^n} \leq \frac{s}{100},$$

where the last inequality is due to the premise on t (choosing $\eta = \zeta^2$ which is constant).

Overall, the probability of the test choosing a pattern that would (wrongfully) pass the test is at most $1 - s$, and the probability of seeing an erasure is at most $s/100$. By a union bound, the probability to accept f at any iteration is at most $1 - s/2$. Using $r = \frac{10}{s}$ iterations, the probability to accept an ϵ -far function is at most $\frac{1}{3}$.

The query complexity is $2^{d+1} \cdot r = O(\max\{\frac{1}{\epsilon}, 2^d\})$.

Note that the probability of seeing *any* erasure is at most $\frac{s}{100} \cdot \frac{10}{s} = \frac{1}{10}$, which allows this tester to deal with corruptions as well, by Lemma 2.5. \square

B Proof of Claim 5.7 (Good Patterns for \mathcal{P}_d)

We first recall the definition of a good pattern for property \mathcal{P} .

Definition 5.2 (Good patterns). A pattern $X_{\ell \times m}$ is *complete* for a property \mathcal{P} if for all functions $f \in \mathcal{P}$,

$$\Pr_{a_1, \dots, a_m} [T_{X, f}(a_1, \dots, a_m) = 1] = 0.$$

A pattern $X_{\ell \times m}$ is *minimally sound* for a property \mathcal{P} if for all functions $f \notin \mathcal{P}$,

$$\Pr_{a_1, \dots, a_m} [T_{X, f}(a_1, \dots, a_m) = 1] > 0.$$

If a pattern is both complete and minimally sound for \mathcal{P} , we call it *good* for \mathcal{P} .

Let us recall the claim we wish to prove.

Claim 5.7. A pattern $X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ is good for \mathcal{P}_d if and only if the two conditions hold:

- (i) For all coordinate sets $S \subseteq [m]$ of size $|S| \leq d$, we have $\sum_{i \in [\ell]} \prod_{j \in S} X_{ij} = 0$.
- (ii) There exists a set $S \subseteq [m]$ of size $|S| = d+1$ such that $\sum_{i \in [\ell]} \prod_{j \in S} X_{ij} = 1$.

PROOF. In the following, we use the notation A_i to denote the i^{th} row of the matrix A , and think of a set $J \subseteq [n]$ as being ordered $J = \{J_1, \dots, J_r\}$ with $J_1 < \dots < J_r$, and size $r = |J|$. Denote by $g_J: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ the monomial corresponding to the set of inputs J , i.e., $g_J(x_1, \dots, x_n) = \prod_{j \in J} x_j$. The class of degree- d functions, \mathcal{P}_d , is a linear space spanned by such monomials:

$$\mathcal{P}_d = \text{sp}\{g_J: J \subseteq [n], |J| \leq d\}. \quad (14)$$

Consider the test with pattern X when run with parameter matrix M and oracle access to the monomial function, g_J . Let us expand the term $T_{X,g_J}(M) \in \mathbb{F}_2$ (that indicates rejection or acceptance).

$$T_{X,g_J}(M) = \sum_{i \in [\ell]} g_J((XM)_i) = \sum_{i \in [\ell]} \prod_{j \in [r]} (XM)_{i,J_j} = \sum_{i \in [\ell]} \prod_{j \in [r]} \sum_{k \in [m]} X_{i,k} M_{k,J_j},$$

where the last equality simply expands matrix multiplication. We further expand the inner multiplication of sums, which results in summation over r running indices, where each summand is a multiplication of r entries from X and r entries from M . This can be written as

$$\begin{aligned} \sum_{i \in [\ell]} \prod_{j \in [r]} \sum_{k \in [m]} X_{i,k} M_{k,J_j} &= \sum_{i \in [\ell]} \sum_{k_1 \in [m]} \sum_{k_2 \in [m]} \dots \sum_{k_r \in [m]} \prod_{a \in [r]} X_{i,k_a} \cdot \prod_{j \in [r]} M_{k_j, J_j} \\ &= \sum_{k_1 \in [m]} \sum_{k_2 \in [m]} \dots \sum_{k_r \in [m]} \left(\prod_{j \in [r]} M_{k_j, J_j} \cdot \sum_{i \in [\ell]} \prod_{a \in [r]} X_{i,k_a} \right), \end{aligned}$$

where the second equality pushes the first summation inside, noting it only affects entries of X .

Finally, we define for any pattern $X_{\ell, m}$ and subset $S \subseteq [m]$ the following

$$\varphi_S(X) := \sum_{i \in [\ell]} \prod_{j \in S} X_{ij},$$

which leads to the useful equality

$$T_{X,g_J}(M) = \sum_{k_1 \in [m]} \sum_{k_2 \in [m]} \dots \sum_{k_r \in [m]} \varphi_{\{k_1, k_2, \dots, k_r\}}(X) \cdot \prod_{j \in [r]} M_{k_j, J_j}. \quad (15)$$

A good pattern X for \mathcal{P}_d satisfies conditions (i) and (ii). Let X be a good pattern for \mathcal{P}_d . To show it satisfies (i), we need to show that for any $S \subseteq [m]$ such that $|S| = r \leq d$ it holds that $\varphi_S(X) = 0$. Fix such subset S , and consider the monomial $g_{[r]} \in \mathcal{P}_d$ (since $r \leq d$), and the instance matrix $M' \in \{0, 1\}^{m \times n}$ such that $M'_{S_j, j} = 1$ for any $j \in [r]$ and all other entries are 0. By completeness of the pattern X , we have $T_{X, g_{[r]}}(M') = 0$. By (15), we get

$$0 = T_{X, g_{[r]}}(M') = \sum_{k_1 \in [m]} \sum_{k_2 \in [m]} \dots \sum_{k_r \in [m]} \varphi_{\{k_1, k_2, \dots, k_r\}}(X) \cdot \prod_{j \in [r]} M'_{k_j, j} = \varphi_S(X),$$

where the last equality is since any choice $k_j \neq S_j$ for some $j \in [r]$ gives $M'_{k_j, j} = 0$, leaving only the unique summand where $k_j = S_j$ for all $j \in [r]$.

We next use the fact that a good pattern X is also minimally sound (recall Definition 5.2), meaning that for any $f \notin \mathcal{P}_d$ there is a rejecting instance. We use the monomial $g_{[d+1]} \notin \mathcal{P}_d$, and let M' be the rejecting instance, i.e., $T_{X, g_{[d+1]}}(M') = 1$. By (15) we get that

$$1 = T_{X, g_{[d+1]}}(M') = \sum_{k_1 \in [m]} \sum_{k_2 \in [m]} \dots \sum_{k_{d+1} \in [m]} \varphi_{\{k_1, k_2, \dots, k_{d+1}\}}(X) \cdot \prod_{j \in [d+1]} M'_{k_j, j}.$$

But this implies that not all summands are zero, and in particular there exists a choice of indices $\{k'_1, \dots, k'_{d+1}\}$ for which the summand is one. Furthermore, there exists such choice where the indices are distinct, as any non-distinct choice (by symmetry) will appear an even number of times and contribute 0 to the sum. This implies that $\varphi_{\{k'_1, k'_2, \dots, k'_{d+1}\}}(X) = 1$, proving condition (ii) is satisfied with $S = \{k'_1, k'_2, \dots, k'_{d+1}\}$ of size $d + 1$.

A pattern X satisfying conditions (i) and (ii) is good for \mathcal{P}_d . First, note that a pattern X satisfying condition (i) must accept any monomial of degree exactly r , for $r \leq d$, with any parameter matrix M . Indeed, use the expansion in (15), and observe that all summands are 0 regardless of M , as any choice of k_1, \dots, k_r has at most d indices, implying $\varphi_{\{k_1, k_2, \dots, k_r\}}(X) = 0$. Finally, recall that by (14) any $f \in \mathcal{P}_d$ can be written as $f = \sum_{J \in F} g_J$ for some choice of F where each $J \in F$ is of size at most d . We get

$$T_{X,f}(M) = \sum_{J \in F} T_{X,g_J}(M) = 0,$$

which proves X is complete for \mathcal{P}_d .

We next use condition (ii) to show X is minimally sound, concluding that it is good for \mathcal{P}_d . Let $S^* = \{s_1, s_2, \dots, s_{d+1}\}$ of size $d + 1$ be a set such that $\varphi_{S^*}(X) = 1$.

Fix $f \notin \mathcal{P}_d$ and write it as $f = \sum_{J \subseteq [n]} \alpha_J g_J$. Let $r^* > d$ be the minimal size such that $\alpha_J = 1$ for some monomial of size $|J| = r^*$, and let $J^* = \{j_1, j_2, \dots, j_{r^*}\}$ denote an arbitrary such monomial. We define the parameter matrix $M^* \in \{0, 1\}^{m \times n}$ to have a unique non-zero entry in each column whose index belongs to J^* . In particular, $M^*_{s_i, j_i} = 1$ for each $i \leq d + 1$, and $M^*_{s_{d+1}, j_i} = 1$ for each $d + 1 < i \leq r^*$, and all other entries are 0. We have

$$T_{X,f}(M^*) = \sum_{i \in [\ell]} f((XM^*)_i) = \sum_{i \in [\ell]} \sum_{J \subseteq [n]} \alpha_J g_J((XM^*)_i) = \sum_{J \subseteq [n]} \alpha_J T_{X,g_J}(M^*).$$

Using the completeness of X (which we already showed), and minimality of r^* , we have that for any $|J| < r^*$, either $\alpha_J = 0$ or $T_{X,g_J}(M^*) = 0$, contributing nothing to the sum above. For any set $|J| \geq r^*$ and $J \neq J^*$, there exists an index $J_i \in J \setminus J^*$ implying that the (J_i) 'th column of M^* is all zeroes. By (15), any choice of indices $k_1, \dots, k_{|J|}$ has a multiplicand from this column $M^*_{k_i, J_i} = 0$, proving that $T_{X,g_J}(M^*) = 0$. The only monomial left is J^* , implying that

$$T_{X,f}(M^*) = T_{X,g_{J^*}}(M^*) = \varphi_{S^*}(X) = 1,$$

where the second equation is due to our choice of M^* . \square

Received 11 April 2024; revised 19 March 2025; accepted 16 October 2025