Property Testing with Online Adversaries

Omri Ben-Eliezer
Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, USA

Esty Kelman
Department of Computer Science and Faculty of Computing & Data Sciences,
Boston University, MA, USA
CSAIL, Massachusetts Institute of Technology, Cambridge, MA, USA

Uri Meir
Blavatnik School of Computer Science, Tel Aviv University, Israel

Sofya Raskhodnikova
Department of Computer Science, Boston University, MA, USA

Abstract

The online manipulation-resilient testing model, proposed by Kalemaj, Raskhodnikova and Varma (ITCS 2022 and Theory of Computing 2023), studies property testing in situations where access to the input degrades continuously and adversarially. Specifically, after each query made by the tester is answered, the adversary can intervene and either erase or corrupt t data points. In this work, we investigate a more nuanced version of the online model in order to overcome old and new impossibility results for the original model. We start by presenting an optimal tester for linearity and a lower bound for low-degree testing of Boolean functions in the original model. We overcome the lower bound by allowing batch queries, where the tester gets a group of queries answered between manipulations of the data. Our batch size is small enough so that function values for a single batch on their own give no information about whether the function is of low degree. Finally, to overcome the impossibility results of Kalemaj et al. for sortedness and the Lipschitz property of sequences, we extend the model to include $t < 1$, i.e., 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 (with any number of queries). Our online tester works for a general class of local properties of sequences. One feature of our results is that we get new (and in some cases, simpler) optimal algorithms for several properties in the standard property testing model.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Linearity testing, low-degree testing, Reed-Muller codes, testing properties of sequences, erasure-resilience, corruption-resilience

Digital Object Identifier 10.4230/LIPIcs.ITCS.2024.11

Related Version *Full Version*: `https://arxiv.org/abs/2311.16566` [15]

Funding Research partly conducted while EK and UM were visiting the Simons Institute for the Theory of Computing.
Esty Kelman: Supported by the Computer Science Department, 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).

Acknowledgements We thank Shachar Lovett for referring us to [12, 43], which led to the result in Section 4.

15th Innovations in Theoretical Computer Science Conference (ITCS 2024).
Editor: Venkatesan Guruswami; Article No. 11; pp. 11:1–11:25
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

The online manipulation-resilient testing model, proposed by Kalemaj, Raskhodnikova and Varma [38], 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 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, where access to the data or the data itself is 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 [38], 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 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).

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 [38]) 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 [38], e.g., allowing the adversary to manipulate one point after every other query. Finally, we also study batch queries, where the tester gets a group of queries answered between manipulations of the data.

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 over $\mathbb{F}_2$, that is, of degree at most d 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 [45] and then show how to overcome the lower bound by using batch queries. Finally, we consider local properties of functions $f : [n] \to \mathbb{R}$. (We use $[n]$ to represent $\{1, 2, \ldots, n\}$). 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. Moreover, we show how to obtain optimal erasure rate with batches of size 2.

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.

1.1 Our Results and Techniques

We study two types of functions: Boolean functions $f : \{0,1\}^n \to \{0,1\}$ and real-valued functions $f : [n] \to \mathbb{R}$, 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 [38], 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. But, 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 [20], linearity testing has been extensively investigated; see, e.g., [9, 10, 29, 7, 8, 66, 65, 62, 34, 16, 61, 63, 64, 39, 38] and the survey in [56]. A function $f : \{0,1\}^n \to \{0,1\}$ 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). *There exists a constant $c > 0$ such that for all $n \in \mathbb{N}, \epsilon \in (0, 1/2]$, and t satisfying $t \log^2 t \le c \cdot \epsilon^{2.5} 2^{n/2}$, there exists an ϵ-tester for linearity of functions $f : \{0,1\}^n \to \{0,1\}$ that works in the presence of a t-online erasure (or corruption) budget-managing adversary and makes $O(\max\{1/\epsilon, \log t\})$ queries. In the case of erasure adversary, the tester has 1-sided error.*

In contrast, the linearity tester from [38] works for a smaller range of t (specifically, $t \le c_0 \cdot \epsilon^{5/4} 2^{n/4}$ for some constant c_0) and makes $O\big(\min\big(\frac{1}{\epsilon}\log\frac{t}{\epsilon}, \frac{t}{\epsilon}\big)\big)$ 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). This follows from the known query lower bounds of $\Omega(1/\epsilon)$ with no erasures and $\Omega(\log t)$ for fixed-rate adversarial erasures [38, Theorem 1.4]).

Our online linearity tester improves on the tester in [38] both in terms of query complexity and in terms of simplicity of the tester. The classical linearity test of [20] looks for witnesses of nonlinearity consisting of three points x_1, x_2 and $x_1 \oplus x_2$ that satisfy $f(x_1)+f(x_2) \neq f(x_1 \oplus x_2)$. Kalemaj et al. generalized it to witnesses consisting of any even number of points and their XOR. Let $\textsc{XorTest}_k$ 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 $\textsc{XorTest}_k$ 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 $\textsc{XorTest}_k$ can be used in the standard linearity testing setting (without erasures) to obtain another optimal $O(1/\epsilon)$-query tester: $\textsc{XorTest}_k$ can be repeated $O(1/(k\epsilon))$ times to obtain constant probability of error. To the best of our knowledge, only testers based on the BLR test (i.e., $\textsc{XorTest}_2$) have been analyzed 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 $\textsc{XorTest}_k$, 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 [38] 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 [5, 4, 31, 29, 30, 60, 58, 1, 2, 47, 46, 41, 61, 63, 37, 19, 33, 59, 24, 40, 38]. For a $d \in \mathbb{N}$, let $\mathcal{P}_d$ denote the set of all polynomials $p : \{0,1\}^n \to \{0,1\}$ of degree at most d over $\mathbb{F}_2$, that is, functions $p(x)$ that can be represented as a sum of monomials[1] of the form $\prod_{i \in S} x[i]$, where $x = (x[1], \ldots, x[n])$ is a vector of n bits and $|S| \leq d$. In the standard property testing model, $\mathcal{P}_d$ can be ϵ-tested with $O(1/\epsilon + 2^d)$ queries by repeating the following test of Alon et al. [1]: select $d+1$ points in $\{0,1\}^n$ uniformly at random, query all of their linear combinations f, 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. [19], and the matching lower bound of $\Omega(1/\epsilon + 2^d)$ 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.

[1] To 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.

▶ **Theorem 1.2** (Lower bound for low-degree testing). *Fix an integer $d > 1$. Let $\mathcal{P}_d$ be the set of polynomials of degree at most d over $\mathbb{F}_2$. There exists $n_0 = n_0(d)$, such that for all $n \geq n_0$ and $\epsilon \in (0, 1/3]$, every ϵ-tester of functions $f\colon \{0,1\}^n \to \{0,1\}$ for property $\mathcal{P}_d$ that works in the presence of a t-online fixed-rate erasure adversary must make $\Omega\left(\log^d t\right)$ queries.*

Our lower bound is nearly tight in terms of the dependence on t and d: in a concurrent work, Minzer and Zheng [45] 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 ϵ).

To prove our lower bound, we define an extended representation $\mathsf{Ext}_d(x)$ for each vector $x \in \mathbb{F}_2^n$, where each entry of $\mathsf{Ext}_d(x)$ 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 [12, Lemma 1.4] (or, equivalently, [43, 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 (see the full version [15] for more details). 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 $d < n/4$ and $t \log^7 t \leq c \cdot \epsilon^{2.5} 2^{-12d} 2^{n/4}$, there exists an ϵ-tester for property $\mathcal{P}_d$ of functions $f : \{0,1\}^n \to \{0,1\}$ that works in the presence of a $(2^{d-1}, t)$-online erasure budget-managing adversary and makes $O\left(1/\epsilon + 2^{3d}\left(d + \log t\right)^3\right)$ 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 [42] on algebraic property testing. 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 [19], 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 $f : [n] \to \mathbb{R}$. 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 [13].

Before stating our results, we define sortedness, the Lipschitz property, and the general class of local properties. A sequence $f : [n] \to \mathbb{R}$ 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 [28, 27, 53, 18, 22, 11, 51] and the survey in [54]). A sequence $f : [n] \to \mathbb{R}$ is Lipschitz if $|f(x+1) - f(x)| \leq 1$ for all $x \in [n-1]$. The Lipschitz property has been studied in [36, 22, 25, 17, 3, 21] and has applications to data privacy. Both sortedness and the Lipschitz property belong to the class of local properties, defined by [13]. A property $\mathcal{P}$ of sequences $f\colon [n] \to \mathbb{R}$ is *local*[2] if there exists a family $\mathcal{F}$ of *forbidden* pairs $(a, b) \in \mathbb{R}^2$ such that

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

Then 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 $\mathcal{F} = \{(a, b) : |a - b| > 1\}$.

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 $\epsilon > 0$ and $n \geq C/\epsilon$:*

1. *For every local property $\mathcal{P}$ of sequences $f\colon [n] \to \mathbb{R}$, 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 $O\big(\frac{\log(\epsilon n)}{\epsilon}\big)$ queries.*
2. *For every $t \geq C\epsilon$, there is no ϵ-tester for sortedness of sequences $f\colon [n] \to \mathbb{R}$ 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 erasure adversary, the tester has one-sided error.

The positive result in Item 1 of Theorem 1.4 matches the query complexity of the offline optimal tester from [13] while also attaining optimal resilience guarantees.

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!

[2] This class of properties is called 2-local in [13], 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).

For fixed-rate adversaries, the threshold rate is arbitrarily close to 1. The negative result is established in [38]; 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 $f\colon [n] \to \mathbb{R}$, 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 $O\big(\frac{\log(\epsilon n)}{\epsilon}\big)$ 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 $O\big(\frac{\log(\epsilon n)}{\epsilon}\big)$. This matches known lower bounds on the query complexity of sortedness [23, 11]. Our upper bound is the same as that obtained in [13], 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 [21] 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 [13], 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. [26] in an offline model. In the model of Dixit et al., also studied in [57, 55, 14, 52, 44, 49], the adversary performs all erasures to the function before the execution of the algorithm.

As discussed, the online testing model was defined in [38]. In addition to the results already mentioned, [38] gave an online ϵ-tester for quadraticity of functions $f : \{0,1\}^n \to \{0,1\}$ that has query complexity $O(1/\epsilon)$ 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 [45] 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., [32, 48]), 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 [6, 35]. 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 $\{1, 2, \ldots, n\}$ and log to denote logarithm base 2. We denote by $\mathcal{P}_d$ the class of Boolean functions $f : \{0, 1\}^n \to \{0, 1\}$ of degree at most d over $\mathbb{F}_2$.

2.1 Online Testing

We model access to the input with a sequence $\{\mathcal{O}_i\}_{i\in\mathbb{N}}$ of oracles, where $\mathcal{O}_i$ is used to answer the i-th query (or, more generally, the i-th batch of queries). Oracle $\mathcal{O}_1$ gives access to the original input (e.g., when the input is a function f, we have $\mathcal{O}_1 \equiv f$). 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 $\mathrm{Dist}(\mathcal{O}, \mathcal{O}')$ 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 sequence*[3] *of oracles $\mathcal{O} = \{\mathcal{O}_i\}_{i \in \mathbb{N}}$ is induced by a t-*online fixed-rate adversary *if $\mathcal{O}_1$ is equal to the input and, for all $i \in \mathbb{N}$,*

$$Dist(\mathcal{O}_i, \mathcal{O}_{i+1}) \leq \lfloor (i+1) \cdot t \rfloor - \lfloor i \cdot t \rfloor .$$

*A sequence of oracles $\mathcal{O} = \{\mathcal{O}_i\}_{i \in \mathbb{N}}$ is induced by a t-*online budget-managing adversary *if $\mathcal{O}_1$ is equal to the input and, for all $i \in \mathbb{N}$,*

$$Dist(\mathcal{O}_1, \mathcal{O}_{i+1}) \leq i \cdot t.$$

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 $\mathcal{O}_1$ to answer the first b queries and, more generally, oracle $\mathcal{O}_i$ 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 $\mathcal{O} = \{\mathcal{O}_i\}_{i \in \mathbb{N}}$ is induced by an* erasure adversary *if for all $i \in \mathbb{N}$ and data points x,*

$$\mathcal{O}_{i+1}(x) \in \{\mathcal{O}_i(x), \perp\}.$$

A corruption adversary *can change answers to anything in the range, i.e., $\mathcal{O}_i$ 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 $dist(f, \mathcal{P})$, 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 $dist(f, \mathcal{P}) \geq \epsilon$. 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 to an input function f via a sequence of oracles $\mathcal{O} = \{\mathcal{O}_i\}_{i \in \mathbb{N}}$ induced by that type of adversary. For all adversarial strategies of the specified type,*

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,*

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

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.

To ease notation, we use $\mathcal{O}(x)$ for the oracle's answer to query x (omitting the timestamp i). If x was queried multiple times, $\mathcal{O}(x)$ 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 a part of our main algorithm. Since it uses Fourier analysis, here we use the standard notation switch for the value returned by a Boolean function (see, e.g., [50]), where true is denoted by -1 and false is denoted by 1. As a result, the input function is of the form $f\colon \{0,1\}^n \to \{-1,1\}$, and it is linear if $f(x_1) \cdot f(x_2) = f(x_1 \oplus x_2)$ for all $x_1, x_2 \in \{0,1\}^n$.

3.1 An Offline Linearity Test

We first define $\textsc{XorTest}_k$, introduced by [38], which naturally extends the BLR test.

Algorithm 1 $\textsc{XorTest}_k$.

Input: Even integer parameter $k \geq 2$ and query access to a function $f : \{0,1\}^n \to \{-1,1\}$

1. Query k points $x_1, \ldots, x_k \in \{0,1\}^n$ chosen uniformly at random (with replacement).
2. Query point $y = \bigoplus_{i\in[k]} x_i$.
3. **Accept** if $f(y) = \prod_{i\in[k]} f(x_i)$ (equivalently, if $f(y) \cdot \prod_{i\in[k]} f(x_i) = 1$); otherwise, **reject**.

The test always accepts all linear functions, as can be shown by induction on k. The next lemma demonstrates that $\textsc{XorTest}_k$ rejects functions that are ϵ-far from linear with sufficient probability. This is a strengthened version of [38, Theorem 1.2] that bounds the rejection probability by ϵ.

▶ **Lemma 3.1.** *If f is ϵ-far from linear, and $k \geq 2$ is even, then*

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

Since $k \geq 2$ and $\epsilon \in (0, 1/2]$, we get $(1-2\epsilon)^{k-1} \leq 1-2\epsilon$, with the first inequality in the lemma yielding the bound proved in [38]. 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 repeating Algorithm 1; this can be done with any value of k from 3 to $O(1/\epsilon)$.

Proof of Lemma 3.1. The key tool used in the proof is Fourier analysis (see, e.g., [50] for an overview of the technique and standard facts). We start by giving a couple of standard definitions.

The *character* functions $\chi_S\colon \{0,1\}^n \to \{-1,1\}$, 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 $\langle g, h\rangle = \mathop{\mathbb{E}}_{x\sim\{0,1\}^n}[g(x)h(x)]$, where $g, h\colon \{0,1\}^n \to \mathbb{R}$. For $g\colon \{0,1\}^d \to \mathbb{R}$ and $S \subseteq [n]$, the Fourier coefficient of g on S is $\widehat{g}(S) = \langle g, \chi_S\rangle = \mathop{\mathbb{E}}_{x\sim\{0,1\}^n}[g(x)\chi_S(x)]$.

Now consider a function $f : \{0,1\}^n \to \{-1,1\}$ 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

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

As shown in [38, Equation (4)],

$$\begin{aligned}\Pr\left[\textsc{XorTest}_k(f) \textit{ rejects}\right] &= \mathop{\mathbb{E}}_{x_1,\ldots,x_k\sim\{0,1\}^n}\left[\frac{1}{2} - \frac{1}{2}\prod_{i\in[k]} f(x_i)\cdot f\left(\oplus_{i=1}^k x_i\right)\right] \\ &= \frac{1}{2} - \frac{1}{2}\sum_{S\subseteq[n]} \widehat{f}(S)^{k+1}. \end{aligned} \tag{2}$$

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

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

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. ◀

3.2 Online-Erasure-Resilient Linearity Tester

Our algorithm (Algorithm 2) is based on multiple simulations of $\textsc{XorTest}_k$. Each simulation starts by querying a reserve of $m = 2k$ initial points to keep many possibilities open for the final $\textsc{XorTest}_k$ query. Specifically, a reserve of $2k$ random points creates roughly 2^{2k} different options for the final $\textsc{XorTest}_k$ 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 $\textsc{XorTest}_k$ 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 $m = \Theta(\log t + 1/\epsilon)$ and the number of iterations, r, to $\Theta(1)$ is enough for the desired

[4] In [38], the reserve is used to simulate $\textsc{XorTest}_k$ 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.

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: $\log t + o(\log t) + \Theta(\log(1/\epsilon))$, and the number of iterations, r, depends on ϵ and m. Crucially, in each iteration, the distribution

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 $m \leftarrow 4\left\lceil \frac{1}{4}\left(14 + \log t + \log\log^2 t + \log \frac{1}{\epsilon^2}\right)\right\rceil$, $\alpha \leftarrow \min\left\{\frac{1}{4}, \frac{m\epsilon}{4}\right\}$ and, $r \leftarrow \left\lceil \frac{5}{4\alpha}\right\rceil$.

3 **repeat** r **times**

4 Sample $X = (x_1, \ldots, x_m) \in (\{0,1\}^n)^m$ uniformly at random.

5 Query f at points $x_1, \ldots, x_m$.

6 Query f at point $y = \oplus_{j \in S} x_j$, where S is a uniformly random subset of $[m]$ of size $\frac{m}{2}$.

7 **if** $\mathcal{O}(y) \cdot \Pi_{j \in S} \mathcal{O}(x_j) = -1$ **then**

8 **Reject** ; `// This implies no erasures in this iteration.`

9 **end**

10 **Accept**

over the queries made by Algorithm 2 is identical to that in the $\textsc{XorTest}_k$ test. Indeed, as S and X are independent, we could draw S first. Then, 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 $\textsc{XorTest}_k$.

The second ingredient we will 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 $3\alpha/25$.*

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

Proof of Theorem 1.1. 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 and since $k = m/2$ in Algorithm 2 is even, 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 $r = \left\lceil \frac{5}{4\alpha}\right\rceil \geq \frac{5}{4\alpha}$ 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 $m \leq 20(\log t + 1/\epsilon)$.

To show that Algorithm 2 is also corruption-resilient, one may apply [38, Lemma 1.8], which essentially says an error-resilient algorithm that has probability at most 1/3 to either err or see a manipulation, 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

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

▶ Remark. Our tester is applicable for $t \leq \text{poly}(\epsilon) \cdot 2^{n/2}$. On the other hand, it can be easily shown that for $t = \Omega\left(\epsilon 2^n\right)$ testing is impossible. This follows from the fact that there exists some constant c such that c/ϵ queries are not enough, and by then the adversary can already manipulate all other entries. We leave it as an open question to understand for which values of t linearity can be tested in the online setting. Such questions are investigated for other properties 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 [38, 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:

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

Proof. We first bound m using its setting in Algorithm 2 and the premise $t \cdot \log^2 t \leq 2^{-21} \cdot \epsilon^{2.5} 2^{n/2}$ from Theorem 1.1:

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

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,

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

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 (3) to get

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

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

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 $\mathcal{O}(x_j) = \perp$ 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

$$\Pr_X[B_1] \leq \frac{tr(m+1)m}{2^n} \leq \frac{tr(m+1)m}{2^{2m}}. \tag{4}$$

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 choice $y \in \{0,1\}^n$. By Claim 3.3, $\Pr[B_2] \leq \frac{\alpha}{25}$.

An erasure on query y. Let B_3 be the event that $\mathcal{O}(y) = \perp$ 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 $\binom{m}{m/2} \geq 2^m/\sqrt{2m}$, to obtain

$$\Pr[B_3|\overline{B_2}] \leq \frac{tr(m+1)}{\binom{m}{m/2}} \leq \frac{tr(m+1)\sqrt{2m}}{2^m}. \tag{5}$$

In terms of the bad events, the lemma states that $\Pr[B_1 \cup B_3] \leq 3\alpha/25$. 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

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

Since $\Pr[B_2] \leq \frac{\alpha}{25}$, it remains to show that $(\star)$ is at most $\frac{2\alpha}{25}$, in order to complete the proof. To do this, we combine the bounds from (4) and (5) to bound $(\star)$:

$$\Pr[B_1] + \Pr[B_3|\overline{B_2}] \leq \frac{tr(m+1)m}{2^{2m}} + \frac{tr(m+1)\sqrt{2m}}{2^m} \leq \frac{trm^2}{2^{m+1}} \leq \frac{rm^2\epsilon^2}{2^{15}\log^2 t}. \tag{6}$$

For the second inequality, we use the fact that $\frac{(m+1)m}{2^m} + (m+1)\sqrt{2m} \leq \frac{m^2}{2}$ for all $m \geq 10$. For the last inequality, we use $2^m \leq \frac{2^{14} t \log^2 t}{\epsilon^2}$ which is derived from the value of m in Algorithm 2. To further bound (6), we split the analysis into two cases, depending on the value of $\alpha = \min\{1/4, m\epsilon/4\}$.

Case I: $\alpha = \frac{m\epsilon}{4}$. In this case, there are $r = \lceil \frac{5}{m\epsilon} \rceil \leq \frac{5}{m\epsilon} + 1$ iterations, which means that $rm\epsilon \leq 5 + m\epsilon \leq 6$. Therefore,

$$\Pr[B_1] + \Pr[B_3|\overline{B_2}] \leq \frac{(rm\epsilon)m\epsilon}{2^{15}\log^2 t} \leq \frac{6m\epsilon}{2^{15}\log^2 t} \leq \frac{m\epsilon}{50} = \frac{2\alpha}{25}.$$

Case II: $\alpha = \frac{1}{4}$. For this case, note that

$$m \le 18 + \log t + 2\log\log t + 2\log(1/\epsilon) \le 18 + 2.1\log t + 1.1/\epsilon \le 11.2 \cdot \max\{2\log t, 1/\epsilon\},$$

where the second inequality uses $2\log y \le 1.1y$ for all $y > 0$ (with $y = 1/\epsilon$ and $y = \log t$), and the last inequality uses $2\log t \ge 2$ and $1/\epsilon \ge 2$. It follows that

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

Finally, we use $r = \lceil \frac{5}{4\alpha} \rceil = 5$ and (6) to obtain

$$\Pr[B_1] + \Pr[B_3|\overline{B_2}] \le \frac{r}{2^{13}}\left(\frac{m\epsilon}{2\log t}\right)^2 \le \frac{5 \cdot (5.7)^2}{2^{13}} \le \frac{1}{50} = \frac{2\alpha}{25}.$$ ◀

4 The Lower Bound for Low-Degree Testing

In this section, we prove our lower bound for online low-degree testing stated in Theorem 1.2.

Proof of Theorem 1.2. Fix a degree $d > 1$. Let $\binom{n}{\le d}$ denote $\sum_{i=0}^{d}\binom{n}{i}$. For a vector $x \in \mathbb{F}_2^n$, define its extension $\mathsf{Ext}_d(x) \in \mathbb{F}_2^{\binom{n}{\le d}}$ to be the vector where each entry is indexed by a set $S \subseteq [n]$ of at most d coordinates and $\mathsf{Ext}_d(x)_S = \prod_{i\in S} x_i$. In other words, each entry of $\mathsf{Ext}_d(x)$ is an evaluation of one monomial of degree at most d on input x. The extended vector $\mathsf{Ext}_d(x)$ 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., $\pi(\mathsf{Ext}_d(x)) = x$.

Consider the following strategy $\mathcal{S}$ for a fixed-rate erasure adversary. Suppose the tester has successfully obtained answers to k distinct queries $y_1, \ldots, y_k$. Consider the linear space in $\mathbb{F}_2^{\binom{n}{\le d}}$ spanned by $\mathsf{Ext}_d(y_1), \ldots, \mathsf{Ext}_d(y_k)$, and let $Z \subseteq \mathbb{F}_2^n$ 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 $\binom{\lfloor \log r \rfloor}{d} \le k$.

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: $rk(A) \le k$. 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 $\mathsf{Ext}_d(z)$. By definition, every column in A' is spanned by the k columns indexed by $y_1, \ldots, y_k$, showing that $rk(A') \le k$.

Lower bound: $rk(A) \ge \binom{\lfloor \log r \rfloor}{d}$. Let B be an arbitrary submatrix of A with only $r' \le r$ columns, where $r' = 2^a$ is the largest power of 2 not exceeding r. Trivially, $rk(A) \ge rk(B)$. We now apply [12, Lemma 1.4] (or [43, Theorem 1.5]) to see that the dimension of the rows of B is at least

$$\binom{a}{\le d} \ge \binom{a}{d} = \binom{\lfloor \log r \rfloor}{d}.$$

The two inequalities together yield the claim. ◁

Now we apply Yao's minimax principle for the online model [38, 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 $\frac{6}{7}$, 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 $\frac{1}{3}$-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 $\mathrm{dist}(f,g)$ be the fraction of domain points on which f and g differ. Then, $\mathbb{E}[dist(f,g)] = 1/2$. By the Hoeffding bound, $\Pr_{f\sim\mathcal{D}^-}[\mathrm{dist}(f,g) \leq \frac{1}{3}] \leq e^{-\frac{2^n}{18}}$. By a union bound over the $2^{\binom{n}{\leq d}}$ functions of degree at most d, we get $\Pr_{f\sim\mathcal{D}^-}[dist(f,\mathcal{P}_d) \geq \frac{1}{3}] \geq 1 - 2^{\binom{n}{\leq d}} \cdot e^{-\frac{2^n}{18}}$. For large enough n, this probability is at least $6/7$.

Let A be a deterministic algorithm that makes $q \leq \binom{\lfloor \log t \rfloor - 1}{d}$ queries to f via 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 $a_1, \ldots, a_{k-1}$ to distinct queries $y_1, \ldots, y_{k-1}$. When $f \sim \mathcal{D}^-$, we have $\Pr_{f\sim\mathcal{D}^-}[f(y_k) = a_k \mid f(y_1) = a_1 \wedge \cdots \wedge f(y_{k-1}) = a_{k-1}] = 1/2$ for all $a_1, \ldots, a_k \in \{0,1\}$, 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 $1_f \in \{0,1\}^{\binom{n}{\leq d}}$, where each entry corresponds to a monomial of degree at most d. For any query it holds that $f(y_i) = \langle 1_f, \mathsf{Ext}_d(y_i)\rangle$. By definition of $\mathcal{S}$, the extension $\mathsf{Ext}_d(y_k)$ is linearly independent of $\mathsf{Ext}_d(y_1), \ldots, \mathsf{Ext}_d(y_{k-1})$, which implies that for any fixed values $b_1, \ldots b_{k-1}$ of $\langle 1_f, \mathsf{Ext}_d(y_i)\rangle, \ldots, \langle 1_f, \mathsf{Ext}_d(y_{k-1})\rangle$, the value of $\langle 1_f, \mathsf{Ext}_d(y_k)\rangle$ is still uniformly distributed.

Thus, $\mathcal{D}^+$ generates degree-d polynomials, $\mathcal{D}^-$ generates functions that are $\frac{1}{3}$-far from $\mathcal{P}_d$ with probability at least $\frac{6}{7}$, and the query-answer histories for any deterministic algorithm A that makes $q \leq \binom{\lfloor \log t \rfloor - 1}{d}$ 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. ◀

5 Low-Degree Testing

This section is dedicated to the proof of Theorem 1.3 that gives an online tester for the property $\mathcal{P}_d$ (being a polynomial of degree at most d) with batches of size $b = 2^{d-1}$. All missing proofs and details appear in the full version [15].

Our strategy is a natural extension of the one employed for linearity in Section 3.2: we view the $\textsc{XorTest}_2$ as a "linear square" $(x, y, x+y$, omitting the origin), and perceive the extended test as a "chain of squares" (see the top part of Figure 1). For quadraticity and any degree $d > 2$, we resort to "chains of cubes" in a similar way (see the bottom part of Figure 1).

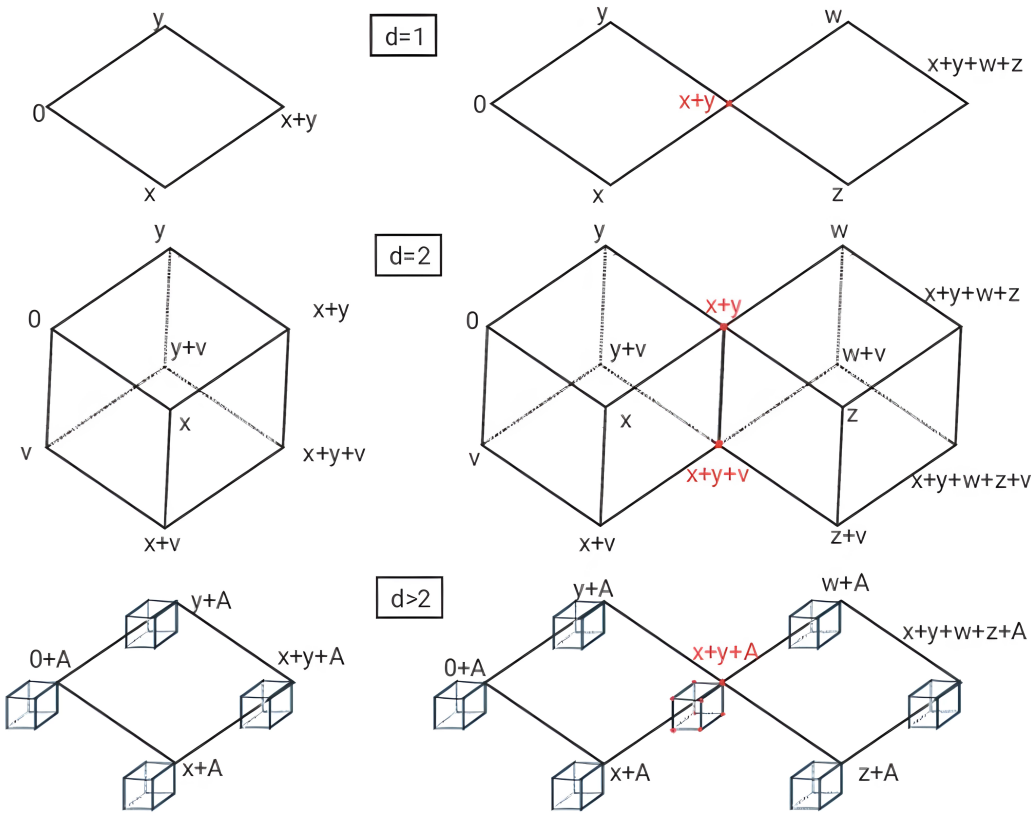

Figure 1 Chains of cubes (with 0 as the first parameter). The left side shows linear subspaces (drawn as cubes); the right side shows chains formed by two cubes. Red points overlap and cancel each other. From top to bottom, we have witnesses for linearity (chains of squares), witnesses for quadraticity (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.

5.1 Algebraic Testing Patterns

In the following, 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 $x, y, x+y$). Setting a value to all parameters induces an *instance* of the testing pattern. To test for a low-degree property $\mathcal{P}_d$, we simply draw a random instance of some "good" pattern and check the parity of the sum of function values on the points in the pattern. This is a generalization of the tester of [1], which checks the parity on a random linear subspace (which is an example of a good testing pattern).

Our notion of a "good" testing pattern is a special case of the more abstract notion of *k-local formal characterizations* (see [42, Definition 2.3]), where each linear map defines a single testing point, and the subspace V of "good answer vectors" is fixed to all answer vectors with parity 0. In particular, a good testing pattern is also a 2-ary independent characterization (see [42, Definition 2.5]).

▶ **Definition 5.1** (Testing pattern, parameter, instance). *A* testing pattern *is a matrix*[5] $X \in \mathcal{M}_{\ell\times m}(\mathbb{F}_2)$ *with rows* $c_1, \ldots, c_\ell$*, such that* $\ell \geq m$*, each row is unique, and* $rank(X) = m$*. 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* $M \in \mathcal{M}_{m\times n}(\mathbb{F}_2)$*. The rows of* XM *specify* ℓ *testing points* $y_1, \ldots, y_\ell$ *in* $\mathbb{F}_2^n$*.*

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

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

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

Each testing pattern induces a parity-checking tester, defined below.

Algorithm 3 An X-tester.

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

1 Choose a parameter matrix $M \in \mathcal{M}_{m \times n}(\mathbb{F}_2)$ uniformly at random (alternatively, choose m row vectors $a_1, \ldots, a_m \in \mathbb{F}_2^n$ independently and uniformly at random).

2 Query f at testing points $y_1, \ldots, y_\ell$, the ℓ row vectors of the instance matrix XM.

3 **Accept** if $T_{X,f}(a_1, \ldots, a_m) = 0$; otherwise, **reject**.

In the full version, we define a good testing pattern for a property $\mathcal{P}$. Essentially, a good pattern $X_{\ell \times m}$ for a property $\mathcal{P}$ characterizes $\mathcal{P}$:

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

5.2 Generalized Witnesses for $\mathcal{P}_d$

The tester of [1], discussed in Section 1.1.2, queries a linear $(d+1)$-dimensional subspace; the variant of the tester considered in [19] uses an affine $(d+1)$-dimensional subspace. Both subspaces can be represented as cubes[6]. The affine version of the tester is equivalent to X-tester where the pattern X has $d+1$ parameters $a_1, \ldots, a_{d+1}$ representing directions and a translation parameter a_{d+2}. The rows of pattern X are all different binary vectors of $d+2$ bits with value 1 at the last coordinate. It is easy to verify this pattern is good for $\mathcal{P}_d$. (For the tester based on a linear subspace simply delete the last column of X, which corresponds to fixing the last parameter a_{d+2} to $\vec{0}$.) For the special case of affinity ($d = 1$), this pattern can be equivalently viewed as taking parameters x_1, x_2, x_3 to testing points $\{x_1, x_2, x_3, x_1 + x_2 + x_3\}$ (the BLR linearity test simply fixes $x_3 = 0$), and the generalized witness used in Algorithm 1 can be viewed as a "chain of squares": the second square shares the point $x_1 + x_2 + x_3$ as if it was its first parameter, and gets additional parameters x_4, x_5. When we check the parity of f on all points in both squares, the intermediate point $x_1 + x_2 + x_3$ appears twice and cancels out. We are left with testing points $x_1, \ldots, x_5$ and their sum $\sum_{i \in [5]} x_i$. To generalize this and test for higher degree, we consider the similar picture with high-dimensional cubes. Alternatively, we can view this as the chain from linearity, but each point is replaced with an entire $(d-1)$-dimensional cube (see Figure 1). Note the parameters of these patterns have two different roles, some parameters impose the chain structure while others are used for the cubical structure. We dub these patterns "chains of cubes".

▶ **Definition 5.2** (Chain of Cubes Pattern). *For degree d and odd integer s (length of the chain), we define the* chain of cubes pattern $\chi_{d,s}$. *It takes parameters $a_1, \ldots, a_d$ for the cube structure, and $a_{d+1}, \ldots, a_{d+s}$ for the chain, and outputs $(s+1) \cdot 2^{d-1}$ 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.*

[6] We call them cubes for easy visualisation, even though most pattern instances are not parallel to the axes.

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

▷ Claim 5.3. 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.

We next show a soundness result for any pattern, and in particular for our chain of cubes pattern, $\chi_{d,s}$. We resort to two previously known results and combine them together. The first is inherited by the soundness guarantee in [42] for any 2-ary local characterization.

The other is a specialized argument for affine subsapces that appears in [19]. The argument deals with small values of ϵ (so most violating instances are such due to a single point). We generalize this argument for any pattern, relying on two properties held by all patterns. First, each two induced testing points, over a random instance, are independent. Second, the tester depends on the parity of all answers to the queried points.

In the full version, we prove the following lemma.

▶ **Lemma 5.4.** *Fix any pattern $X \in \mathcal{M}_{\ell \times m}(\mathbb{F}_2)$ with $\ell \geq 3$. For any function f that is ϵ-far from $\mathcal{P}_d$, we have*

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

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

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}

$t \leftarrow \max\{t, 2\}$; `// If` $t < 2$`, replace it with` $t = 2$.

$m \leftarrow 2\left\lceil \log\left(\frac{2^{20}2^d t^{1/4}(d\log t)^{3/2}}{\epsilon^{1/2}}\right)\right\rceil + 1$, $\alpha \leftarrow \min\left\{\frac{\epsilon\ell}{2}, \frac{1}{2\ell^2}\right\}, r \leftarrow \left\lceil \frac{2}{\alpha}\right\rceil$.

repeat r **times**

  Sample $V = (v_1, \ldots, v_{d-1}) \in (\{0,1\}^n)^{d-1}$ uniformly at random.

  Sample $X = (x_1, \ldots, x_{2m}) \in (\{0,1\}^n)^{2m}$ uniformly at random.

  for each $i \in [2m]$: batch query f at points $x_i + \sum_{j \in T} v_j$ for all $T \subseteq [d-1]$.

  Sample uniformly at random a subset of $[2m]$ of size m, denoted $S = \{s_1, \ldots, s_m\}$.

  Set $y = \oplus_{j \in S} x_j$ and batch query f at points $y + \sum_{j \in T} v_j$ for all $T \subseteq [d-1]$.

  if $T_{\chi_{d-1,m},\mathcal{O}}(v_1, \ldots, v_{d-1}, x_{s_1}, \ldots, x_{s_m}) = 1$ **then**

    Reject ; `// This implies no erasures in this iteration.`

  end

Accept

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 $\textsc{XorTest}_k$ in Algorithm 2. The number of iterations and the length of the chain, m, are chosen according to the soundness guarantee of these generalized patterns[7].

[7] Improving the soundness guarantee in Lemma 5.4 to $\Theta(\min\{\ell\epsilon, 1\})$ would allow a better choice m, r, similar to Algorithm 4, and result in a better query complexity.

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. 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 $[n]$ by[8]

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

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 [13].

▶ **Definition 6.1** (Unrepairability). *Let $\mathcal{P}$ be a local property of sequences $f\colon [n] \to \mathbb{R}$ 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 every sequence $f'\colon [n] \to \mathbb{R}$, 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.

Algorithm 5 Pair tester for local properties.

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

```
Set ℓ = ⌊log(εn/4)⌋.
repeat 200 log(εn)/ε times
    Sample i ∈ [0 : ℓ] uniformly at random,
    Sample (x, y) ∈ D_{2^i}(n) uniformly at random; query f(x) and f(y).
    if f is unrepairable on {x, y} then
        Reject
    end
Accept
```

We first show that the pair tester works for every local property $\mathcal{P}$ in the *offline* property testing model. We state the following lemma, which is proved in the full version [15].

▶ **Lemma 6.2** (Offline correctness of pair tester). *Algorithm 5 is a nonadaptive ϵ-tester with one-sided error probability bounded by $1/7$ 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.

[8] Note that the definition is cyclic, in the sense that 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.

▶ **Lemma 6.3** (Pair tester, batch size 1). *Fix a local property $\mathcal{P}$, $\epsilon > 0$, $n \in \mathbb{N}$, and $t \leq \epsilon/10^6$. The probability that Algorithm 5, when run in the online erasure model with parameters $\mathcal{P}, \epsilon, n, t$ and batch size $b = 1$, queries an erased element is at most $1/7$.*

▶ **Lemma 6.4** (Pair tester, batch size 2). *Fix a local property $\mathcal{P}$, $\epsilon > 0$, $n \in \mathbb{N}$, and $t \leq \frac{\epsilon^2 n}{3 \cdot 10^5 \log^2(\epsilon n)}$. The probability that Algorithm 5, when run in the online erasure model with parameters $\mathcal{P}, \epsilon, n, t$ and batch size $b = 2$, queries an erased element is at most $1/7$.*

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 deferred to the full version.

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

$$\Pr(A \setminus B) \geq \Pr(A) - \Pr(B) \geq \frac{6}{7} - \frac{1}{7} > \frac{2}{3},$$

where the second inequality follows from Lemmas 6.2 and 6.3.

For a corruption adversary, we use the same argument combined with [38, Lemma 1.8]. ◀

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. ◀

We conclude by stating our main result for *fixed-rate* adversaries, showing that the threshold rate is arbitrarily close to 1. The proof appears in the full version. A matching negative result is established in [38], see Theorems 1.5 and 1.7 there.

▶ **Proposition 6.5.** *Fix $\epsilon > 0$, $t < 1$ and a local property $\mathcal{P}$ of sequences $f\colon [n] \to \mathbb{R}$, where $n \geq \frac{C\left(\log \frac{1}{\epsilon} + \log \frac{1}{1-t}\right)^2}{\epsilon^2(1-t)}$ 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.*

References

1. Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. *IEEE Transactions on Information Theory*, 51(11):4032–4039, 2005. doi:10.1109/TIT.2005.856958.
2. Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003. doi:10.1007/s00493-003-0025-0.
3. Pranjal Awasthi, Madhav Jha, Marco Molinaro, and Sofya Raskhodnikova. Testing Lipschitz functions on hypergrid domains. *Algorithmica*, 74(3):1055–1081, 2016. doi:10.1007/s00453-015-9984-y.
4. László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 21–31, 1991. doi:10.1145/103418.103428.
5. László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991. doi:10.1007/BF01200056.

6 József Beck. *Combinatorial Games: Tic-Tac-Toe Theory.* Cambridge: Cambridge University Press, 2008.

7 Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996. doi:10.1109/18.556674.

8 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability-towards tight results. *SIAM Journal on Computing (SICOMP)*, 27(3):804–915, 1998. doi:10.1137/S0097539796302531.

9 Mihir Bellare, Shafi Goldwasser, Carsten Lund, and Alexander Russell. Efficient probabilistically checkable proofs and applications to approximations. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 294–304, 1993. doi:10.1145/167088.167174.

10 Mihir Bellare and Madhu Sudan. Improved non-approximability results. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 184–193, 1994. doi:10.1145/195058.195129.

11 Aleksandrs Belovs. Adaptive lower bound for testing monotonicity on the line. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 31:1–31:10, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.31.

12 Ido Ben-Eliezer, Rani Hod, and Shachar Lovett. Random low-degree polynomials are hard to approximate. *Computational Complexity*, 21(1):63–81, 2012.

13 Omri Ben-Eliezer. Testing local properties of arrays. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019*, pages 11:1–11:20, 2019.

14 Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Ron D. Rothblum. Hard properties with (very) short PCPPs and their applications. In *Proceedings, Innovations in Theoretical Computer Science (ITCS)*, pages 9:1–9:27, 2020. doi:10.4230/LIPIcs.ITCS.2020.9.

15 Omri Ben-Eliezer, Esty Kelman, Uri Meir, and Sofya Raskhodnikova. Property testing with online adversaries, 2023. arXiv:2311.16566.

16 Eli Ben-Sasson, Madhu Sudan, Salil P. Vadhan, and Avi Wigderson. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 612–621, 2003. doi:10.1145/780542.780631.

17 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 164–173, 2014. doi:10.1145/2591796.2591887.

18 Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM Journal on Computing (SICOMP)*, 41(6):1380–1425, 2012. doi:10.1137/110826655.

19 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of Reed-Muller codes. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 488–497, 2010. doi:10.1109/FOCS.2010.54.

20 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-W.

21 Deeparnab Chakrabarty, Kashyap Dixit, Madhav Jha, and C. Seshadhri. Property testing on product distributions: Optimal testers for bounded derivative properties. *ACM Transactions on Algorithms (TALG)*, 13(2):20:1–20:30, 2017. doi:10.1145/3039241.

22 Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 419–428, 2013. doi:10.1145/2488608.2488661.

23 Deeparnab Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10:453–464, 2014. doi:10.4086/toc.2014.v010a017.

24 Irit Dinur and Venkatesan Guruswami. PCPs via low-degree long code and hardness for constrained hypergraph coloring. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 340–349, 2013. doi:10.1109/FOCS.2013.44.

25 Kashyap Dixit, Madhav Jha, Sofya Raskhodnikova, and Abhradeep Thakurta. Testing the Lipschitz property over product distributions with applications to data privacy. In *Theory of Cryptography Conference (TCC)*, pages 418–436, 2013. doi:10.1007/978-3-642-36594-2_24.

26 Kashyap Dixit, Sofya Raskhodnikova, Abhradeep Thakurta, and Nithin Varma. Erasure-resilient property testing. *SIAM Journal on Computing (SICOMP)*, 47(2):295–329, 2018. doi:10.1137/16M1075661.

27 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 97–108, 1999. doi:10.1007/978-3-540-48413-4_10.

28 Funda Ergün, Sampath Kannan, Ravi Kumar, Ronitt Rubinfeld, and Mahesh Viswanathan. Spot-checkers. *Journal of Computer and System Sciences*, 60(3):717–751, 2000. doi:10.1006/jcss.1999.1692.

29 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. doi:10.1145/226643.226652.

30 Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Third Israel Symposium on Theory of Computing and Systems (ISTCS)*, pages 190–198, 1995. doi:10.1109/ISTCS.1995.377032.

31 Peter Gemmell, Richard J. Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 32–42, 1991. doi:10.1145/103418.103429.

32 Oded Goldreich and Dana Ron. On learning and testing dynamic environments. *Journal of the ACM (JACM)*, 64(3):1–90, 2017.

33 Elad Haramaty, Amir Shpilka, and Madhu Sudan. Optimal testing of multivariate polynomials over small prime fields. *SIAM Journal on Computing (SICOMP)*, 42(2):536–562, 2013. doi:10.1137/120879257.

34 Johan Håstad and Avi Wigderson. Simple analysis of graph tests for linearity and PCP. *Random Structures and Algorithms*, 22(2):139–160, 2003. doi:10.1002/rsa.10068.

35 Dan Hefetz, Michael Krivelevich, Miloš Stojaković, and Tibor Szabó. *Positional Games*. Oberwolfach Seminars. Vol. 44. Basel: Birkhäuser Verlag GmbH, 2014.

36 Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of Lipschitz functions with applications to data privacy. *SIAM Journal on Computing (SICOMP)*, 42(2):700–731, 2013. doi:10.1137/110840741.

37 Charanjit S. Jutla, Anindya C. Patthak, Atri Rudra, and David Zuckerman. Testing low-degree polynomials over prime fields. *Random Structures and Algorithms*, 35(2):163–193, 2009. doi:10.1002/rsa.20262.

38 Iden Kalemaj, Sofya Raskhodnikova, and Nithin Varma. Sublinear-time computation in the presence of online erasures. *Theory of Computing*, 19(1):1–48, 2023. URL: http://theoryofcomputing.org/articles/v019a001/.

39 Tali Kaufman, Simon Litsyn, and Ning Xie. Breaking the epsilon-soundness bound of the linearity test over GF(2). *SIAM Journal on Computing (SICOMP)*, 39(5):1988–2003, 2010. doi:10.1137/080715548.

40 Tali Kaufman and Dor Minzer. Improved optimal testing results from global hypercontractivity. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 98–109. IEEE, 2022.

41 Tali Kaufman and Dana Ron. Testing polynomials over general fields. *SIAM Journal on Computing (SICOMP)*, 36(3):779–802, 2006. doi:10.1137/S0097539704445615.

42 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 403–412, 2008.

43 Peter Keevash and Benny Sudakov. Set systems with restricted cross-intersections and the minimum rank ofinclusion matrices. *SIAM Journal on Discrete Mathematics*, 18(4):713–727, 2005.

44 Amit Levi, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Erasure-resilient sublinear-time graph algorithms. In *Proceedings, Innovations in Theoretical Computer Science (ITCS)*, pages 80:1–80:20, 2021. doi:10.4230/LIPIcs.ITCS.2021.80.

45 Dor Minzer and Kai Zheng. Adversarial low degree testing. *arXiv*, 2308.15441, 2023. arXiv:2308.15441.

46 Dana Moshkovitz. Low-degree test with polynomially small error. *Computational Complexity*, 26(3):531–582, 2017. doi:10.1007/s00037-016-0149-4.

47 Dana Moshkovitz and Ran Raz. Sub-constant error low degree test of almost-linear size. *SIAM Journal on Computing (SICOMP)*, 38(1):140–180, 2008. doi:10.1137/060656838.

48 Yonatan Nakar and Dana Ron. Testing Dynamic Environments: Back to Basics. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 98:1–98:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2021.98.

49 Ilan Newman and Nithin Varma. New sublinear algorithms and lower bounds for LIS estimation. In *Proceedings, International Colloquium on Automata, Languages and Programming (ICALP)*, pages 100:1–100:20, 2021. doi:10.4230/LIPIcs.ICALP.2021.100.

50 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions.

51 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Parameterized property testing of functions. *ACM Transactions on Computation Theory*, 9(4):17:1–17:19, 2018. doi:10.1145/3155296.

52 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of boolean functions. *Random Struct. Algorithms*, 60(2):233–260, 2022. doi:10.1002/rsa.21029.

53 Sofya Raskhodnikova. Monotonicity testing. *Masters Thesis, MIT*, 1999.

54 Sofya Raskhodnikova. Testing if an array is sorted. *Encyclopedia of Algorithms*, pages 2219–2222, 2016. doi:10.1007/978-1-4939-2864-4_700.

55 Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma. Erasures versus errors in local decoding and property testing. *Random Structures and Algorithms*, 59(4):640–670, 2021. doi:10.1002/rsa.21031.

56 Sofya Raskhodnikova and Ronitt Rubinfeld. Linearity testing/testing Hadamard codes. In *Encyclopedia of Algorithms*, pages 1107–1110. Springer, 2016. doi:10.1007/978-1-4939-2864-4_202.

57 Sofya Raskhodnikova and Nithin Varma. Brief announcement: Erasure-resilience versus tolerance to errors. In *Proceedings, International Colloquium on Automata, Languages and Programming (ICALP)*, pages 111:1–111:3, 2018. doi:10.4230/LIPIcs.ICALP.2018.111.

58 Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 475–484, 1997. doi:10.1145/258533.258641.

59 Noga Ron-Zewi and Madhu Sudan. A new upper bound on the query complexity of testing generalized Reed-Muller codes. *Theory of Computing*, 9:783–807, 2013. doi:10.4086/toc.2013.v009a025.

60 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing (SICOMP)*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

61 Alex Samorodnitsky. Low-degree tests at large distances. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 506–515, 2007. doi:10.1145/1250790.1250864.

62 Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 191–199, 2000. doi:10.1145/335305.335329.

63 Alex Samorodnitsky and Luca Trevisan. Gowers uniformity, influence of variables, and PCPs. *SIAM Journal on Computing (SICOMP)*, 39(1):323–360, 2009. doi:10.1137/070681612.

64 Amir Shpilka and Avi Wigderson. Derandomizing homomorphism testing in general groups. *SIAM Journal on Computing (SICOMP)*, 36(4):1215–1230, 2006. doi:10.1137/S009753970444658X.

65 Madhu Sudan and Luca Trevisan. Probabilistically checkable proofs with low amortized query complexity. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 18–27, 1998. doi:10.1109/SFCS.1998.743425.

66 Luca Trevisan. Recycling queries in PCPs and in linearity tests (extended abstract). In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 299–308, 1998. doi:10.1145/276698.276769.