THEORY OF COMPUTING, Volume 19 (1), 2023, pp. 1–48
www.theoryofcomputing.org

Sublinear-Time Computation in the Presence of Online Erasures

Iden Kalemaj* Sofya Raskhodnikova† Nithin Varma

Received June 13, 2022; Revised June 29, 2023; Published August 23, 2023

Abstract. We initiate the study of sublinear-time algorithms that access their input via an online adversarial erasure oracle. After answering each input query, such an oracle can erase t input values. Our goal is to understand the complexity of basic computational tasks in extremely adversarial situations, where the algorithm's access to data is blocked during the execution of the algorithm in response to its actions. Specifically, we focus on property testing in the model with online erasures. We show that two fundamental properties of functions, linearity and quadraticity, can be tested for constant t with asymptotically the same complexity as in the standard property testing model. For linearity testing, we prove tight bounds in terms of t, showing that the query complexity is $\Theta(\log t)$. In contrast to linearity and quadraticity, some other properties, including sortedness and the Lipschitz property of sequences, cannot be tested at all, even for $t = 1$. Our investigation leads to a deeper understanding of the structure of violations of linearity and other widely studied properties. We also consider implications of our results for algorithms that are resilient to online adversarial corruptions instead of erasures.

A preliminary version of this paper appeared in the Proceedings of the 13th "Innovations in Theoretical Computer Science" Conference (ITCS'22). [43].

*Supported by NSF award CCF-1909612 and Boston University's Dean's Fellowship.

†Supported by NSF award CCF-1909612.

ACM Classification: F.2

AMS Classification: 68Q25

Key words and phrases: randomized algorithms, property testing, Fourier analysis, linear functions, quadratic functions, Lipschitz and monotone functions, sorted sequences

1 Introduction

We initiate the study of sublinear-time algorithms that compute in the presence of an online adversary that blocks access to some data points in response to the algorithm's queries. A motivating scenario is when a user wishes to remove their data from a dataset due to privacy concerns, as enabled by right to be forgotten laws such as the EU General Data Protection Regulation [48]. The online aspect of our model suitably captures the case of individuals who are prompted to restrict access to their data after noticing an inquiry into their or others' data. The user could decide to remove their data after noticing, for example, that their phone number is available on websites that scrape personal contact information. We choose to model such user actions as adversarial in order to perform worst-case analysis and refrain from making distributional and other assumptions on how the data access is affected. We give two other motivating scenarios that are naturally adversarial and justify a worst-case analysis. In one, an algorithm is trying to detect some fraud (e.g., tax fraud) and the adversary wants to obstruct access to data in order to make it hard to uncover any evidence. In the other scenario, an algorithm's goal is to determine an optimal course of action (e.g., whether to invest in a stock or to buy an item), whereas the adversary leads the algorithm astray by adaptively blocking access to pertinent information.

In our model, after answering each query to the input object, the adversary can hide a small number of input values. Our goal is to understand the complexity of basic computational tasks in extremely adversarial situations, where the algorithm's access to data is blocked during the execution of the algorithm in response to its actions. Specifically, we represent the input object as a function f on an arbitrary finite domain[1], which the algorithm can access by querying a point x from the domain and receiving the answer $O(x)$ from an oracle. At the beginning of computation, $O(x) = f(x)$ for all points x in the domain of the function. We parameterize our model by a natural number t that controls the number of function values the adversary can erase *after* the oracle answers each query[2]. Mathematically, we represent the oracle and the adversary as one entity. However, it might be helpful to think of the oracle as the data holder and of the adversary as the obstructionist. A *t-online-erasure* oracle can replace values $O(x)$ on up to t points x with a special symbol $\perp$, thus erasing them. The new values will be used by the oracle to answer future queries to the corresponding points. The locations of erasures are unknown to the algorithm. The actions of the oracle can depend on the input, the queries made so far, and even on the publicly known code that the algorithm is running, but *not* on future coin tosses of the algorithm.

We focus on investigating property testing in the presence of online erasures. In the

[1]Input objects such as strings, sequences, images, matrices, and graphs can all be represented as functions. For example, an $n \times n$ real-valued matrix is equivalent to a function mapping $[n]^2$ to $\mathbb{R}$.

[2]If the adversary were allowed to erase the query of the algorithm before answering it, the algorithm would only see erased values. We give several motivating scenarios for the adversarial behavior in our model. The first example is a situation where the adversary reacts by deleting additional data after some bank records are pulled by authorities as part of an investigation. In the GDPR example mentioned previously, we argued that individuals could be prompted to restrict access to their data only after noticing an inquiry into their or others' data. Finally, in a legal setting, if the adversary is served a subpoena, they are legally bound to answer the query, but could nonetheless destroy related evidence that is not included in the subpoena.

property testing model, introduced by [64, 37] with the goal of formally studying sublinear-time algorithms, a property is represented by a set $\mathcal{P}$ (of functions satisfying the desired property). A function f is ε-far from $\mathcal{P}$ if f differs from each function $g \in \mathcal{P}$ on at least an ε fraction of domain points. The goal is to distinguish, with constant probability, functions $f \in \mathcal{P}$ from functions that are ε-far from $\mathcal{P}$. We call an algorithm a *t-online-erasure-resilient ε-tester* for property $\mathcal{P}$ if, given parameters $t \in \mathbb{N}$ and $\varepsilon \in (0, 1)$, and access to an input function f via a t-online-erasure oracle, the algorithm accepts with probability at least $2/3$ if $f \in \mathcal{P}$ and rejects with probability at least $2/3$ if f is ε-far from $\mathcal{P}$.

We study the query complexity of online-erasure-resilient testing of several fundamental properties. We show that for linearity and quadraticity of functions $f : \{0, 1\}^d \to \{0, 1\}$, the query complexity of t-online-erasure-resilient testing for constant t is asymptotically the same as in the standard model. For linearity, we also prove tight bounds in terms of t, showing that the query complexity is $\Theta(\log t)$. A function $f(x)$ is *linear* if it can be represented as a sum of monomials of the form $x[i]$, where $x = (x[1], \ldots, x[d])$ is a vector of d bits; the function is *quadratic* if it can be represented as a sum of monomials of the form $x[i]$ or $x[i]x[j]$.

To understand the difficulty of testing in the presence of online erasures, consider the case of linearity and $t = 1$. The celebrated tester for linearity in the standard property testing model was proposed by Blum, Luby, and Rubinfeld [21]. It looks for witnesses of non-linearity that consist of three points x, y, and $x \oplus y$ satisfying $f(x) + f(y) \neq f(x \oplus y)$, where addition is mod 2, and $\oplus$ denotes bitwise XOR. Bellare et al. [8] show that if f is ε-far from linear, then a triple $(x, y, x \oplus y)$ is a witness to non-linearity with probability at least ε when $x, y \in \{0, 1\}^d$ are chosen uniformly and independently at random. In our model, after x and y are queried, the oracle can erase the value of $x \oplus y$. To overcome this, our tester considers witnesses with more points, namely, of the form $\sum_{x \in T} f(x) \neq f(\bigoplus_{x \in T} x)$ for sets $T \subset \{0, 1\}^d$ of even size.

Witnesses of non-quadraticity are even more complicated. The tester of Alon et al. [2] looks for witnesses consisting of points x, y, z, and all four of their linear combinations. We describe a two-player game that models the interaction between the tester and the adversary and give a winning strategy for the tester-player. We also consider *witness structures* in which all specified tuples are witnesses of non-quadraticity (to allow for the possibility of the adversary erasing some points from the structure). We analyze the probability of getting a witness structure under uniform sampling when the input function is ε-far from quadratic. Our investigation leads to a deeper understanding of the structure of witnesses for both properties, linearity and quadraticity.

In contrast to linearity and quadraticity, we show that several other properties, specifically, sortedness and the Lipschitz property of sequences, and the Lipschitz property of functions $f : \{0, 1\}^d \to \{0, 1, 2\}$, cannot be tested in the presence of an online-erasure oracle, even with $t = 1$, no matter how many queries the algorithm makes. Interestingly, witnesses for these properties have a much simpler structure than witnesses for linearity and quadraticity. Consider the case of sortedness of integer sequences, represented by functions $f : [n] \to \mathbb{N}$. A sequence is *sorted* (or the corresponding function is *monotone*) if $f(x) \leq f(y)$ for all $x < y$ in $[n]$. A witness of non-sortedness consists of two points $x < y$, such that $f(x) > f(y)$. In the standard model, sortedness can be ε-tested with an algorithm that queries $O(\sqrt{n/\varepsilon})$

uniform and independent points [32]. (The fastest testers for this property have $O(\frac{\log \varepsilon n}{\varepsilon})$ query complexity [29, 28, 18, 23, 12], but they make correlated queries that follow a more complicated distribution.) Our impossibility result demonstrates that even the simplest testing strategy of querying independent points can be thwarted by an online adversary. To prove this result, we use sequences that are far from being sorted, but where each point is involved in only one witness, allowing the oracle to erase the second point of the witness as soon as the first one is queried. Using a version of Yao's principle that is suitable for our model, we turn these examples into a general impossibility result for testing sortedness with a 1-online-erasure oracle.

Our impossibility result for testing sortedness uses sequences with many (specifically, n) distinct integers. We show that this is not a coincidence by designing a t-online-erasure-resilient sortedness tester that works for sequences that have $O(\frac{\varepsilon^2 n}{t})$ distinct values. However, the number of distinct values does not have to be large to preclude testing the Lipschitz property in our model. A function $f : [n] \to \mathbb{N}$, representing an n-integer sequence, is *Lipschitz* if $|f(x) - f(y)| \leq |x - y|$ for all $x, y \in [n]$. Similarly, a function $f : \{0,1\}^d \to \mathbb{R}$ is *Lipschitz* if $|f(x) - f(y)| \leq \|x - y\|_1$ for all $x, y \in \{0,1\}^d$. We show that the Lipschitz property of sequences, as well as d-variate functions, cannot be tested even when the range has size 3, even with $t = 1$, no matter how many queries the algorithm makes.

Comparison to related models. Our model is closely related to (offline) erasure-resilient testing of Dixit et al. [27]. In the model of Dixit et al., also investigated in [61, 58, 13, 54, 47, 51], the adversary performs all erasures to the function before the execution of the algorithm. An (offline) erasure-resilient tester is given a parameter $\alpha \in (0, 1)$, an upper bound on the fraction of the values that are erased. The adversary we consider is more powerful in the sense that it can perform erasures online, during the execution of the tester. However, in some parameter regimes, our oracle cannot perform as many erasures. Importantly, all three properties that we show are impossible to test in our model, are testable in the model of Dixit et al. with essentially the same query complexity as in the standard model [27]. It is open if there are properties that have lower query complexity in the online model than in the offline model. The models are not directly comparable because the erasures are budgeted differently.

Another widely studied model in property testing is that of tolerant testing [56]. As explained by Dixit et al., every tolerant tester is also (offline) erasure-resilient with corresponding parameters. As pointed out in [56], the BLR tester is a tolerant tester of linearity for α significantly smaller than ε. Tolerant testing of linearity with distributional assumptions was studied in [46] and tolerant testing of low-degree polynomials over large alphabets was studied in [38]. Tolerant testing of sortedness is closely related to approximating the distance to monotonicity and estimating the longest increasing subsequence. These tasks can be performed with polylogorithmic in n number of queries [56, 1, 65]. As we showed, sortedness is impossible to test in the presence of online erasures.

Adversarial models are also studied in related contexts, such as sampling from a stream [15], more general stream computations [14], and dynamic algorithms [7]. Notably, in these models, the input is formed adversarially online while the algorithm is running; there is no notion of corrupted input. In contrast, in our model, algorithms solve problems on the original, ground-truth input while the access to the input is degrading.

1.1 Our results

We design t-online-erasure-resilient testers for linearity and quadraticity, two properties widely studied because of their connection to probabilistically checkable proofs, hardness of approximating NP-hard problems, and coding theory. Our testers have *1-sided error*, that is, they always accept functions with the property. They are also *nonadaptive*, that is, their queries do not depend on answers to previous queries. This is despite the adversary being allowed to respond online (i.e., adaptively) to the algorithm's queries.

Linearity. Starting from the pioneering work of [21], linearity testing has been investigated, e.g., in [10, 11, 30, 8, 9, 71, 70, 67, 40, 16, 66, 68, 69, 44] (see [59] for a survey). Linearity can be ε-tested in the standard property testing model with $O(1/\varepsilon)$ queries by the BLR tester. We say that a pair (x, y) *violates* linearity if $f(x) + f(y) \neq f(x \oplus y)$. The BLR tester repeatedly selects a uniformly random pair of domain points and rejects if it violates linearity. A tight lower bound of $\Theta(\varepsilon)$ on the probability that a uniformly random pair violates linearity was proven by Bellare et al. [8] and Kaufman et al. [44].

We show that linearity can be ε-tested with $\widetilde{O}(\log t/\varepsilon)$ queries with a t-online-erasure oracle.

Theorem 1.1. *There exist a constant $c_0 \in (0,1)$ and a 1-sided error, nonadaptive, t-online-erasure-resilient ε-tester for linearity of functions $f\colon \{0,1\}^d \to \{0,1\}$ that works for all $t \leq c_0 \cdot \varepsilon^{5/4} \cdot 2^{d/4}$ and makes $O\big(\min\big(\frac{1}{\varepsilon}\log\frac{t}{\varepsilon}, \frac{t}{\varepsilon}\big)\big)$ queries.*

Our linearity tester has query complexity $O(1/\varepsilon)$ for constant t, which is optimal even in the standard property testing model, with no erasures. The tester looks for more general witnesses of non-linearity than the BLR tester, namely, it looks for tuples T of elements from $\{0,1\}^d$ such that $\sum_{x\in T} f(x) \neq f(\bigoplus_{x\in T} x)$ and $|T|$ is even. We call such tuples *violating*. The analysis of our linearity tester crucially depends on the following structural theorem.

Theorem 1.2. *Let T be a tuple of a fixed even size, where each element of T is sampled uniformly and independently at random from $\{0,1\}^d$. If a function $f\colon \{0,1\}^d \to \{0,1\}$ is ε-far from linear, then*

$$\Pr_T\Big[\sum_{x\in T} f(x) \neq f\big(\bigoplus_{x\in T} x\big)\Big] \geq \varepsilon.$$

Our theorem generalizes the result of [8], which dealt with the case $|T| = 2$. We remark that the assertion in Theorem 1.2 does not hold for odd $|T|$. Consider the function $f(x) = x[1] + 1 \pmod 2$, where $x[1]$ is the first bit of x. Function f is $\frac{1}{2}$-far from linear, but has no violating tuples of odd size.

The core procedure of our linearity tester queries $\Theta(\log(t/\varepsilon))$ uniformly random points from $\{0,1\}^d$ to build a reserve and then queries sums of the form $\bigoplus_{x\in T} x$, where T is a uniformly random tuple of reserve elements such that $|T|$ is even. The *quality* of the reserve is the probability that T is violating. The likelihood that the procedure catches a violating tuple depends on the quality of the reserve (which is a priori unknown to the tester) and the number of sums queried. Instead of querying the same number of sums in each iteration of the core procedure, we obtain a better query complexity by guessing different reserve qualities for each iteration and querying the number of sums that is inversely proportional to the reserve quality. We decide on the number of sums to query based on the *work investment* strategy by Berman, Raskhodnikova, and Yaroslavtsev [17], which builds on an idea proposed by Levin and popularized by Goldreich [35].

Next, we show that our tester has optimal query complexity in terms of the erasure budget t.

Theorem 1.3. *For all $\varepsilon \in (0, \frac{1}{4}]$, every t-online-erasure-resilient ε-tester for linearity of functions $f\colon \{0,1\}^d \to \{0,1\}$ must make more than $\log_2 t$ queries.*

The main idea in the proof of Theorem 1.3 is that when a tester makes $\lfloor \log_2 t \rfloor$ queries, the adversary has the budget to erase all linear combinations of the previous queries after every step. As a result, the tester cannot distinguish a random linear function from a random function.

Quadraticity. Quadraticity and, more generally, low-degree testing have been studied, e.g., in [6, 5, 34, 30, 33, 64, 62, 2, 3, 50, 49, 45, 66, 68, 42, 19, 39, 63, 25]. Low-degree testing is closely related to local testing of Reed-Muller codes. The Reed-Muller code $C(k, d)$ consists of codewords, each of which corresponds to all evaluations of a polynomial $f\colon \{0,1\}^d \to \{0,1\}$ of degree at most k. A local tester for a code queries a few locations of a codeword; it accepts if the codeword is in the code; otherwise, it rejects with probability proportional to the distance of the codeword from the code.

In the standard property testing model, quadraticity can be ε-tested with $O(1/\varepsilon)$ queries by the tester of Alon et al. [2] that repeatedly selects $x, y, z \sim \{0,1\}^d$ and queries f on all of their linear combinations—the points themselves, the double sums $x \oplus y$, $x \oplus z$, $y \oplus z$, and the triple sum $x \oplus y \oplus z$. The tester rejects if the values of the function on all seven queried points sum to 1, since this cannot happen for a quadratic function. A tight lower bound on the probability that the resulting 7-tuple is a witness of non-quadraticity was proved by Alon et al. [2] and Bhattacharyya et al. [19].

We prove that quadraticity can be ε-tested with $O(1/\varepsilon)$ queries with a t-online-erasure-oracle for constant t. Our tester can be easily modified to give a local tester for the Reed-Muller code $C(2, d)$ that works with a t-online-erasure oracle.

Theorem 1.4. *There exists a 1-sided error, nonadaptive, t-online-erasure-resilient ε-tester for quadraticity of functions $f\colon \{0,1\}^d \to \{0,1\}$ that makes $O(\frac{1}{\varepsilon})$ queries for constant t.*

The dependence on t in the query complexity of our quadraticity tester is at least doubly exponential, and it is an open question whether it can be improved. The main ideas behind our quadraticity tester are explained in Section 1.2.

Sortedness. Sortedness testing (see [57] for a survey) was introduced by Ergun et al. [29]. Its query complexity has been pinned down to $\Theta(\frac{\log(\varepsilon n)}{\varepsilon})$ by [29, 31, 24, 12].

We show that online-erasure-resilient testing of integer sequences is, in general, impossible.

Theorem 1.5. *For all $\varepsilon \in (0, \frac{1}{12}]$, there is no 1-online-erasure-resilient ε-tester for sortedness of integer sequences.*

In the case without erasures, sortedness can be tested with $O(\sqrt{n/\varepsilon})$ uniform and independent queries [32]. Theorem 1.5 implies that a uniform tester for a property does not translate into the existence of an online-erasure-resilient tester, counter to the intuition that testers that make only uniform and independent queries should be less prone to adversarial attacks. Our lower bound construction demonstrates that the structure of violations to a property plays an important role in determining whether the property is testable.

The hard sequences from the proof of Theorem 1.5 have n distinct values. Pallavoor et al. [53, 55] considered the setting when the tester is given an additional parameter r, the number of distinct elements in the sequence, and obtained an $O(\frac{\log r}{\varepsilon})$-query tester. Two lower bounds apply to this setting: $\Omega(\log r)$ for nonadaptive testers [20] and $\Omega(\frac{\log r}{\log\log r})$ for all testers for the case when $r = n^{1/3}$ [12]. Pallavoor et al. also showed that sortedness can be tested with $O(\sqrt{r}/\varepsilon)$ uniform and independent queries. We extend the result of Pallavoor et al. to the setting with online erasures for the case when r is small.

Theorem 1.6. *Let $c_0 > 0$ be a constant. There exists a 1-sided error, nonadaptive, t-online-erasure-resilient ε-tester for sortedness of n-element sequences with at most r distinct values. The tester makes $O(\frac{\sqrt{r}}{\varepsilon})$ uniform and independent queries and works when $r < \frac{\varepsilon^2 n}{c_0 t}$.*

Thus, sortedness is not testable with online erasures when r is large and is testable in the setting when r is small. For example, for Boolean sequences, it is testable with $O(1/\varepsilon)$ queries.

The Lipschitz property. Lipschitz testing, introduced by [41], was subsequently studied in [23, 26, 17, 4, 22]. Lipschitz testing of functions $f : [n] \to \{0, 1, 2\}$ can be performed with $O(\frac{1}{\varepsilon})$ queries [41]. For functions $f : \{0, 1\}^d \to \mathbb{R}$, it can be done with $O(\frac{d}{\varepsilon})$ queries [41, 23].

We show that the Lipschitz property is impossible to test in the online-erasures model even when the range of the function has only 3 distinct values. This applies to both domains, $[n]$ and $\{0, 1\}^d$.

Theorem 1.7. *For all $\varepsilon \in (0, \frac{1}{8}]$, there is no 1-online-erasure-resilient ε-tester for the Lipschitz property of functions $f : [n] \to \{0, 1, 2\}$. The same statement holds when the domain is $\{0, 1\}^d$ instead of $[n]$.*

Yao's minimax principle. All our lower bounds use Yao's minimax principle. A formulation of Yao's principle suitable for our online-erasures model is described in Section 9.

1.2 The ideas behind our quadraticity tester

One challenge in generalizing the tester of Alon et al. [2] to work with an online-erasure oracle is that its queries are correlated. First, we want to ensure that the tester can obtain function values on tuples of the form $(x, y, z, x \oplus y, x \oplus z, y \oplus z, x \oplus y \oplus z)$. Then we want to ensure that, if the original function is far from the property, the tester is likely to catch such a tuple that is also a witness to not satisfying the property. Next, we formulate a two-player game[3] that abstracts the first task. In the game, the tester-player sees what erasures are made by the oracle-player. This assumption is made to abstract out the most basic challenge and is not used in the algorithms' analyses.

Quadraticity testing as a two-player game. Player 1 represents the tester and Player 2 represents the adversary. The players take turns drawing points, connecting points with edges, and coloring triangles specified by drawn points, each in their own color. Player 1 wins the game if it draws in blue all the vertices and edges of a triangle and colors the triangle blue. The vertices represent the points $x, y, z \in \{0,1\}^d$, the edges are the sums $x \oplus y$, $x \oplus z$, $y \oplus z$, and the triangle is the sum $x \oplus y \oplus z$. A move of Player 1 consists of drawing a point or an edge between two existing non-adjacent points or coloring an uncolored triangle between three existing points (in blue). A move of Player 2 consists of at most t steps; in each step, it can draw a red edge between existing points or color a triangle between three existing points (in red).

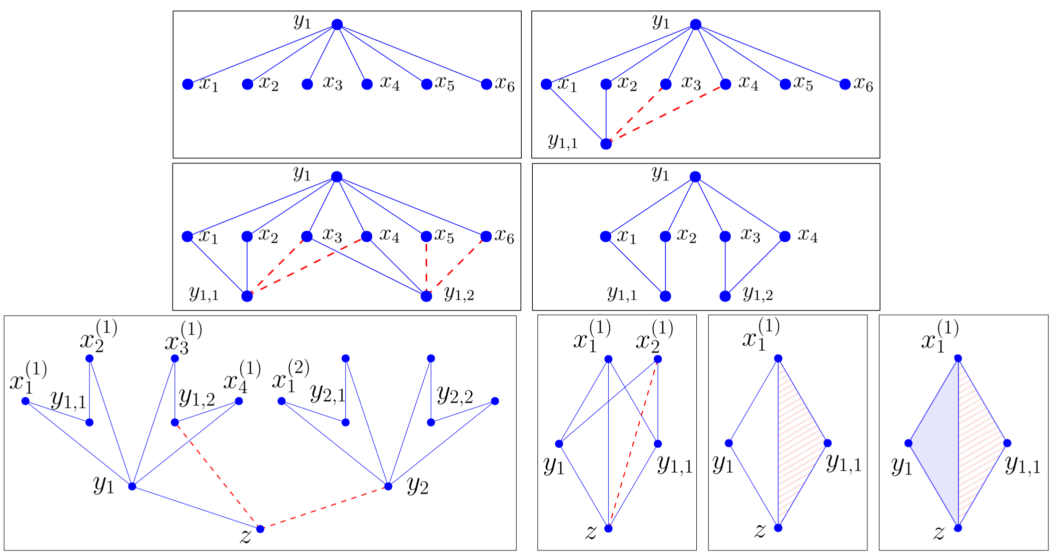

Figure 1: Stages in the quadraticity game for $t = 1$, played according to the winning strategy for Player 1: connecting the y-decoys from the first tree to x-decoys (frames 1-4); drawing z and connecting it to y-decoys and an x-decoy (frames 5-6), and creation of a blue triangle (frames 7–8). Frame 5 contains edges from z to two structures, each replicating frame 4. We depict only points and edges relevant for subsequent frames.

Our online-erasure-resilient quadraticity tester is based on a winning strategy for Player 1 with $t^{O(t)}$ moves. At a high level, Player 1 first draws many decoys for x. The y-decoys are organized in $t+1$ full $(t+1)$-ary trees of depth t. The root for tree i is y_i, its children are $y_{i,j}$,

[3] This game has been tested on real children, and they spent hours playing it.

where $j \in [t+1]$, etc. We jot the rest of the winning strategy for $t = 1$ and depict it in Fig. 1. In this case, Player 1 does the following for each of two trees: it draws points $x_1^{(i)}, \ldots, x_{12}^{(i)}, y_i$; connects y_i to half of x-decoys (w.l.o.g., $x_1^{(i)}, \ldots, x_6^{(i)}$); draws point $y_{i,1}$, connects it to two of the x-decoys adjacent to y_1 (w.l.o.g., $x_1^{(i)}$ and $x_2^{(i)}$); draws point $y_{i,2}$, connects it to two of $x_3^{(i)}, \ldots, x_6^{(i)}$ (w.l.o.g., $x_3^{(i)}$ and $x_4^{(i)}$); draws z and connects it to one of the roots (w.l.o.g., y_1), connects z to one of $y_{1,1}$ and $y_{1,2}$ (w.l.o.g., $y_{1,1}$), connects z to one of $x_1^{(1)}$ and $x_2^{(1)}$ (w.l.o.g., $x_1^{(1)}$), and finally colors one of the triangles $x_1^{(1)}y_1z$ and $x_1^{(1)}y_{1,1}z$, thus winning the game. The decoys are arranged to guarantee that Player 1 always has at least one available move in each step of the strategy.

For general t, the winning strategy is described in full detail in Algorithm 3. Recall that the y-decoys are organized in $t+1$ full $(t+1)$-ary trees of depth t. For every root-to-leaf path in every tree, Player 1 draws edges from all the nodes in that path to a separate set of $t+1$ decoys for x. After z is drawn, the tester "walks" along a root-to-leaf path in one of the trees, drawing edges between z and the y-decoys on the path. The goal of this walk is to avoid the parts of the tree spoiled by Player 2. Finally, Player 1 connects z to an x-decoy that is adjacent to all vertices in the path, and then colors a triangle involving this x-decoy, a y-decoy from the chosen path, and z. The structure of decoys guarantees that Player 1 always has $t+1$ options for its next move, only t of which can be spoiled by Player 2.

From the game to a tester. There are two important aspects of designing a tester that are abstracted away in the game: First, the tester does not actually know which values are erased until it queries them. Second, the tester needs to catch a witness demonstrating a violation of the property, not merely a tuple of the right form with no erasures. Here, we briefly describe how we overcome these challenges.

Our quadraticity tester is based directly on the game. It converts the moves of the winning strategy of Player 1 into a corresponding procedure, making a uniformly random guess at each step about the choices that remain nonerased. There are three core technical lemmas used in the analysis of the algorithm. Lemma 5.4 lower bounds the probability that the tester makes correct guesses at each step about which edges (double sums) and triangles (triple sums) remain nonerased, thus addressing part of the first challenge. This probability depends only on the erasure budget t. To address the second challenge, Lemma 5.3 gives a lower bound on the probability that uniformly random sampled points (the x- and y- decoys together with z) form a large violation structure, where all triangles that Player 1 might eventually complete violate quadraticity. Building on a result of Alon et al., we show that even though the number of triangles involved in the violation structure is large, namely $t^{O(t)}$, the probability of sampling such a structure is $\alpha = \Omega(\min(\varepsilon, c_t))$, where $c_t \in (0,1)$ depends only on t. Finally, Lemma 5.2 shows that despite the online adversarial erasures, the tester has a probability of $\alpha/2$ of sampling the points of such a large violation structure and obtaining their values from the oracle. These three lemmas combined show that quadraticity can be tested with $O(1/\varepsilon)$ queries for constant t.

1.3 Sublinear-time computation in the presence of online corruptions

Our model of online erasures extends naturally to a model of online corruptions, where the adversary is allowed to modify values of the input function. Specifically, a *t-online-corruption* oracle $\mathcal{O}$, after answering each query, can replace values $\mathcal{O}(x)$ on up to t domain points x with arbitrary values in the range of f. Our algorithmic results in the presence of online erasures have implications for computation in the presence of online corruptions.

We consider two types of computational tasks in the presence of online corruptions. The first one is *t-online-corruption-resilient* testing, defined analogously to t-online-erasure-resilient testing. Specifically, we call an algorithm a *t-online-corruption-resilient ε-tester* for property $\mathcal{P}$ if, given parameters $t \in \mathbb{N}$ and $\varepsilon \in (0, 1)$, and access to an input function f via a t-online-erasure oracle, the algorithm accepts with probability at least $2/3$ if $f \in \mathcal{P}$ and rejects with probability at least $2/3$ if f is ε-far from $\mathcal{P}$. In Lemma 1.8 we study a general computational task for which the algorithm has oracle access to an input function and has to output a correct answer with high probability. We consider algorithms that access their input via an online-erasure oracle and show that if with high probability they output a correct answer *and* encounter no erased queries during their execution, they also output a correct answer with high probability in the presence of online corruptions. Note that the correctness of the answer is with respect to the original function f, as opposed to the corrupted values represented by $\mathcal{O}$. Lemma 1.8 uses the notion of an *adversarial strategy*. We model an adversarial strategy as a distribution on decision trees, where each tree dictates the erasures to be made for a given input and queries of the algorithm.

Lemma 1.8. *Let 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 $2/3$, the algorithm T outputs a correct answer and queries no erased values during its execution. Then, for the same computational task, when T is given access to a function via a t-online-corruption oracle, it outputs a correct answer with probability at least $2/3$.*

This lemma can be applied to our online-erasure-resilient testers for linearity and for sortedness with few distinct values. These testers have 1-sided error, i.e., they err only when the function is far from the property. We can amplify the success probability of the testers to $5/6$ for functions that are far from the property, without changing the asymptotic query complexity of the testers. In addition, the resulting testers have a small probability (specifically, at most $1/6$) of encountering an erasure during their computation. As a result, Lemma 1.8 implies online-corruption-resilient testers for these properties. Their performance is summarized in Corollaries 8.1 and 8.2.

For the second type of computational tasks, we return to the motivation of the algorithm looking for evidence of fraud. In Lemma 1.9, we show that every nonadaptive, 1-sided error, online-erasure-resilient tester for any property $\mathcal{P}$ can be modified so that, given access to an input function f that is far from $\mathcal{P}$ via an online-corruption oracle, it outputs a witness of f not being in $\mathcal{P}$. This result applies to our testers for sortedness with few distinct values, linearity, and quadraticity. Note that the values of the witness could be corrupted values of f. For this task, it is helpful to think of the input f as changing dynamically rather than there being a ground truth input f as is the case for Lemma 1.8.

Lemma 1.9. *Let $\mathcal{P}$ be a property of functions that has a 1-sided error, nonadaptive, t-online-erasure-resilient tester. Then there exists a nonadaptive algorithm with the same query complexity that, given a parameter ε and access via a t-online-corruption-oracle $\mathcal{O}$ to a function f that is ε-far from $\mathcal{P}$, outputs with probability at least $2/3$ a witness of $f \notin \mathcal{P}$. The witness consists of queried domain points $x_1, \ldots, x_k$ and values $\mathcal{O}(x_1), \ldots, \mathcal{O}(x_k)$ obtained by the algorithm, such that no $g \in \mathcal{P}$ has $g(x_i) = \mathcal{O}(x_i)$ for all $i \in [k]$.*

1.4 Conclusions and open questions

We initiate a study of sublinear-time algorithms in the presence of online adversarial erasures. We design efficient online-erasure-resilient testers for several important properties (linearity, quadraticity, and—for the case of small number of distinct values—sortedness). For linearity, we prove tight upper and lower bounds in terms of t. We also show that several basic properties, specifically, sortedness of integer sequences and the Lipschitz properties, cannot be tested in our model. We now list several open problems.

- Sortedness is an example of a property that is impossible to test with online erasures, but is easy to test with offline erasures, as well as tolerantly. Is there a property that has smaller query complexity in the online-erasure-resilient model than in the (offline) erasure-resilient model of [27]?

- We design a t-online-erasure-resilient quadraticity tester that makes $O(1/\varepsilon)$ queries for constant t. What is the query complexity of t-online-erasure-resilient quadraticity testing in terms of t and ε? Specifically, the dependence on t in the query complexity of our quadraticity tester is at least doubly exponential, and it is open whether it can be improved.

- The query complexity of ε-testing if a function is a polynomial of degree at most k is $\Theta(\frac{1}{\varepsilon} + 2^k)$ [2, 19]. Is there a low-degree test for $k \geq 3$ that works in the presence of online erasures?

- Our tester for linearity works in the presence of online corruptions, but our tester for quadraticity does not. The reason for this is that our linearity tester is unlikely to see erasures, but that is not the case for our quadraticity tester. Is there an online-corruption-resilient quadraticity tester?

2 An online-erasure-resilient linearity tester

To prove Theorem 1.1, we present and analyze two testers. Our main online-erasure-resilient linearity tester (Algorithm 1) is presented in this section. Its query complexity has optimal dependence on t and nearly optimal dependence on ε. Its performance is summarized in Theorem 2.1. To complete the proof of Theorem 1.1, we give a $O(t/\varepsilon)$-query linearity tester in Section 3. It has optimal query complexity for constant t.

Theorem 2.1. *There exists $c_0 \in (0,1)$ and a 1-sided error, nonadaptive, t-online-erasure-resilient ε-tester for linearity of functions $f\colon \{0,1\}^d \to \{0,1\}$ that works for $t \le c_0 \cdot \varepsilon^{5/4} \cdot 2^{d/4}$ and makes $O(\frac{1}{\varepsilon}\log\frac{t}{\varepsilon})$ queries.*

The t-online-erasure-resilient tester guaranteed by Theorem 2.1 is presented in Algorithm 1.

Algorithm 1 An Online-Erasure-Resilient Linearity Tester

Input: $\varepsilon \in (0, \frac{1}{2})$, erasure budget $t \in \mathbb{N}$, access to function $f : \{0,1\}^d \to \{0,1\}$ via a t-online-erasure oracle.

```
Let q = 2 log(50t/ε).
for all j ∈ [log(8/ε)]:
    repeat (8 ln 5)/(2^j ε) times:
        for all i ∈ [q]:
            Sample x_i ~ {0,1}^d and query f at x_i.
        repeat 4 · 2^j times:
            Sample a uniform nonempty subset I of [q] of even size.
            Query f at ⊕_{i∈I} x_i.
            Reject if Σ_{i∈I} f(x_i) ≠ f(⊕_{i∈I} x_i) and all points x_i for i ∈ I and ⊕_{i∈I} x_i are nonerased.
Accept.
```

2.1 Proof of Theorem 1.2

In this section, we prove Theorem 1.2, the main structural result used in Theorem 2.1. Recall that a k-tuple $(x_1, \dots, x_k) \in (\{0,1\}^d)^k$ violates linearity if $f(x_1) + \cdots + f(x_k) \neq f(x_1 \oplus \cdots \oplus x_k)$. (Addition is mod 2 when adding values of Boolean functions.) Theorem 1.2 states that if f is ε-far from linear, then for all even k, with probability at least ε, independently and uniformly sampled points $x_1, \dots, x_k \sim \{0,1\}^d$ form a violating k-tuple. Our proof of Theorem 1.2 builds on the proof of [8, Theorem 1.2], which is a special case of Theorem 1.2 for $k = 2$. The proof is via Fourier analysis. Next, we state some standard facts and definitions related to Fourier analysis. See, e.g., [52] for proofs of these facts.

Consider the space of all real-valued functions on $\{0,1\}^d$ equipped with the inner-product

$$\langle g, h \rangle = \mathop{\mathbb{E}}_{x \sim \{0,1\}^d}[g(x)h(x)],$$

where $g, h\colon \{0,1\}^d \to \mathbb{R}$. The *character* functions $\chi_S\colon \{0,1\}^d \to \{-1,1\}$, defined as $\chi_S = (-1)^{\sum_{i\in S} x[i]}$ for $S \subseteq [d]$, form an orthonormal basis for the space of functions under consideration. Hence, every function $g : \{0,1\}^d \to \mathbb{R}$ can be uniquely expressed as a linear combination of the functions χ_S, where $S \subseteq [d]$. The Fourier coefficients of g are the coefficients on the functions χ_S in this linear representation of g.

Definition 2.2 (Fourier coefficient)**.** For $g\colon \{0,1\}^d \to \mathbb{R}$ and $S \subseteq [d]$, the Fourier coefficient of g on S is $\widehat{g}(S) = \langle g, \chi_S \rangle = \mathop{\mathbb{E}}_{x\sim\{0,1\}^d}[g(x)\chi_S(x)]$.

We will need the following facts about Fourier coefficients.

Theorem 2.3 (Parseval's Theorem)**.** *For all functions $g\colon \{0,1\}^d \to \mathbb{R}$, it holds that $\langle g, g\rangle = \sum_{S\subseteq[d]} \widehat{g}(S)^2$. In particular, if $g\colon \{0,1\}^d \to \{-1,1\}$ then $\sum_{S\subseteq[d]} \widehat{g}(S)^2 = 1$.*

Theorem 2.4 (Plancherel's Theorem)**.** *For all functions $g, h\colon \{0,1\}^d \to \mathbb{R}$, it holds that $\langle g, h\rangle = \sum_{S\subseteq[d]} \widehat{g}(S)\widehat{h}(S)$.*

A function $g\colon \{0,1\}^d \to \{-1,1\}$ is linear if $g(x)g(y) = g(x\oplus y)$ for all $x, y \in \{0,1\}^d$.

Lemma 2.5. *The distance of $g\colon \{0,1\}^d \to \{-1,1\}$ to linearity is $\frac{1}{2} - \frac{1}{2}\max_{S\subseteq[d]} \widehat{g}(S)$.*

Finally, we also use the convolution operation, defined below, and one of its key properties.

Definition 2.6 (Convolution)**.** Let $g, h\colon \{0,1\}^d \to \mathbb{R}$. Their convolution is the function $g * h : \{0,1\}^d \to \mathbb{R}$ defined by $(g*h)(x) = \mathop{\mathbb{E}}_{y\sim\{0,1\}^d}[g(y)h(x\oplus y)]$.

Theorem 2.7. *Let $g, h\colon \{0,1\}^d \to \mathbb{R}$. Then, for all $S \subseteq [d]$, it holds $\widehat{g*h}(S) = \widehat{g}(S)\widehat{h}(S)$.*

Proof of Theorem 1.2. Define $g\colon \{0,1\}^d \to \{-1,1\}$ so that $g(x) = (-1)^{f(x)}$. That is, g is obtained from the function f by encoding its output with ± 1. Note that the distance to linearity of g is the same as the distance to linearity of f. We have that the expression $\frac{1}{2} - \frac{1}{2}g(x_1)\ldots g(x_k)g(x_1\oplus\cdots\oplus x_k)$ is an indicator for the event that points $x_1,\ldots,x_k \sim \{0,1\}^d$ violate linearity for g. Define g^{*k} to be the convolution of g with itself k times, i.e., $g^{*k} = g*\cdots*g$, where the $*$ operator appears $k-1$ times. We obtain

$$
\begin{aligned}
&\Pr_{x_1,\ldots,x_k\sim\{0,1\}^d}[(x_1,\ldots,x_k) \text{ violates linearity}]\\
&= \mathop{\mathbb{E}}_{x_1,\ldots,x_k\sim\{0,1\}^d}\left[\frac{1}{2} - \frac{1}{2}g(x_1)\ldots g(x_k)g(x_1\oplus\cdots\oplus x_k)\right]\\
&= \frac{1}{2} - \frac{1}{2}\mathop{\mathbb{E}}_{x_1,\ldots,x_{k-1}\sim\{0,1\}^d}[g(x_1)\ldots g(x_{k-1})\cdot \mathop{\mathbb{E}}_{x_k\sim\{0,1\}^d}[g(x_k)g(x_1\oplus\cdots\oplus x_k)]]\\
&= \frac{1}{2} - \frac{1}{2}\mathop{\mathbb{E}}_{x_1,\ldots,x_{k-1}\sim\{0,1\}^d}[g(x_1)\ldots g(x_{k-1})(g*g)(x_1\oplus\cdots\oplus x_{k-1})] && (2.1)\\
&= \frac{1}{2} - \frac{1}{2}\mathop{\mathbb{E}}_{x_1\sim\{0,1\}^d}[g(x_1)\cdot(g^{*k})(x_1)] && (2.2)\\
&= \frac{1}{2} - \frac{1}{2}\sum_{S\subseteq[d]} \widehat{g}(S)\widehat{g^{*k}}(S) && (2.3)\\
&= \frac{1}{2} - \frac{1}{2}\sum_{S\subseteq[d]} \widehat{g}(S)^{k+1}, && (2.4)
\end{aligned}
$$

where (2.1) holds by the definition of convolution, (2.2) follows by repeated application of the steps used to obtain (2.1), (2.3) follows from Plancherel's Theorem (Theorem 2.4), and (2.4) follows from Theorem 2.7.

Note that $|\widehat{g}(S)| \le 1$ for all $S \subseteq [d]$, because $\widehat{g}(S)$ is the inner product of two functions with range in $\{-1, 1\}$. In addition, $\widehat{g}(S) \ge 0$ for S such that χ_S is the closest linear function to g. Then, for even k,

$$\sum_{S\subseteq[d]} \widehat{g}(S)^{k+1} \le \max_{S\subseteq[d]}(\widehat{g}(S)^{k-1}) \sum_{S\subseteq[d]} \widehat{g}(S)^2 = \max_{S\subseteq[d]}(\widehat{g}(S)^{k-1}) \le \max_{S\subseteq[d]} \widehat{g}(S),$$

where the equality follows from Parseval's Theorem (Theorem 2.3). By Lemma 2.5, the distance of g to linearity is $\frac{1}{2} - \frac{1}{2}\max_{S\subseteq[d]} \widehat{g}(S)$, which is at least ε, since g is ε-far from linear. This concludes the proof. □

2.2 Proof of Theorem 2.1

In this section, we prove Theorem 2.1 using Theorem 1.2. In Lemma 2.8, we analyze the probability of good events that capture, roughly, that the queries made in the beginning of each iteration haven't already been "spoiled" by the previous erasures. Then we use the work investment strategy of [17], stated in Lemma 2.9, together with Theorem 1.2 and Lemma 2.8 to prove Theorem 2.1.

Each iteration of the outer **repeat** loop in Steps 3-9 of Algorithm 1 is called a *round*. We say a query x is *successfully obtained* if it is nonerased when queried, i.e., the tester obtains $f(x)$ as opposed to $\perp$.

Lemma 2.8 (Good events). *Fix one round of Algorithm 1. Consider the points $x_1, \ldots, x_q$ queried in Step 5 of this round, where $q = 2\log(50t/\varepsilon)$, and the set S of all sums $\bigoplus_{i\in I} x_i$, where I is a nonempty subset of $[q]$ of even size. Let G_1 be the (good) event that all points in S are distinct. Let G_2 be the (good) event that all points $x_1, \ldots, x_q$ are successfully obtained and all points in S are nonerased at the beginning of the round. Finally, let $G = G_1 \cap G_2$. Then $\Pr[\overline{G}] \le \frac{\varepsilon}{2}$ for all adversarial strategies.*

Proof. First, we analyze event G_1. Consider points $x_{i_1}, \ldots, x_{i_k}, x_{i'_1}, \ldots, x_{i'_\ell} \sim \{0,1\}^d$, where $\{i_1, \ldots, i_k\} \ne \{i'_1, \ldots, i'_\ell\}$, $k, \ell \in [q]$ and k, ℓ are even. Since the points are distributed uniformly and independently, so are the sums $x_{i_1} \oplus \cdots \oplus x_{i_k}$ and $x_{i'_1} \oplus \cdots \oplus x_{i'_\ell}$. The probability that two uniformly and independently sampled points $x, y \sim \{0,1\}^d$ are identical is $\frac{1}{2^d}$. The number of sets $I \subseteq [q]$ of even size is 2^{q-1} because every subset of $[q-1]$ can be uniquely completed to such a set I. By a union bound over all pairs of sums, $\Pr[\overline{G_1}] \le \frac{2^{2q}}{4\cdot 2^d}$.

To analyze G_2, fix any adversarial strategy. The number of queries made by Algorithm 1 is at most

$$\sum_{j=1}^{\log(8/\varepsilon)} \frac{8\ln 5}{2^j\,\varepsilon}\left(q + 4\cdot 2^j\right) \le \frac{8\ln 5}{\varepsilon} q + \frac{32\ln 5}{\varepsilon}\log\frac{8}{\varepsilon} \le \frac{8\ln 5}{\varepsilon} q + \frac{16\ln 5}{\varepsilon} q \le \frac{40q}{\varepsilon}. \tag{2.5}$$

Hence, the oracle erases at most $\frac{40qt}{\varepsilon}$ points. Since each point x_i is sampled uniformly from $\{0,1\}^d$,

$$\Pr_{x_i \sim \{0,1\}^d}[x_i \text{ is erased when queried}] \le \frac{40qt}{\varepsilon \cdot 2^d}.$$

Additionally, before the queries x_i are revealed to the oracle, each sum $\bigoplus_{i \in I} x_i$ is distributed uniformly at random. Therefore, for every $\{i_1, \ldots, i_k\} \subseteq [q]$,

$$\Pr_{x_{i_1}, \ldots, x_{i_k} \sim \{0,1\}^d}[x_{i_1} \oplus \cdots \oplus x_{i_k} \text{ is erased at the beginning of the round}] \le \frac{40qt}{\varepsilon \cdot 2^d}.$$

By a union bound over the q points sampled in Step 5 and at most 2^{q-1} sums, we get

$$\Pr[\overline{G_2}] \le \frac{40qt}{\varepsilon 2^d}(q + 2^{q-1}) \le \frac{40qt \cdot 2^q}{\varepsilon \cdot 2^d}.$$

Since $q = 2\log\frac{50t}{\varepsilon}$, we get $\frac{40qt}{\varepsilon} \le \frac{3}{4} \cdot 2^q$ and, consequently,

$$\Pr[\overline{G}] \le \frac{2^{2q}}{4 \cdot 2^d} + \frac{40qt \cdot 2^q}{\varepsilon \cdot 2^d} \le \frac{2^{2q}}{2^d} \le \frac{50^4 \cdot t^4}{\varepsilon^4 2^d} \le \frac{50^4 \cdot c_0^4 \cdot \varepsilon^5}{\varepsilon^4} \le \frac{\varepsilon}{2},$$

since $t \le c_0 \cdot \varepsilon^{5/4} \cdot 2^{d/4}$, as stated in Theorem 2.1, and assuming c_0 is sufficiently small. □

Next, we state the work investment lemma.

Lemma 2.9 (Lemma 2.5 of [17]). *Let X be a random variable taking values in $[0,1]$. Suppose $\mathbb{E}[X] \ge \alpha$. Let $s = \lceil \log(\frac{4}{\alpha}) \rceil$ and $\delta \in (0,1)$ be the desired probability of error. For all $j \in [s]$, let $p_j = \Pr[X \ge 2^{-j}]$ and $k_j = \frac{4\ln(1/\delta)}{2^j \alpha}$. Then $\prod_{j=1}^{s}(1-p_j)^{k_j} \le \delta$.*

Proof of Theorem 2.1. By (2.5), the query complexity of Algorithm 1 is $O(\frac{q}{\varepsilon}) = O(\frac{\log(t/\varepsilon)}{\varepsilon})$. Algorithm 1 is nonadaptive and always accepts if f is linear. Suppose now that f is ε-far from linear and fix any adversarial strategy. We show that Algorithm 1 rejects with probability at least 2/3.

Consider the *last* round of Algorithm 1. For points $x_1, \ldots, x_q \sim \{0,1\}^d$ sampled in Step 5 of this last round, let Y denote the fraction of nonempty sets $\{i_1, \ldots, i_k\} \subseteq [q]$ such that k is even and $(x_{i_1}, \ldots, x_{i_k})$ violates linearity. Recall the event G defined in Lemma 2.8. Let $\mathbf{1}_G$ be the indicator random variable for the event G for the last round.

Claim 2.10. *Let $X = Y \cdot \mathbf{1}_G$, where Y is as defined above. Then $\mathbb{E}[X] \ge \frac{\varepsilon}{2}$.*

Proof. For all nonempty $\{i_1, \ldots, i_k\} \subseteq [q]$, such that k is even, let $Y_{i_1,\ldots,i_k}$ be the indicator for the event that $(x_{i_1}, \ldots, x_{i_k})$ violates linearity. By Theorem 1.2 and the fact that k is even,

$$\mathbb{E}[Y_{i_1,\ldots,i_k}] = \Pr_{x_{i_1}, \ldots, x_{i_k} \sim \{0,1\}^d}[(x_{i_1}, \ldots, x_{i_k}) \text{ violates linearity}] \ge \varepsilon.$$

We obtain a lower bound on $\mathbb{E}[Y]$ by linearity of expectation.

$$\mathbb{E}[Y] = \frac{1}{2^{q-1}-1} \sum_{\{i_1,\ldots,i_k\}\subseteq[q],k \text{ even}} \mathbb{E}[Y_{i_1,\ldots,i_k}] \geq \varepsilon. \tag{2.6}$$

Observe that $X = Y$ when G occurs, and $X = 0$ otherwise. By the law of total expectation,

$$\begin{aligned}\mathbb{E}[X] &= \mathbb{E}[X|G]\cdot\Pr[G] + \mathbb{E}[X|\overline{G}]\cdot\Pr[\overline{G}] = \mathbb{E}[X|G]\cdot\Pr[G] = \mathbb{E}[Y|G]\cdot\Pr[G] \\ &= \mathbb{E}[Y] - \mathbb{E}[Y|\overline{G}]\cdot\Pr[\overline{G}] \geq \varepsilon - 1\cdot(\varepsilon/2) = \varepsilon/2,\end{aligned}$$

where the inequality follows from (2.6), the fact that $Y \leq 1$, and Lemma 2.8. □

Fix any round of Algorithm 1 and the value of j used in this round (as defined in Step 2). Let X' be defined as X, but for this round instead of the last one. The round is *special* if $X' \geq 2^{-j}$. Let $p'_j = \Pr[X' \geq 2^{-j}]$ and $p_j = \Pr[X \geq 2^{-j}]$. Then $p'_j \geq p_j$, since the number of erasures only increases with each round. For each j, there are $k_j = \frac{8\ln 5}{2^j\varepsilon}$ rounds of Algorithm 1 that are run with this particular value of j. Since Algorithm 1 uses independent random coins for each round, the probability that no round is special is at most

$$\prod_{j=1}^{\log\frac{8}{\varepsilon}} (1-p'_j)^{k_j} \leq \prod_{j=1}^{\log\frac{8}{\varepsilon}} (1-p_j)^{k_j} \leq \frac{1}{5},$$

where the last inequality follows by Lemma 2.9 applied with $\delta = 1/5$ and $\alpha = \varepsilon/2$ and Claim 2.10. Therefore, with probability at least $\frac{4}{5}$, Algorithm 1 has a special round.

Consider a special round of Algorithm 1 and fix the value of j for this round. We show that Algorithm 1 rejects in the special round with probability at least 5/6. We call a sum $\bigoplus_{i\in I} x_i$ *violating* if the tuple $(x_i : i \in I)$ violates linearity. Since G occurred, all q points queried in Step 5 of Algorithm 1 were successfully obtained. So, the algorithm will reject as soon as it successfully obtains a violating sum. Since G occurred, there are at least $2^{q-1}-1$ distinct sums that can be queried in Step 8, all of them nonerased at the beginning of the round. Algorithm 1 makes at most $q + 4\cdot 2^j$ queries in this round, and thus the fraction of these sums erased during the round is at most

$$\begin{aligned} t\cdot\frac{q+4\cdot 2^j}{2^{q-1}-1} &\leq t\cdot\left(\frac{1}{2^{q/2}} + \frac{3\cdot 2^j}{2^{q-2}}\right) \leq \frac{t}{t\cdot\frac{50}{\varepsilon}} + \frac{12t\cdot 2^j}{t^2\cdot(\frac{50}{\varepsilon})^2} \\ &= \frac{8}{50\cdot\frac{8}{\varepsilon}} + \frac{12\cdot 8^2\cdot 2^j}{50^2 t\cdot(\frac{8}{\varepsilon})^2} \leq \frac{0.16}{2^j} + \frac{0.3072}{2^j} \leq \frac{1}{2\cdot 2^j},\end{aligned}$$

where in the first inequality we used that $\frac{q}{2^{q-1}-1} \leq \frac{1}{2^{q/2}}$ for $q \geq 9$ and that $2^{q-1}-1 \geq (4/3)\cdot 2^{q-2}$ for $q \geq 3$ (note that $q \geq 2\log(50t/\varepsilon) \geq 2\log 100 > 13$), in the second inequality we used $q \geq 2\log(50t/\varepsilon)$, and in the third inequality we used $2^j \leq \frac{8}{\varepsilon}$ and $t \geq 1$.

Since the round is special, at least a 2^{-j} fraction of the sums that can be queried in Step 8 are violating. Thus, the fraction of the sums that are violating and nonerased before each iteration of Steps 8-9 in this round is at least $2^{-j} - 2^{-j-1} = 2^{-j-1}$.

Then, each iteration of Steps 8-9 rejects with probability at least 2^{-j-1}. Since there are $4 \cdot 2^j$ iterations with independently chosen sums, the probability that the special round accepts is at most

$$(1 - 2^{-j-1})^{4\cdot 2^j} \le \mathrm{e}^{-2} \le 1/6.$$

That is, the probability that Algorithm 1 rejects in the special round is at least 5/6. Since the special round exists with probability at least $\frac{4}{5}$, Algorithm 1 rejects with probability at least $\frac{4}{5} \cdot \frac{5}{6} = \frac{2}{3}$. □

3 An online-erasure-resilient linearity tester with $O(t/\varepsilon)$ queries

In this section, we present our online-erasure-resilient linearity tester with query complexity $O(t/\varepsilon)$ (Algorithm 2) and prove Theorem 3.1 below. Theorem 3.1 together with Theorem 2.1 implies Theorem 1.1.

Theorem 3.1. *There exist a constant $c_0 \in (0,1)$ and a 1-sided error, nonadaptive, t-online-erasure-resilient ε-tester for linearity of functions $f\colon \{0,1\}^d \to \{0,1\}$ that works for $t \le c_0 \cdot \varepsilon \cdot 2^{d/4}$ and makes $O(\frac{t}{\varepsilon})$ queries.*

Algorithm 2 An Online-Erasure-Resilient Linearity Tester

Input: $\varepsilon \in (0, \frac{1}{2})$, erasure budget $t \in \mathbb{N}$, access to function $f : \{0,1\}^d \to \{0,1\}$ via a t-online-erasure oracle.

1: Let $c = 88$ and $q = \frac{ct}{\varepsilon}$.
2: **for all** $i \in [q]$:
3: Sample $x_i \sim \{0,1\}^d$ and query f at x_i.
4: **repeat** $\frac{24}{\varepsilon}$ times:
5: Sample a uniform (i, j) such that $i, j \in [q]$ and $i < j$, then query f at $x_i \oplus x_j$.
6: **Reject** if x_i, x_j, and $x_i \oplus x_j$ are nonerased and $f(x_i) + f(x_j) \ne f(x_i \oplus x_j)$.
7: **Accept.**

In the analysis of Algorithm 2, we let x_i, for all $i \in [q]$, be random variables denoting the points queried in Step 3. Recall that a pair (x, y) of domain points *violates* linearity if $f(x) + f(y) \ne f(x \oplus y)$. The proof of Theorem 3.1 crucially relies on the following lemma about the concentration of pairs violating linearity.

Lemma 3.2. *Suppose f is ε-far from linear and $q = \frac{ct}{\varepsilon}$. Let Y denote the number of pairs $(i, j) \in [q]^2$ such that $i < j$ and (x_i, x_j) violates linearity. Then* $\Pr[Y \le \frac{\varepsilon}{4} \cdot \binom{q}{2}] \le 0.25$.

We prove Lemma 3.2 in Section 3.1. In the rest of this section, we first prove an auxiliary lemma (Lemma 3.3) and then use Lemmas 3.2 and 3.3 to prove Theorem 3.1.

We say a query $u \in \{0,1\}^d$ is *successfully obtained* if it is nonerased right before it is queried, i.e., the tester obtains $f(u)$ as opposed to the symbol $\perp$.

Lemma 3.3. *Let G_1 be the (good) event that all $\binom{q}{2}$ sums $x_i \oplus x_j$ of points queried in Step 3 of Algorithm 2 are distinct, and G_2 be the (good) event that all q points x_i queried in Step 3 are successfully obtained. Then $\Pr[\overline{G_1} \cup \overline{G_2}] \leq 0.05$ for all adversarial strategies.*

Proof. First, we analyze event G_1. Consider points $x_i, x_j, x_k, x_\ell \sim \{0,1\}^d$, where $\{i,j\} \neq \{k,\ell\}$. Since these points are distributed uniformly and independently, so are the sums $x_i \oplus x_j$ and $x_k \oplus x_\ell$. The probability that two uniformly and independently sampled points $x, y \sim \{0,1\}^d$ are identical is $\frac{1}{2^d}$. Therefore, by a union bound over all pairs of sums, $\Pr[\overline{G_1}] \leq \binom{q}{2}^2 \cdot \frac{1}{2^d} \leq \frac{q^4}{4 \cdot 2^d}$.

To analyze G_2, fix an arbitrary adversarial strategy. In Steps 2-3, the algorithm makes at most q queries. Hence the oracle erases at most qt points from $\{0,1\}^d$. Since each point x_i is sampled uniformly from $\{0,1\}^d$,

$$\Pr_{x_i \sim \{0,1\}^d}[x_i \text{ is erased when queried}] \leq \frac{qt}{2^d}.$$

By a union bound over the q points sampled in Step 3, we get $\Pr[\overline{G_2}] \leq \frac{q^2 t}{2^d}$. We substitute $q = \frac{ct}{\varepsilon}$ and get

$$\Pr[\overline{G_1} \cup \overline{G_2}] \leq \frac{q^4}{4 \cdot 2^d} + \frac{q^2 t}{2^d} \leq \frac{q^4}{2^d} = \frac{c^4 t^4}{\varepsilon^4 \cdot 2^d} \leq c^4 c_0^4 \leq 0.05,$$

since $t \leq c_0 \cdot \varepsilon \cdot 2^{d/4}$, as stated in Theorem 3.1, and assuming c_0 is sufficiently small. □

Proof of Theorem 3.1. The query complexity of Algorithm 2 is evident from its description. Clearly, Algorithm 2 is nonadaptive and accepts if f is linear. Suppose now that f is ε-far from linear and fix an arbitrary adversarial strategy. We show that Algorithm 2 rejects with probability at least $\frac{2}{3}$.

We call a sum $x_i \oplus x_j$ queried in Step 5 *violating* if the pair (x_i, x_j) violates linearity. For Algorithm 2 to reject, it must sample $i, j \in [q]$ such that x_i, x_j, and $x_i \oplus x_j$ are successfully obtained and $x_i \oplus x_j$ is a violating sum. Let E_k, for all $k \in [24/\varepsilon]$, be the event that in iteration k of Step 4, the algorithm successfully obtains a violating sum. Let $E = \bigcup_{k \in [24/\varepsilon]} E_k$. We define the following good event concerning the execution of the **for** loop: $G = G_1 \cap G_2 \cap G_3$, where G_1 and G_2 are as defined in Lemma 3.3, and G_3 is the event that Y, defined in Lemma 3.2, is at least $\frac{\varepsilon}{4} \cdot \binom{q}{2}$. By a union bound and Lemmas 3.2 and 3.3, we know that $\Pr[G] \geq 0.7$. Therefore, the probability that Algorithm 2 rejects is at least

$$\Pr[E \cap G] = \Pr[G] \cdot \Pr[E|G] \geq 0.7 \cdot \Pr[E|G].$$

Suppose that G occurs for the execution of the **for** loop. Since G_2 occurred, all q points queried in Step 3 of Algorithm 2 were successfully obtained. So, the algorithm will reject as soon as it successfully obtains a violating sum. Since G_1 occurred, there are at least $\binom{q}{2}$ distinct sums that can be queried in Step 5. Since G_3 occurred, at least $\frac{\varepsilon}{4} \cdot \binom{q}{2}$ of them are violating. The total number of erasures the adversary makes over the course of the entire execution is at most $2qt$, since Algorithm 2 queries at most $2q$ points. Therefore, the fraction of sums that are nonerased and violating before each iteration of Step 5 is at least

$$\frac{\varepsilon}{4} - \frac{2qt}{\binom{q}{2}} \geq \frac{\varepsilon}{4} - \frac{5t}{q} = \frac{\varepsilon}{4} - \frac{5\varepsilon}{c} = \frac{\varepsilon}{4} - \frac{5\varepsilon}{88} > \frac{\varepsilon}{6}.$$

That is, $\Pr[E_k|G] > \varepsilon/6$ for all $k \in [24/\varepsilon]$. Observe that the indicator for the event E_k, for every fixed setting of random coins (of the algorithm and the adversary) used prior to iteration k of Step 5, is a Bernoulli random variable with the probability parameter equal to the fraction of sums that are violating and nonerased right before iteration k. Since independent random coins are used in each iteration of Step 5,

$$\Pr[\overline{E}|G] \leq \Pr\Big[\bigcap_{k\in[24/\varepsilon]} \overline{E_k} \mid G\Big] < (1-\varepsilon/6)^{24/\varepsilon} \leq \mathrm{e}^{-4}.$$

This yields that $\Pr[E \cap G] > 0.7 \cdot (1 - \mathrm{e}^{-4}) > 2/3$, completing the proof of the theorem. □

3.1 Concentration of pairs that violate linearity

In this section, we prove Lemma 3.2 on the concentration of pairs that violate linearity. The proof relies on Claim 3.4 that gives upper bounds on the fraction of pairs $(x, y) \in (\{0,1\}^d)^2$ that violate linearity and on the fraction of triples $(x, y, z) \in \{0,1\}^{d\times 3}$ such that (x, y) and (x, z) violate linearity. The upper bounds are stated as a function of the distance to linearity. The distance of a function f to linearity, denoted ε_f, is the minimum over all linear functions g over the same domain as f of $\Pr_x[f(x) \neq g(x)]$.

We remark that a tighter upper bound than (3.1) in Claim 3.4 is shown in [8, Lemma 3.2]. Since we only need a looser upper bound, we provide a proof for completeness.

Claim 3.4. *For all $f\colon \{0,1\}^d \to \{0,1\}$, the following hold:*

$$\Pr_{x,y\sim\{0,1\}^d}[(x,y) \text{ violates linearity}] \leq 3\varepsilon_f; \tag{3.1}$$

$$\Pr_{x,y,z\sim\{0,1\}^d}[(x,y) \text{ and } (x,z) \text{ violate linearity}] \leq \varepsilon_f + 4\varepsilon_f^2. \tag{3.2}$$

Proof. Let g be the closest linear function to f. Let $S \subseteq \{0,1\}^d$ be the set of points on which f and g differ, i.e., $S = \{x \in \{0,1\}^d \mid f(x) \neq g(x)\}$. Then $|S| = \varepsilon_f \cdot 2^d$.

For a pair (x, y) to violate linearity, at least one of x, y, and $x \oplus y$ must be in S. By a union bound,

$$\begin{aligned}\Pr_{x,y\sim\{0,1\}^d}[(x,y) \text{ violates linearity for } f] &\leq \Pr_{x,y\sim\{0,1\}^d}[x \in S \vee y \in S \vee x \oplus y \in S]\\ &\leq 3 \Pr_{x\sim\{0,1\}^d}[x \in S] = 3\varepsilon_f,\end{aligned}$$

completing the proof of (3.1).

To prove (3.2), let $E_{x,y,z}$ be the event that (x,y) and (x,z) violate linearity. By the law of total probability,

$$\begin{aligned}
&\Pr_{x,y,z\sim\{0,1\}^d}[E_{x,y,z}] \\
&\qquad = \Pr[E_{x,y,z} \mid x \in S]\Pr[x \in S] + \Pr[E_{x,y,z} \mid x \notin S]\Pr[x \notin S] \\
&\qquad \le 1 \cdot \varepsilon_f + \Pr[E_{x,y,z} \mid x \notin S] \\
&\qquad \le \varepsilon_f + \Pr[(y,z \in S) \vee (x \oplus y, z \in S) \vee (y, x \oplus z \in S) \vee (x \oplus y, x \oplus z \in S) \mid x \notin S] \\
&\qquad \le \varepsilon_f + 4\Pr[y,z \in S \mid x \notin S] = \varepsilon_f + 4\varepsilon_f^2,
\end{aligned}$$

where, in the last line, we used symmetry and a union bound, and then independence of x, y, and z. ◻

Proof of Lemma 3.2. We prove the lemma by applying Chebyshev's inequality to Y. For all $i, j \in [q]$, where $i < j$, let $Y_{i,j}$ be the indicator for the event that (x_i, x_j) violates linearity. Then $Y = \sum_{i<j\in[q]} Y_{i,j}$.

First, we obtain a lower bound on $\mathbb{E}[Y]$. By linearity of expectation and symmetry,

$$\mathbb{E}[Y] = \sum_{i<j\in[q]} \mathbb{E}[Y_{i,j}] = \binom{q}{2} \cdot \Pr_{x_i,x_j\sim\{0,1\}^d}[(x_i,x_j) \text{ violates linearity}] \ge \binom{q}{2}\varepsilon_f,$$

where the inequality is due to [8, Theorem 1.2].

Next, we obtain an upper bound on $\mathrm{Var}[Y]$. We have

$$\mathrm{Var}[Y] = \sum_{i<j} \mathrm{Var}[Y_{i,j}] + \sum_{\substack{i<j,k<l \\ \{i,j\}\ne\{k,\ell\}}} \mathrm{Cov}[Y_{i,j}Y_{k,l}].$$

By (3.1), we get $\mathrm{Var}[Y_{i,j}] \le \mathbb{E}[Y_{i,j}^2] = \mathbb{E}[Y_{i,j}] \le 3\varepsilon_f$. Therefore,

$$\sum_{i<j} \mathrm{Var}[Y_{i,j}] \le 3\binom{q}{2}\varepsilon_f. \tag{3.3}$$

When $\{i,j\} \cap \{k,\ell\} = \emptyset$, the random variables $Y_{i,j}$ and $Y_{k,\ell}$ are independent, and thus $\mathrm{Cov}[Y_{i,j}Y_{k,l}] = 0$. Consider the case when $|\{i,j\} \cap \{k,\ell\}| = 1$. Suppose that $i = k$. Then

$$\mathrm{Cov}[Y_{i,j}Y_{i,\ell}] = \mathbb{E}[Y_{i,j}Y_{i,\ell}] - \mathbb{E}[Y_{i,j}]\mathbb{E}[Y_{i,\ell}] \le \varepsilon_f + 4\varepsilon_f^2 - \varepsilon_f^2 = \varepsilon_f + 3\varepsilon_f^2,$$

where the inequality follows from (3.2) and [8, Theorem 1.2]. By symmetry,

$$\sum_{\substack{i<j,k<l \\ \{i,j\}\ne\{k,\ell\}}} \mathrm{Cov}[Y_{i,j}Y_{k,l}] = 6\sum_{i<j<\ell} \mathrm{Cov}[Y_{i,j}Y_{i,l}] \le 6\binom{q}{3}(\varepsilon_f + 3\varepsilon_f^2) \le 15\binom{q}{3}\varepsilon_f, \tag{3.4}$$

where the last inequality holds since $\varepsilon_f \leq \frac{1}{2}$. Combining (3.3) with (3.4), we get that $\mathrm{Var}[Y] \leq 18\binom{q}{3}\varepsilon_f$.

Finally, we use $\varepsilon_f \geq \varepsilon$, our lower bound on $\mathbb{E}[Y]$, and Chebyshev's inequality:

$$\begin{aligned}\Pr\left[Y \leq \binom{q}{2}\frac{\varepsilon}{4}\right] &\leq \Pr\left[Y \leq \binom{q}{2}\frac{\varepsilon_f}{4}\right] \leq \Pr\left[Y \leq \frac{\mathbb{E}[Y]}{4}\right] \\ &\leq \Pr\left[\left|Y - \mathbb{E}[Y]\right| \geq \frac{3}{4}\cdot\mathbb{E}[Y]\right] \leq \frac{16}{9}\cdot\frac{\mathrm{Var}[Y]}{\mathbb{E}[Y]^2} \leq \frac{16\cdot 18}{9}\cdot\frac{\binom{q}{3}\varepsilon_f}{\binom{q}{2}^2\varepsilon_f^2} \\ &= \frac{16\cdot 18}{9}\cdot\frac{2}{3}\cdot\frac{q-2}{q(q-1)\varepsilon_f} < \frac{22}{q\cdot\varepsilon_f} = \frac{22\varepsilon}{ct\varepsilon_f} \leq \frac{22}{ct} \leq \frac{1}{4},\end{aligned}$$

since $q = \frac{ct}{\varepsilon}$ and $c = 88$. This completes the proof of Lemma 3.2. □

4 A lower bound for online-erasure-resilient linearity testing

In this section, we prove Theorem 1.3 that shows that every t-online-erasure-resilient ε-tester for linearity of functions $f\colon \{0,1\}^d \to \{0,1\}$ must make more than $\log_2 t$ queries.

Proof of Theorem 1.3. The proof is via Yao's minimax principle for the online-erasures model (stated in Theorem 9.1 and Corollary 9.4). Let $\mathcal{D}^+$ be the uniform distribution over all linear Boolean functions on $\{0,1\}^d$ and $\mathcal{D}^-$ be the uniform distribution over all Boolean functions functions on $\{0,1\}^d$.

We show that a function $f \sim \mathcal{D}^-$ is $\frac{1}{4}$-far from linear with probability at least 6/7. Let g be a linear function, $f \sim \mathcal{D}^-$, and $\mathrm{dist}(f,g)$ be the fraction of domain points on which f and g differ. Then, $\mathbb{E}[\mathrm{dist}(f,g)] = \frac{1}{2}$. By the Hoeffding bound, $\Pr_{f\sim\mathcal{D}^-}[\mathrm{dist}(f,g) \leq \frac{1}{4}] \leq \mathrm{e}^{-\frac{2^d}{8}}$. By a union bound over the 2^d linear functions, $\Pr_{f\sim\mathcal{D}^-}[f \text{ is } \frac{1}{4}\text{-far from linear}] \geq 1 - 2^d\cdot\mathrm{e}^{-\frac{2^d}{8}}$. For d large enough, this probability is at least 6/7.

We fix the following strategy for a t-online-erasure oracle $\mathcal{O}$: after responding to each query, erase t sums of the form $\bigoplus_{x\in T} x$, where T is a subset of the queries made so far, choosing the subsets T in some fixed order. If at most $\log_2 t$ queries are made, the adversary erases all the sums of queried points.

Let A be a deterministic algorithm that makes $q \leq \log_2 t$ queries to the oracle $\mathcal{O}$. Assume w.l.o.g. that A does not repeat queries. We describe two random processes $\mathcal{P}^+$ and $\mathcal{P}^-$ that interact with algorithm A in lieu of oracle $\mathcal{O}$ and provide query answers consistent with a random function from $\mathcal{D}^+$ and $\mathcal{D}^-$, respectively. For each query of A, both processes $\mathcal{P}^+$ and $\mathcal{P}^-$ return $\perp$ if the value has been previously erased by $\mathcal{O}$; otherwise, they return 0 or 1 with equal probability. Thus, the distribution over query-answer histories when A interacts with $\mathcal{P}^+$ is the same as when A interacts with $\mathcal{P}^-$.

Next, we describe how the processes $\mathcal{P}^+$ and $\mathcal{P}^-$ assign values to the locations of f that were either not queried by A or erased when queried, and show that they generate $\mathcal{D}^+$ and $\mathcal{D}^-$, respectively. After A finishes its queries, $\mathcal{P}^-$ sets the remaining unassigned locations (including the erased locations) of the function to be 0 or 1 with equal probability. Clearly, $\mathcal{P}^-$ generates a function from the distribution $\mathcal{D}^-$.

To describe $\mathcal{P}^+$ fully, first let $Q \subseteq \{0,1\}^d$ denote the queries of A that are answered with a value other than $\perp$. Since $q \le \log_2 t$, by our choice of the oracle $\mathcal{O}$, the sum of any subset of vectors in Q is not contained in Q. Hence, the vectors in Q are linearly independent. Then, $\mathcal{P}^+$ completes Q to a basis B for $\{0,1\}^d$ and sets the value of f on all vectors in $B \setminus Q$ independently to 0 or 1 with equal probability.

Since B is a basis, each vector $y \in \{0,1\}^d$ can be expressed as a linear combination of vectors in B (with coefficients in $\{0,1\}$), that is, $y = \bigoplus_{x\in T} x$ for some $T \subseteq B$. The process $\mathcal{P}^+$ sets $f(y) = \sum_{x\in T} f(x)$, where addition is mod 2. The function f is linear and agrees with all values previously assigned by $\mathcal{P}^+$ to the vectors in Q. Moreover, f is distributed according to $\mathcal{D}^+$, since one can obtain a uniformly random linear function by first specifying a basis for $\{0,1\}^d$, and then setting the value of f to be 0 or 1 with equal probability for each basis vector.

Thus, $\mathcal{P}^+$ generates linear functions, $\mathcal{P}^-$ generates functions that are $\frac{1}{4}$-far from linear with probability at least $\frac{6}{7}$, and the query-answer histories for any deterministic algorithm A that makes at most $\log_2 t$ queries and runs against our t-online-erasure oracle $\mathcal{O}$ are identical under $\mathcal{P}^+$ and $\mathcal{P}^-$. Consequently, Corollary 9.4 implies the desired lower bound. ◻

5 An online-erasure-resilient quadraticity tester

In this section, we state our online-erasure-resilient quadraticity tester (Algorithm 3) and prove Theorem 1.4.

The main idea behind Algorithm 3 and its representation as a two-player game appear in Section 1.2, accompanied by explanatory figures for the case when $t = 1$. We now give a high level overview of Algorithm 3. For a function $f\colon \{0,1\}^d \to \{0,1\}$ and $x, y, z \in \{0,1\}^d$, let

$$T_f(x,y,z) = \sum_{\emptyset \ne S \subseteq \{x,y,z\}} f\Big(\bigoplus_{u\in S} u\Big),$$

where the first sum is mod 2. We say a triple (x,y,z) *violates* quadraticity if $T_f(x,y,z) = 1$. The tester of Alon et al. samples three vectors $x, y, z \in \{0,1\}^d$ uniformly and independently at random and rejects if (x,y,z) violates quadraticity. Our tester looks for the same kind of violations as the tester of Alon et al.

The main challenge in the design of our online-erasure-resilient tester is to ensure it can query all three such points and their sums in the presence of a t-online-erasure adversary. In each iteration of the **repeat** loop of Algorithm 3, the following steps are performed. For each $\ell \in [t+1]$, we first query a *reserve* of uniformly and independently sampled points $x_i^{(\ell)}$ for $i \in [(t+1)^2(2t+1)^t]$. Next, for each $\ell \in [t+1]$, we query a set of points that we visualize as being the nodes of a $(t+1)$-ary tree of depth t. There is a one-to-one correspondence between the nodes of such a tree and vectors of length up to $t+1$ over the alphabet $[t+1]$. For $m \in [t]$, we represent by $y_{(\ell,j_1,\dots,j_m)}$, the sampled point visualized as a node at depth m in the ℓ-th tree, where the j_i's specify the unique path from the root to that node in the tree. Now, for $\ell \in [t+1]$, and for each node $y_{\boldsymbol{j}}$ in the ℓ-th tree, where $\boldsymbol{j}$ is shorthand for $(\ell, j_1, \ldots, j_m)$, we associate with that node a subset $S_{\boldsymbol{j}}$ of points from the reserve, and query the points $y_{\boldsymbol{j}} \oplus x$ for each x in $S_{\boldsymbol{j}}$. The set $S_{\boldsymbol{j}}$ is a subset of a specified cardinality of the set $S_{\boldsymbol{j}^{(-1)}}$, where $S_{\boldsymbol{j}^{(-1)}}$ is the set associated with the parent node of $y_{\boldsymbol{j}}$ in the ℓ-th tree. Finally, the algorithm queries a point z sampled uniformly and independently at random from $\{0,1\}^d$, and samples a uniformly random leaf $y_{\boldsymbol{j}}$ of a uniformly random tree $\ell \in [t+1]$. The set $S_{\boldsymbol{j}}$ associated with the leaf $y_{\boldsymbol{j}}$ has, by construction, $t+1$ points in it. All the points in $S_{\boldsymbol{j}}$, again, by construction, also belong to the sets $S_{\boldsymbol{j}'}$ associated with the nodes $y_{\boldsymbol{j}'}$ on the path from the root to the leaf $y_{\boldsymbol{j}}$ of the ℓ-th tree. Our algorithm queries $y_{\boldsymbol{j}'} \oplus z$ for all such nodes $y_{\boldsymbol{j}'}$. It then samples x uniformly at random from $S_{\boldsymbol{j}}$ and queries $x \oplus z$. Finally, it samples a uniformly random node $y_{\boldsymbol{j}'}$ on the path from the root to the leaf $y_{\boldsymbol{j}}$ and queries $x \oplus y_{\boldsymbol{j}'} \oplus z$. Observe that, by design, the point $x \oplus y_{\boldsymbol{j}'}$ has already been queried in an earlier step. The algorithm rejects if all the points involved in the sum $T_f(x, y_{\boldsymbol{j}'}, z)$ are nonerased and the triple $(x, y_{\boldsymbol{j}'}, z)$ violates quadraticity.

Algorithm 3 uses the following notation. For a vector $\boldsymbol{j} = (\ell, j_1, \ldots, j_m)$, where $m \in [0,t]$ and $\ell, j_1, \ldots, j_t \in [t+1]$, we use the convention that $\boldsymbol{j} = (\ell)$ for $m = 0$. For $k \in [0,m]$, let $\boldsymbol{j}^{(k)} = (\ell, j_1, \ldots, j_k)$ be the vector containing the first $k+1$ entries of $\boldsymbol{j}$. Finally, let $\boldsymbol{j}^{(-1)}$ be the vector $\boldsymbol{j}$ with its last entry removed. If $\boldsymbol{j} = (\ell)$, then $\boldsymbol{j}^{(-1)}$ is the empty vector $\emptyset$.

Algorithm 3 A t-Online-Erasure-Resilient Quadraticity Tester

Input: $\varepsilon \in (0,1)$, access to function $f : \{0,1\}^d \longrightarrow \{0,1\}$ via a t-online-erasure oracle.

```
Let I = (t+1)^2 (2t+1)^t, J = ((t+1)^{(t+1)} − 1)/t, and α = min(ε/2, 7/(18·IJ(t+1))^{IJ(t+1)}).
repeat 4c_t/α times:                          ▷ c_t is a constant from Lemma 5.4 that depends only on t.
    for all ℓ ∈ [t+1]:
        Query f at independent x_1^(ℓ), ..., x_I^(ℓ) ~ {0,1}^d, and let S_∅ = {x_1^(ℓ), ..., x_I^(ℓ)}.
        for all integer m ∈ [0,t]:
            for all (j_1, j_2, ..., j_m) ∈ [t+1]^m:          ▷ When m = 0, the loop is run once.
                Let j = (ℓ, j_1, ..., j_m) and query f at y_j ~ {0,1}^d.      ▷ j = (ℓ) for m = 0.
                Let S_j be a uniformly random subset of S_{j^(−1)} of size (t+1)(2t+1)^{t−m}.
                Query x ⊕ y_j for all x ∈ S_j.
                Remove S_j from S_{j^(−1)}.
    Sample z ~ {0,1}^d and query f at z.
    Sample j = (ℓ, j_1, ..., j_t) ~ [t+1]^(t+1) and query f at y_{j^(m)} ⊕ z for all m ∈ [0,t].
    Suppose S_j = {x_1^(ℓ), x_2^(ℓ), ..., x_{t+1}^(ℓ)}. Sample i ~ [t+1] and query f at x_i^(ℓ) ⊕ z.
    Sample integer m ~ [0,t] and query f at x_i^(ℓ) ⊕ y_{j^(m)} ⊕ z.
    Reject if T_f(x_i^(ℓ), y_{j^(m)}, z) = 1.     ▷ All points needed for computing T_f are nonerased.
Accept.
```

Proof of Theorem 1.4. If f is quadratic, then Algorithm 3 always accepts. Suppose that f is ε-far from quadratic. Fix an adversarial strategy and a round of Algorithm 3. We show that Algorithm 3 rejects with probability $\Omega(\varepsilon)$ in this round.

Observe that Algorithm 3 makes queries of three types: *singletons* (of the form $x_i^{(\ell)}$, $y_{\boldsymbol{j}^{(m)}}$, and z), *doubles* (of the form $x_i^{(\ell)} \oplus y_{\boldsymbol{j}^{(m)}}$,$x_i^{(\ell)} \oplus z$, and $y_{\boldsymbol{j}^{(m)}} \oplus z$), and *triples* (of the form $x_i^{(\ell)} \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$), where $i \in [I]$, $\boldsymbol{j} = (\ell, j_1, \dots, j_t) \in [t+1]^{(t+1)}$, $m \in [0,t]$. We call points of the form $x_i^{(\ell)} \oplus y_{\boldsymbol{j}^{(m)}}$, $x_i^{(\ell)} \oplus z$, and $y_{\boldsymbol{j}^{(m)}} \oplus z$ *double decoys*, and points of the form $x_i^{(\ell)} \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$ *triple decoys*. We refer to double and triple decoys simply as decoys. Only some of the decoys become actual queries.

Let $\mathcal{G}$ denote the *good* event that, for the fixed round, all of the following hold:

- all singleton queries are successfully obtained;
- all double decoys of the form $x_i^{(\ell)} \oplus y_{\boldsymbol{j}^{(m)}}$ are nonerased right before $y_{\boldsymbol{j}^{(m)}}$ is queried;
- all double and triple decoys involving z (as well as $x_i^{(\ell)}$ and/or $y_{\boldsymbol{j}^{(m)}}$) are nonerased right before z is queried.

To lower bound the rejection probability of the algorithm, we consider the event that all triples of the form $(x_i^{(\ell)}, y_{\boldsymbol{j}^{(m)}}, z)$, where $i \in [I]$, $\boldsymbol{j} = (\ell, j_1, \dots, j_t) \in [t+1]^{(t+1)}$, $m \in [0,t]$, violate quadraticity, and all queries in the round are successfully obtained.

Definition 5.1 (Witness)**.** The singleton queries form a *witness* if $T_f(x_i^{(\ell)}, y_{\boldsymbol{j}^{(m)}}, z) = 1$ for all $i \in [I]$, $\boldsymbol{j} = (\ell, j_1, \dots, j_t) \in [t+1]^{(t+1)}$, $m \in [0,t]$, and, in addition, all singletons and all decoys are distinct.

Let W be the event that the singleton queries form a witness. Let α be as defined in Step 1.

Lemma 5.2 (Probability of successful singleton queries)**.** *If $f\colon \{0,1\}^d \to \{0,1\}$ is ε-far from being quadratic, then* $\Pr[W \cap \mathcal{G}] \geq \alpha/2$.

In other words, Lemma 5.2 shows that for every adversarial strategy, with probability at least $\frac{\alpha}{2}$, the tester successfully obtains singleton queries that form a witness and, in addition, right before each singleton is queried, the decoys involving that singleton are nonerased. The proof of Lemma 5.2 (in Section 5.1) relies on the following key structural result about the fraction of large structures where all triples of a certain form violate quadraticity.

Lemma 5.3 (Probability of singletons forming a violation structure)**.** *Let $I, J, t \in \mathbb{N}$. Suppose $f\colon \{0,1\}^d \to \{0,1\}$ is ε-far from being quadratic. For points $x_i^{(\ell)}, y_j^{(\ell)}, z \sim \{0,1\}^d$, where $(i,j,\ell) \in [I] \times [J] \times [t+1]$,*

$$\Pr\Bigg[\bigcap_{i\in[I],j\in[J],\ell\in[t+1]} [T_f(x_i^{(\ell)}, y_j^{(\ell)}, z) = 1]\Bigg] \geq \alpha, \tag{5.1}$$

where $\alpha = \min\left(\frac{\varepsilon}{2}, \frac{7}{(18\cdot IJ(t+1))^{IJ(t+1)}}\right)$.

We prove Lemma 5.3 in Section 5.2, building on a result of [2]. To prove Lemma 5.2, we use Lemma 5.3 with $I = (t+1)^2(2t+1)^t$ and $J = \sum_{m=0}^{t}(t+1)^m = \frac{(t+1)^{(t+1)}-1}{t}$, which is the number of nodes in each tree.

Next, in Lemma 5.4, we show that the probability of successfully obtaining all double and triple queries in the round, given the event $W \cap \mathcal{G}$, depends only on t. The proof of Lemma 5.4 appears in Section 5.3.

Lemma 5.4 (Probability of all double and triple queries being nonerased)**.** *The probability that all queries made in one round of Algorithm 3 are successfully obtained, conditioned on the event $W \cap \mathcal{G}$, is at least $1/c_t$, where c_t depends only on t.*

Let H denote the event that all queries in the round are successfully obtained, and recall that W is the event that the singleton queries form a witness. The probability that Algorithm 3 rejects in the given round is at least $\Pr[W \cap H]$. Note that

$$\Pr[W \cap H] \geq \Pr[W \cap H \cap \mathcal{G}] = \Pr[H \mid W \cap \mathcal{G}] \Pr[W \cap \mathcal{G}].$$

By Lemma 5.2 and Lemma 5.4, we have that Algorithm 3 rejects with probability at least $\frac{1}{c_t} \cdot \frac{\alpha}{2}$ for a fixed round. Thus, after $\frac{4c_t}{\alpha}$ rounds, Algorithm 3 rejects with probability at least $\frac{2}{3}$. ◻

5.1 Proof of Lemma 5.2

In this section, we prove Lemma 5.2 that bounds from below the probability that the singleton queries are successfully obtained and form a witness, and, in addition, right before each singleton is queried, the decoys involving that singleton are nonerased. We show that if the fraction of large violation structures is at least α (Lemma 5.3), then, despite the adversarial erasures, the probability of successful singleton queries, as described above, is at least $\alpha/2$.

Proof of Lemma 5.2. The key idea of the proof is to keep track of *active* witnesses during the execution of the round. To simplify notation, let $K = (I + J)(t + 1) + 1$ and denote by $u_1, \ldots, u_K$ the singleton queries made by Algorithm 3 in the given round.

Definition 5.5 (Active witness)**.** For $i \in [K]$, let $u_1, \ldots, u_i$ denote the singleton queries made by Algorithm 3 up until a given timepoint of the fixed round. We say a witness $(v_1, \ldots, v_K) \in (\{0,1\}^d)^K$ is *i-active* if

1. The first i entries of the tuple are equal to $u_1, u_2, \ldots, u_i$.
2. All decoys involving u_j, where $j \leq i$, were nonerased right before u_j was queried.

Furthermore, let A_i be a random variable denoting the number of active witnesses right after the i-th singleton query, and let B_i denote the number of active witnesses right before the i-th singleton query. Let B_1 denote the number of witnesses at the beginning of the round, such that all singletons and decoys for the witness are nonerased. Note that $\Pr[W \cap \mathcal{G}] = \Pr[A_K = 1] = \mathbb{E}[A_K]$. To lower bound $\mathbb{E}[A_K]$, we first show a lower bound on B_1, obtained in turn by a lower bound on the total fraction of witnesses. We then bound the difference between A_i and B_{i+1} and show a relationship between $\mathbb{E}[A_i]$ and $\mathbb{E}[B_i]$ for general i. All expectations in this proof are over the choice of singletons $u_1, \ldots, u_K \sim \{0,1\}^d$.

Claim 5.6. *Let α be as defined in Lemma 5.3. If f is ε-far from being quadratic then*

$$\Pr_{u_1,\ldots,u_K \sim \{0,1\}^d}[(u_1,\ldots,u_K) \text{ is a witness}] \geq \frac{7\alpha}{8}.$$

Proof. The probability that the event of Lemma 5.3 is true for the singleton queries $u_1,\ldots,u_K$ is at least α. Let S denote the set of all singletons and decoys associated with the K-tuple $(u_1,\ldots,u_K)$. The number of triple decoys associated with the K-tuple is at most $IJ(t+1)$, whereas the number of double decoys associated with the K-tuple is at most $I(J+1)(t+1)$. Therefore, $|S| \leq 3IJ(t+1)$. Any two elements in S are uniformly and independently distributed. Thus, the probability that two elements in S are identical is $\frac{1}{2^d}$. By a union bound over all pairs of elements in S, the probability that not all elements in S are distinct is at most $\frac{|S|^2}{2^d}$. For d large enough, we have $\frac{|S|^2}{2^d} \leq \frac{\alpha}{8}$. Hence, with probability at least $\alpha - \frac{\alpha}{8} = \frac{7\alpha}{8}$, the singleton queries $u_1,\ldots,u_K$ form a witness. ◻

Claim 5.7. *For all adversarial strategies, $B_1 \geq \frac{3\alpha}{4} \cdot 2^{Kd}$.*

Proof. Recall that B_1 counts the number of witnesses at the beginning of the round, such that all singletons and decoys for the witness are nonerased. Algorithm 3 makes at most $|S|$ queries in each round, so it makes at most $\frac{4c_t|S|}{\alpha}$ queries in total. Therefore, the oracle can erase at most $\frac{4t\cdot|S|\cdot c_t}{\alpha}$ points from $\{0,1\}^d$. Each erasure can deactivate at most $|S| \cdot 2^{(K-1)d}$ witnesses, since the point erased could be any one of the $|S|$ singletons and decoys associated with the witness. By Claim 5.6, we obtain $B_1 \geq \frac{7\alpha}{8} \cdot 2^{Kd} - \frac{4t\cdot|S|^2\cdot c_t}{\alpha} \cdot 2^{(K-1)d}$. Since $|S| \leq 3IJ(t+1)$, for d large enough, we have $\frac{4t\cdot|S|^2\cdot c_t}{\alpha} \cdot 2^{(K-1)d} \leq \frac{\alpha}{8} \cdot 2^{Kd}$. Therefore, $B_1 \geq \frac{3\alpha}{4} \cdot 2^{Kd}$. ◻

Claim 5.8. *For all $i \in [K-1]$ and all adversarial strategies,*

$$A_i - B_{i+1} \leq (t+1)^2(2t+1)^t \cdot |S| \cdot 2^{(K-1-i)d}.$$

Proof. Observe that Algorithm 3 makes at most $(t+1)(2t+1)^t$ decoy queries between any two singleton queries (this is the number of double queries made in Step 9 for $m=0$). Therefore, in the period right after the i-th singleton query and before the $(i+1)$-st singleton query, the oracle can perform at most $((t+1)(2t+1)^t+1)\cdot t \leq (t+1)^2(2t+1)^t$ erasures. Let $u_1,u_2,\ldots,u_i$ be the first i singleton queries of the algorithm. For each erasure, the oracle can deactivate at most $|S| \cdot 2^{(K-1-i)d}$ witnesses whose first i entries are $u_1,u_2,\ldots,u_i$, by erasing one of the remaining singletons or decoys. Therefore, the number of active witnesses between right after the i-th singleton query and right before the $(i+1)$-st singleton query can decrease by at most $(t+1)^2(2t+1)^t \cdot |S| \cdot 2^{(K-1-i)d}$. ◻

Claim 5.9. *For all $i \in [K]$, it holds that $\mathbb{E}[A_i] = \frac{1}{2^d}\mathbb{E}[B_i]$.*

Proof. For $v \in \{0,1\}^d$, let $B_{i,v}$ denote the number of witnesses that are active right before the i-th singleton query and whose i-th entry is equal to v. Then, $B_i = \sum_{v\in\{0,1\}^d} B_{i,v}$. Let $\mathbf{1}(v)$ be the indicator random variable for the event that the i-th singleton query is equal to v. Then,

$$\mathbb{E}[A_i] = \mathbb{E}\Big[\sum_{v\in\{0,1\}^d} B_{i,v}\mathbf{1}(v)\Big] = \mathbb{E}[\mathbf{1}(v)]\mathbb{E}\Big[\sum_{v\in\{0,1\}^d} B_{i,v}\Big] = \frac{1}{2^d}\mathbb{E}[B_i]\,. \qquad \square$$

We combine the claims above to complete the proof of the lemma. By Claim 5.9,

$$\begin{aligned}\mathbb{E}[A_K] &= \frac{1}{2^d}\mathbb{E}[B_K] = \frac{1}{2^d}\mathbb{E}[A_{K-1} + B_K - A_{K-1}] = \frac{1}{2^d}\mathbb{E}[A_{K-1}] - \frac{1}{2^d}\mathbb{E}[A_{K-1} - B_K]\\ &= \frac{1}{2^{2d}}\mathbb{E}[B_{K-1}] - \frac{1}{2^d}\mathbb{E}[A_{K-1} - B_K] = \cdots = \frac{1}{2^{Kd}}\mathbb{E}[B_1] - \sum_{i=1}^{K-1}\frac{\mathbb{E}[A_i - B_{i+1}]}{2^{(K-i)d}}.\end{aligned}$$

By Claim 5.7,

$$\frac{1}{2^{Kd}}\mathbb{E}[B_1] \geq \frac{3\alpha}{4}.$$

In addition, Claim 5.8 yields that

$$\begin{aligned}\sum_{i=1}^{K-1}\frac{\mathbb{E}[A_i - B_{i+1}]}{2^{(K-i)d}} &\leq \sum_{i=1}^{K-1}\frac{(t+1)^2(2t+1)^t\cdot|S|\cdot 2^{(K-1-i)d}}{2^{(K-i)d}}\\ &\leq \frac{K\cdot(t+1)^2(2t+1)^t\cdot|S|}{2^d} \leq \frac{\alpha}{4},\end{aligned}$$

where the last inequality holds for large enough d. We obtain that

$$\Pr[W\cap\mathcal{G}] = \mathbb{E}[A_K] \geq \frac{3\alpha}{4} - \frac{\alpha}{4} = \frac{\alpha}{2}\,. \qquad \square$$

5.2 Proof of Lemma 5.3

In this section, we prove Lemma 5.3 on the fraction of large violation structures for which all triples $(x_i^{(\ell)}, y_j^{(\ell)}, z)$, where $(i,j,\ell)\in[I]\times[J]\times[t+1]$, violate quadraticity. Our proof builds on a result of [2].

Proof of Lemma 5.3. Let η denote the fraction of violating triples for f, i.e.,

$$\eta := \Pr_{x,y,z\sim\{0,1\}^d}[T_f(x,y,z) = 1].$$

The distance of f to quadraticity, denoted by ε_f, is the minimum of $\Pr_x[f(x)\neq g(x)]$ over all quadratic functions g over the same domain as f. Using this notation, we state a result from [2] for the special case of quadraticity.

Claim 5.10 (Theorem 1 of [2])**.** *For all f, we have $\eta \geq \min(\frac{7}{3}\varepsilon_f, \frac{1}{40})$.*

Let η' denote the left-hand side of (5.1), that is, the probability that for $x_i^{(\ell)}, y_j^{(\ell)}, z \sim \{0,1\}^d$, all triples $(x_i^{(\ell)}, y_j^{(\ell)}, z)$ are violating, where $(i,j,\ell) \in [I]\times[J]\times[t+1]$. Claim 5.11 lower bounds η' in terms of η for all values of ε_f. Claim 5.13 lower bounds η' for small values of ε_f. We combine these results and use Claim 5.10 to conclude the proof of the lemma.

Claim 5.11. *For all f and points $x_i^{(\ell)}, y_j^{(\ell)}, z \sim \{0,1\}^d$, where $i \in [I]$, $j \in [J]$, $\ell \in [t+1]$, it holds that*

$$\Pr\Big[\bigcap_{i\in[I],j\in[J],\ell\in[t+1]} [T_f(x_i^{(\ell)}, y_j^{(\ell)}, z) = 1]\Big] \geq \eta^{IJ(t+1)}.$$

Proof. The proof uses the Hölder's inequality as its key ingredient.

Claim 5.12 (Hölder's inequality). *Let $p, q \geq 1$ such that $\frac{1}{p}+\frac{1}{q} = 1$. For all vectors $(a_1,\dots,a_n)$, $(b_1,\dots,b_n) \in \mathbb{R}^n$,*

$$\sum_{i\in[n]} |a_i b_i| \leq \Big(\sum_{i\in[n]} |a_i|^p\Big)^{1/p}\Big(\sum_{i\in[n]} |b_i|^q\Big)^{1/q}.$$

For $x_i^{(\ell)}, y_j^{(\ell)}, z \sim \{0,1\}^d$, let $T_{ij}^{(\ell)}$ be the event $T_f(x_i^{(\ell)}, y_j^{(\ell)}, z) = 1$. Then

$$\begin{aligned}
\Pr\Big[\bigcap_{i\in[I],j\in[J],\ell\in[t+1]} T_{ij}^{(\ell)}\Big] &= \sum_{u\in\{0,1\}^d} \Pr\Big[\bigcap_{i\in[I],j\in[J],\ell\in[t+1]} T_{ij}^{(\ell)} \mid z = u\Big]\Pr[z=u] \\
&= \frac{1}{2^d}\sum_{u\in\{0,1\}^d} \Pr\Big[\bigcap_{i\in[I],j\in[J],\ell\in[t+1]} T_{ij}^{(\ell)} \mid z = u\Big] \\
&= \frac{1}{2^d}\sum_{u\in\{0,1\}^d} \Pr\Big[\bigcap_{i\in[I],j\in[J]} T_{ij}^{(1)} \mid z = u\Big]^{t+1} \\
&\geq \Big(\frac{1}{2^d}\sum_{u\in\{0,1\}^d} \Pr\Big[\bigcap_{i\in[I],j\in[J]} T_{ij}^{(1)} \mid z = u\Big]\Big)^{t+1} \\
&= \Pr\Big[\bigcap_{i\in[I],j\in[J]} T_{ij}^{(1)}\Big]^{t+1},
\end{aligned}$$

where the first equality holds by the law of total probability; the third equality holds because, conditioned on z taking a specific value, the events $\bigcap_{i\in[I],j\in[J]} T_{ij}^{(\ell)}$ for $\ell \in [t+1]$ are independent and have the same probability; the inequality follows from the Hölder's inequality.

We use a similar argument to obtain

$$\Pr\Big[\bigcap_{i\in[I],j\in[J]} T_{ij}^{(1)}\Big] \geq \Big(\Pr\Big[\bigcap_{j\in[J]} T_{1j}^{(1)}\Big]\Big)^{I},$$

where we condition on the values of $y_1^{(1)}, \ldots, y_J^{(1)}$, and z. Similarly, by conditioning on the values of $x_1^{(1)}$ and z, we obtain

$$\Pr\Big[\bigcap_{j\in[J]} T_{1j}^{(1)}\Big] \geq (\Pr[T_{11}^{(1)}])^J.$$

Since $\Pr[T_{11}^{(1)}] = \eta$, the claim follows. $\square$

Next, we consider the case when ε_f is small. For $u_1, u_2, u_3 \in \{0,1\}^d$, let $\mathrm{span}(u_1, u_2, u_3)$ be the set of points $\bigoplus_{i\in T} u_i$ for $\emptyset \neq T \subseteq [3]$ and

$$S = \bigcup_{i\in[I], j\in[J], \ell\in[t+1]} \mathrm{span}(x_i^{(\ell)}, y_j^{(\ell)}, z).$$

The set S has $(I+J)(t+1)+1$ singletons, at most $I(J+1)(t+1)$ double sums, and at most $IJ(t+1)$ triple sums. Therefore, $|S| \leq 3IJ(t+1)$.

Claim 5.13. *Suppose $\varepsilon_f \leq \frac{1}{2|S|}$. Then $\eta' \geq \frac{\varepsilon_f}{2}$.*

Proof. Let g be a closest quadratic function to f. Any two elements of S are uniformly and independently distributed in $\{0,1\}^d$. Then, for $x_i^{(\ell)}, y_j^{(\ell)}, z \sim \{0,1\}^d$, we have

$$\begin{aligned} \eta' &\geq \Pr[f(z) \neq g(z) \text{ and } f(u) = g(u)\ \forall\, u \in S \setminus \{z\}] \\ &\geq \Pr[f(z) \neq g(z)] - \sum_{u\in S\setminus\{z\}} [f(z) \neq g(z) \text{ and } f(u) \neq g(u)] \\ &\geq \varepsilon_f - (|S|-1)\varepsilon_f^2 = \varepsilon_f(1-(|S|-1)\varepsilon_f). \end{aligned}$$

If $\varepsilon_f \leq \frac{1}{2|S|}$, then $1-(|S|-1)\varepsilon_f \geq 1 - |S|\cdot\frac{1}{2|S|} \geq \frac{1}{2}$, which concludes the proof. $\square$

Suppose $\frac{1}{2|S|} \leq \varepsilon_f \leq \frac{3}{7\cdot 40}$. In this case, by Claim 5.10 and Claim 5.11 and using the fact that $|S| \leq 3IJ(t+1)$, we have

$$\eta' \geq \left(\frac{7}{3}\varepsilon_f\right)^{IJ(t+1)} \geq \left(\frac{7}{6\cdot|S|}\right)^{IJ(t+1)} \geq \frac{7}{(18\cdot IJ(t+1))^{IJ(t+1)}}.$$

Finally, if $\varepsilon_f \geq \frac{1}{40}$, then again by Claim 5.10 and Claim 5.11, we have $\eta' \geq \frac{1}{40^{(IJ(t+1))}}$. We have obtained that $\eta' \geq \min\big(\frac{\varepsilon}{2}, \frac{7}{(18\cdot IJ(t+1))^{IJ(t+1)}}\big)$. $\square$

5.3 Proof of Lemma 5.4

In this section, we prove Lemma 5.4, which shows that conditioned on the good event $W \cap \mathcal{G}$, all queries in one round of Algorithm 3 are successfully obtained. In particular, this probability only depends on the per-query erasure budget t.

Proof of Lemma 5.4. We first prove a lower bound on the probability that the queries made in Step 9 are successfully obtained.

Claim 5.14. *Conditioned on the event $W \cap \mathcal{G}$, the probability that in one execution of Step 9 at level $m \in [0, t]$, all queries in the step are successfully obtained is at least*

$$\left(\frac{1}{(t^2+t)(2t+1)^{t-m}+1}\right)^{(t+1)(2t+1)^{t-m}}.$$

Proof. Fix the values of $\ell, m, j_1, \ldots, j_m$ for the given execution of Step 9 and let $\boldsymbol{j} = (\ell, j_1, \ldots, j_m)$. We can assume, without loss of generality, that Step 6 considers the tuples $(j_1, \ldots, j_m)$ in the lexicographic order. When $j_m = 1$, then right before Step 9 is executed, we have $|S_{\boldsymbol{j}^{(-1)}}| = (t+1)(2t+1)^{t-m+1}$. The size of the set $S_{\boldsymbol{j}^{(-1)}}$ decreases as the value of j_m increases, so that for $j_m = t+1$, we have

$$\begin{aligned}|S_{\boldsymbol{j}^{(-1)}}| &= (t+1)(2t+1)^{t-m+1} - t(t+1)(2t+1)^{t-m}\\ &= (t+1)(2t+1)^{t-m}(2t+1-t) = (t+1)^2(2t+1)^{t-m}.\end{aligned}$$

Thus, right before Step 9 is executed, the size of $S_{\boldsymbol{j}^{(-1)}}$ is at least $(t+1)$ times the size of the subset $S_{\boldsymbol{j}}$.

Let s denote the size of $S_{\boldsymbol{j}^{(-1)}}$ right before the execution of Step 8. In Step 8 we sample a uniformly random subset $S_{\boldsymbol{j}}$ of $S_{\boldsymbol{j}^{(-1)}}$ of size $s' = (t+1)(2t+1)^{t-m}$ and in Step 9 we query $x \oplus y_{\boldsymbol{j}}$ for all $x \in S_{\boldsymbol{j}}$. Conditioned on the event $W \cap \mathcal{G}$, all sums $x \oplus y_{\boldsymbol{j}}$, where $x \in S_{\boldsymbol{j}}$, are distinct and nonerased right before $y_{\boldsymbol{j}}$ is queried. On the i-th query of Step 9, the tester selects a uniformly random x out of $s-(i-1)$ elements. Right before the i-th query of the tester, the oracle can have erased at most ti of the sums $x \oplus y_{\boldsymbol{j}}$, for $x \in S_{\boldsymbol{j}^{(-1)}}$, since the adversary is not aware of the points belonging to $S_{\boldsymbol{j}}$ until their sums with $y_{\boldsymbol{j}}$ are queried in Step 9. Therefore, the probability that the tester successfully obtains the sum on its i-th query is at least $1 - \frac{ti}{s-i+1}$, where $i \in [s']$. We argued that, for all values of j_m, right before the execution of Step 9, it always holds that $s \geq (t+1)s'$. Therefore, the probability that the tester successfully obtains a sum is always positive. In particular, using the bound $s \geq (t+1)s'$, the probability that all queries in Step 9 are successfully obtained is at least

$$\prod_{i=1}^{s'}\left(1 - \frac{ti}{s-i+1}\right) \geq \left(1 - \frac{ts'}{s-s'+1}\right)^{s'} \geq \left(\frac{1}{(t+1)s'-s'+1}\right)^{s'} = \left(\frac{1}{s't+1}\right)^{s'}.$$

Substituting $s' = (t+1)(2t+1)^{t-m}$ concludes the proof. ◻

In Steps 4-11, the tester makes only singleton and double queries. Conditioned on $W \cap \mathcal{G}$, all singleton queries are successfully obtained. All double queries are made in Step 9. By Claim 5.14, the probability that all double queries in Step 9 are successfully obtained depends only on t.

Before z is queried in Step 11, conditioned on the event $W \cap \mathcal{G}$, all double and triple sums of the form $y_{\boldsymbol{j}^{(m)}} \oplus z$, $x \oplus z$, and $x \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$, where $m \in [0, t]$, $\boldsymbol{j} = (\ell, j_1 \ldots, j_t) \in [t+1]^{(t+1)}$, and $x \in S_{\boldsymbol{j}}$, are distinct and nonerased. After z is queried, the oracle can perform at most t erasures. Therefore, there exists $\ell \in [t+1]$, say $\ell = 1$, such that all double and triple sums $y_{\boldsymbol{j}^{(m)}} \oplus z$, $x \oplus z$, and $x \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$, where $m \in [0,t]$, $\boldsymbol{j} = (1, j_1 \dots, j_t) \in [t+1]^{(t+1)}$, and $x \in S_{\boldsymbol{j}}$, are nonerased after z is queried. With probability $\frac{1}{t+1}$, the tester samples $\ell = 1$ in Step 12. By a similar reasoning, with probability $\frac{1}{(t+1)^t}$ the tester samples values of $j_1, \dots, j_t$ in Step 12, say $j_1 = \dots = j_t = 1$, such that all queries $y_{\boldsymbol{j}^{(m)}} \oplus z$, where $m \in [0,t]$, are successfully obtained. In addition, right before $y_{\boldsymbol{j}^{(t)}} \oplus z$ is queried, all sums of the form $x \oplus z$ and $x \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$, where $\boldsymbol{j} = (1, \dots, 1, j_t) \in [t+1]^{(t+1)}$, $m \in [0,t]$, and $x \in S_{\boldsymbol{j}}$, are nonerased.

After $y_{\boldsymbol{j}^{(t)}} \oplus z$ is queried, the oracle performs at most t erasures, so there exists $x \in S_{\boldsymbol{j}}$ such that $x \oplus z$ and $x \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$, where $\boldsymbol{j} = (1, 1, \dots, 1)$ and $m \in [0,t]$, are nonerased. With probability $\frac{1}{t+1}$, the tester samples this x in Step 13, and with probability $\frac{1}{t+1}$, it samples $m \in [0,t]$ such that the triple sum $x \oplus y_{\boldsymbol{j}^{(m)}} \oplus z$ is successfully obtained in Step 14. Thus, with probability $\frac{1}{(t+1)^{t+3}}$, all queries in Steps 12-14 are successfully obtained. Therefore the probability that all queries in one round of Algorithm 3 are successfully obtained, conditioned on $W \cap \mathcal{G}$, is $1/c_t$, where c_t is a constant depending only on t. ◻

6 Online-erasure-resilient sortedness testing

In this section, we prove Theorem 1.5 on the impossibility of general online-erasure-resilient sortedness testing and Theorem 1.6 on online-erasure-resilient sortedness testing of sequences with few distinct values.

6.1 Impossibility of online-erasure-resilient sortedness testing

In this section, we prove Theorem 1.5 which shows that online-erasure-resilient testing of sortedness of integer sequences is impossible.

Proof of Theorem 1.5. By Yao's principle (see Corollary 9.4), it is enough to give a pair of distributions, one over monotone functions over $[n]$ and the other one over functions over $[n]$ that are far from monotone, and an adversarial strategy, such that there is no deterministic algorithm that can distinguish the distributions with high probability, when given access to them via a 1-online-erasure oracle that uses the given strategy.
Let $n \in \mathbb{N}$ be even. Consider the following distributions on functions $f : [n] \to [n]$.

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

- $f(2i-1) \leftarrow 2i-1$ and $f(2i) \leftarrow 2i-1$, with probability $1/3$.
- $f(2i-1) \leftarrow 2i-1$ and $f(2i) \leftarrow 2i$, with probability $1/3$.
- $f(2i-1) \leftarrow 2i$ and $f(2i) \leftarrow 2i$, with probability $1/3$.

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

- $f(2i-1) \leftarrow 2i$ and $f(2i) \leftarrow 2i-1$, with probability $1/3$.
- $f(2i-1) \leftarrow 2i-1$ and $f(2i) \leftarrow 2i$, with probability $2/3$.

Every function sampled from the distribution $\mathcal{D}^+$ is monotone.

For a function sampled from the distribution $\mathcal{D}^-$, the expected number of index pairs $(2i-1, 2i)$ such that the respective function values are $2i$ and $2i-1$ is equal to $n/6$. By a Chernoff bound, the probability that the number of such index-pairs is less than $n/12$ is at most $\exp(-n/48)$. In other words, with probability at least $1-\exp(-n/48)$, a function sampled from the $\mathcal{D}^-$ distribution is $1/12$-far from being monotone.

Consider a deterministic adaptive algorithm T with two-sided error for online-erasure-resilient $\frac{1}{12}$-testing of monotonicity of functions from $[n]$ to $[n]$. Assume that T is given access to a function sampled from $\mathcal{D}^+$ or $\mathcal{D}^-$ with equal probability, where the access is via a 1-online-erasures oracle. The oracle uses the following strategy. For each $i \in [n/2]$, if T queries the point $2i-1$, then the oracle erases the function value at the point $2i$; if T queries the point $2i$ then the oracle erases the function value at the point $2i-1$. The oracle erases exactly one function value between queries. Given this strategy, the tester T never sees a violation to monotonicity.

The distribution of function values restricted to any particular index is identical for distributions $\mathcal{D}^+$ and $\mathcal{D}^-$. Hence, the tester T cannot distinguish the distributions, no matter how many queries it makes. □

6.2 Online-erasure-resilient testing of sortedness with small r

In this section, we prove Theorem 1.6 by showing that $O(\frac{\sqrt{r}}{\varepsilon})$ uniform queries are sufficient for t-online-erasure-resilient ε-testing of sortedness, when r, the number of distinct values in the sequence, is small. Pallavoor et al. [53] gave a 1-sided error ε-tester for sortedness of real-valued sequences containing at most r distinct values that makes $O(\frac{\sqrt{r}}{\varepsilon})$ uniform and independent queries. The proof of [53, Theorem 1.5] is the starting point for our analysis.

Proof of Theorem 1.6. We call $(u,v) \in [n]^2$ a *violating* pair if $u < v$ and $f(u) > f(v)$. The tester, given input $\varepsilon \in (0,1)$ and oracle access to a function f with image size at most r, queries f on $\frac{2c\sqrt{r}}{\varepsilon}$ points selected uniformly and independently and rejects if and only if it queries $u, v \in [n]$ such that (u,v) is a violating pair. Since the decision made by the tester does not depend on the specific values being queried, but only on the ordering of the queried values, we can assume without loss of generality that the input function is of the form $f : [n] \to [r]$. Clearly, the tester accepts all sorted functions. We show that for $c = 32$, if f is ε-far from sorted, the tester rejects with probability at least $\frac{2}{3}$.

Suppose f is ε-far from sorted. Define the violation graph $G([n], E)$ to be the graph whose edges are the violating pairs $(u,v) \in [n]^2$. Dodis et al. [28, Lemma 7] show that G has a matching M of size at least $\varepsilon n/2$. For each range value $i \in [r]$, let M_i be the set of edges $(u,v) \in M$ whose lower endpoint u has value i, i.e., $f(u) = i$. Let L_i consist of the $|M_i|/2$ smallest indices amongst the lower endpoints of the edges in M_i. Let U_i consist of the $|M_i|/2$ largest indices amongst the upper endpoints of the edges in M_i. It is not hard to see that, for every $i \in [r]$, all pairs in $L_i \times U_i$ are violating. By construction,

$$\sum_{i\in[r]} |L_i| = \sum_{i\in[r]} |U_i| = \sum_{i\in[r]} |M_i|/2 = \frac{|M|}{2} \geq \frac{\varepsilon n}{4}. \tag{6.1}$$

We show that with probability at least $\frac{2}{3}$, for some $i \in [r]$, the tester samples a nonerased element both from L_i and from U_i, in which case it rejects.

Our proof relies on the following claim from [36], for which we first introduce some notation. Let $L_1, \dots, L_r$ be disjoint subsets of $[n]$. Let $p_i = \frac{|L_i|}{n}$ and $\rho = \sum_{i\in[r]} p_i$. Suppose we sample $c\sqrt{r}/\rho$ uniform points from $[n]$. Let I be a random variable denoting the set of indices $i \in [r]$ such that the sample contains at least one element from the set L_i.

Claim 6.1 (Claim 1 of [36])**.** *With probability* $1 - \mathrm{e}^{-c}$ *over the choice of the sample,*

$$\sum_{i\in I} p_i \geq \frac{\rho}{\sqrt{r}}.$$

Split the sample into two parts of size $\frac{c\sqrt{r}}{\varepsilon}$ each. Let $L'_i \subseteq L_i$ be the set of nonerased indices of L_i after the tester queries the first part of the sample, and define U'_i similarly. Let $p'_i = \frac{|L'_i|}{n}$, $q'_i = \frac{|U'_i|}{n}$, and $\rho' = \sum_{i\in[r]} \min(p'_i, q'_i)$. Without loss of generality assume $p'_i \leq q'_i$ for all $i \in [r]$. The adversary performs at most $\frac{c\sqrt{r}t}{\varepsilon}$ erasures after the first part of the sample is queried, therefore

$$\rho' = \sum_{i\in[r]} \frac{|L'_i|}{n} \geq \left(\sum_{i\in[r]} \frac{|L_i|}{n} \right) - \frac{c\sqrt{r}t}{\varepsilon n} \geq \frac{\varepsilon}{4} - \frac{c\sqrt{r}t}{\varepsilon n} \geq \frac{\varepsilon}{8},$$

where we use (6.1) and the assumption that $\sqrt{r} \leq r \leq \frac{\varepsilon^2 n}{8c^2 t}$. Let I' be a random variable denoting the set of indices $i \in [r]$ such that at least one element of L'_i is contained in the first part of the sample. By Claim 6.1, with probability $1 - \mathrm{e}^{-\frac{c}{8}}$ over the first part of the sample,

$$\sum_{i\in I'} p'_i \geq \frac{\varepsilon}{8\sqrt{r}}. \tag{6.2}$$

The probability that the second part of the sample does not include a nonerased element from $\cup_{i\in[r]} U'_i$ is

$$\left(1 - \sum_{i\in[r]} q'_i + \frac{c\sqrt{r}t}{\varepsilon n}\right)^{\frac{c\sqrt{r}}{\varepsilon}} \leq \left(1 - \frac{\varepsilon}{8\sqrt{r}} + \frac{\varepsilon}{16\sqrt{r}}\right)^{\frac{c\sqrt{r}}{\varepsilon}} \leq \mathrm{e}^{-\frac{c}{16}},$$

where we use (6.2) to lower bound $\sum_{i\in[r]} q'_i$, and the assumption that $r < \frac{\varepsilon^2 n}{ct}$ to upper bound the number of erasures. Therefore, the probability that for some $i \in [r]$ the tester samples a nonerased element both from L_i and U_i and rejects is at least

$$(1 - \mathrm{e}^{-\frac{c}{8}}) \cdot (1 - \mathrm{e}^{-\frac{c}{16}}) \geq \frac{2}{3},$$

where the inequality holds for $c = 32$. □

7 Impossibility of online-erasure-resilient Lipschitz testing

In this section, we prove Theorem 1.7. We start by showing that 1-online-erasure-resilient Lipschitz testing of functions $f : [n] \to \{0, 1, 2\}$ is impossible.

Proof of Theorem 1.7, part 1. We prove the theorem using Yao's principle. The proof is analogous to that of Theorem 1.5 once the hard distributions are described. In particular, the strategy of the oracle is identical to that used in the proof of Theorem 1.5.

Let $n \in \mathbb{N}$ be a multiple of 4. The following distributions are over functions $f : [n] \to \{0, 1, 2\}$.

Distribution $\mathcal{D}^+$: Independently, for all $i \in [n]$ such that $i \bmod 4 \equiv 1$:

- $f(i) \leftarrow 0$ and $f(i+1) \leftarrow 1$, with probability $1/2$.
- $f(i) \leftarrow 1$ and $f(i+1) \leftarrow 2$, with probability $1/2$.

Independently, for all $i \in [n]$ such that $i \bmod 4 \equiv 3$:

- $f(i) \leftarrow 1$ and $f(i+1) \leftarrow 0$, with probability $1/2$.
- $f(i) \leftarrow 2$ and $f(i+1) \leftarrow 1$, with probability $1/2$.

Distribution $\mathcal{D}^-$: Independently, for all i such that $i \bmod 4 \equiv 1$:

- $f(i) \leftarrow 0$ and $f(i+1) \leftarrow 2$, with probability $1/2$.
- $f(i) \leftarrow 1$ and $f(i+1) \leftarrow 1$, with probability $1/2$.

Independently, for all i such that $i \bmod 4 \equiv 3$:

- $f(i) \leftarrow 2$ and $f(i+1) \leftarrow 0$, with probability $1/2$.
- $f(i) \leftarrow 1$ and $f(i+1) \leftarrow 1$, with probability $1/2$.

Every function sampled from the distribution $\mathcal{D}^+$ is Lipschitz. For a function sampled from the distribution $\mathcal{D}^-$, the expected distance to being Lipschitz is $1/4$. The rest of the proof is similar to that of Theorem 1.5. ☐

Next, we prove the second part of Theorem 1.7 showing that 1-online-erasure-resilient Lipschitz testing of functions $f : \{0, 1\}^d \to \{0, 1, 2\}$ is impossible.

Proof of Theorem 1.7, part 2. The proof is analogous to the proof of the first part of the theorem once the hard distributions are described.

The following distributions are over functions $f : \{0, 1\}^d \to \{0, 1, 2\}$. Let $e_1 \in \{0, 1\}^d$ denote the first standard basis vector.

Distribution $\mathcal{D}^+$: Independently, for all $x \in \{0,1\}^d$ such that $x_1 = 0$:

- $f(x) \leftarrow 0$ and $f(x \oplus e_1) \leftarrow 1$, with probability $1/2$.
- $f(x) \leftarrow 1$ and $f(x \oplus e_1) \leftarrow 2$, with probability $1/2$.

Distribution $\mathcal{D}^-$: Independently, for all $x \in \{0,1\}^d$ such that $x_1 = 0$:

- $f(x) \leftarrow 0$ and $f(x \oplus e_1) \leftarrow 2$, with probability $1/2$.
- $f(x) \leftarrow 1$ and $f(x \oplus e_1) \leftarrow 1$, with probability $1/2$.

Every function sampled from the distribution $\mathcal{D}^+$ is Lipschitz. For a function sampled from the distribution $\mathcal{D}^-$, the expected distance to being Lipschitz is $1/4$. The rest of the proof is similar to that of Theorem 1.5. This completes the proof of Theorem 1.7. ◻

8 Computation in the presence of online corruptions

This section discusses implications of our results to computation in the presence of online corruptions.

8.1 Online-corruption-resilience from online-erasure-resilience in testing

In this section, we prove Lemma 1.8, showing that an algorithm that accesses its input via an online-erasure oracle and, with high probability, encounters no erasures and outputs a correct answer, is also correct with high probability for all adversarial strategies (and for the same computational task) when it accesses its input via an online-corruption oracle.

Proof of Lemma 1.8. Fix an input function f and a strategy of the t-online-corruption oracle for this function. Let $\mathcal{O}^{(c)}$ denote an oracle that uses this strategy. Map the strategy of $\mathcal{O}^{(c)}$ to a strategy for a t-online-erasure oracle $\mathcal{O}^{(e)}$, so that whenever $\mathcal{O}^{(c)}$ modifies the value at a point x to $\mathcal{O}^{(c)}(x) \neq f(x)$, the online-erasure oracle erases the value at the same point, i.e., it sets $\mathcal{O}^{(e)}(x) = \perp$.

Fix the random coins of an execution of T, where access to f is via $\mathcal{O}^{(e)}$, and for which all queries are successfully obtained and T outputs a correct answer. Let $T^{(e)}$ be the execution of T with those coins. Similarly, let $T^{(c)}$ be the execution of T with the same coins, when access to f is via $\mathcal{O}^{(c)}$. Since $T^{(e)}$ encounters no erasures, $T^{(c)}$ will make the same queries as $T^{(e)}$ and get the same query answers. Consequently, $T^{(c)}$ will also output the same (correct) answer as $T^{(e)}$. Therefore, when T accesses f via an online-corruption oracle, it outputs a correct answer with probability at least $2/3$. ◻

8.2 Online-corruption-resilient linearity tester

In this section, we show the existence of a 2-sided error, online-corruption-resilient tester for linearity.

Corollary 8.1. *There exists $c_0 \in (0,1)$ and a nonadaptive, t-online-corruption-resilient ε-tester for linearity of functions $f: \{0,1\}^d \to \{0,1\}$ that works for $t \le c_0 \cdot \varepsilon^{5/4} \cdot 2^{d/4}$ and makes $O(\frac{1}{\varepsilon}\log\frac{t}{\varepsilon})$ queries.*

Proof. Consider Algorithm 1, modified to use $q = 2\log\frac{Ct}{\varepsilon^2}$ for $C = 3000$. Note that the larger the size q of the reserve, the higher is the probability that Algorithm 1 detects a violation to linearity. For the choice of $q = 2\log\frac{3000t}{\varepsilon^2}$, Algorithm 1 is also a 1-sided error online-erasure-resilient tester. Furthermore, for this choice of q, Algorithm 1 rejects with probability at least 5/6 (as opposed to 2/3) when f is ε-far from linear.

By Lemma 1.8, it remains to show that, for all adversarial strategies, Algorithm 1 queries an erased point during its execution with probability at most 1/6. Fix an adversarial strategy and one round of Algorithm 1. (Recall that each iteration of the outer **repeat** loop in Steps 3-9 is called a *round*). Let G be the event defined in Lemma 2.8 for the fixed round. Then $\Pr[\overline{G}] \le \frac{\varepsilon}{800}$.

Let B denote the event that the algorithm queries an erased point during the fixed round. Then

$$\Pr[B] = \Pr[B \mid G]\Pr[G] + \Pr[B \mid \overline{G}]\Pr[\overline{G}] \le \Pr[B \mid G] + \Pr[\overline{G}].$$

To upper bound $\Pr[B \mid G]$ it suffices to upper bound the probability that the algorithm queries an erased sum in some iteration of Step 8 of the fixed round. Since G occurred, there are at least $2^{q-1}-1$ distinct sums that can be queried in Step 8, all of them nonerased at the beginning of the round. Algorithm 1 makes at most $q + 4\cdot 2^j \le q + \frac{32}{\varepsilon}$ queries in this round. Thus, the fraction of these sums erased during the round is at most

$$t \cdot \frac{q+\frac{32}{\varepsilon}}{2^{q-1}-1} \le t\cdot\left(\frac{1}{2^{q/2}} + \frac{70}{\varepsilon\cdot 2^q}\right) \le \frac{\varepsilon^2}{C} + \frac{70\varepsilon^3}{4t\cdot C^2} \le \varepsilon^2\left(\frac{1}{C}+\frac{8.75}{C^2}\right) \le \frac{\varepsilon^2}{88\cdot 32},$$

where in the first inequality we used that $\frac{q}{2^{q-1}-1} \le \frac{1}{2^{q/2}}$ for $q \ge 9$ and that $(2^{q-1}-1)/32 \ge 2^q/70$ for $q \ge 5$ (note that $q \ge 2\log(3000t/\varepsilon^2) \ge 25$), in the second inequality we used $q \ge 2\log(3000t/\varepsilon^2)$, and in the third inequality we used $\varepsilon \le \frac{1}{2}$.

Since there are at most $4\cdot 2^j \le \frac{32}{\varepsilon}$ iterations of Step 8, we obtain by a union bound that

$$\Pr[B \mid G] \le \frac{32}{\varepsilon}\cdot\frac{\varepsilon^2}{88\cdot 32} = \frac{\varepsilon}{88}.$$

Consequently, $\Pr[B] \le \frac{\varepsilon}{88} + \frac{\varepsilon}{800} \le \frac{\varepsilon}{80}$. The number of rounds of Algorithm 1 is at most

$$\sum_{j=1}^{\log 8/\varepsilon} \frac{8\ln 5}{2^j\varepsilon} = \frac{8\ln 5}{\varepsilon}\sum_{j=1}^{\log\frac{8}{\varepsilon}}\frac{1}{2^j} \le \frac{8\ln 5}{\varepsilon}.$$

Therefore, Algorithm 1 queries an erased point during its execution with probability at most $\frac{8\ln 5}{\varepsilon}\cdot\Pr[B] \le \frac{1}{6}$. This concludes the proof. ∎

8.3 Online-corruption-resilient tester of sortedness with few distinct values

In this section, we show the existence of nonadaptive online-corruption-resilient tester for sortedness of sequences with few distinct values.

Corollary 8.2. *There exists a nonadaptive, t-online-corruption-resilient ε-tester for sortedness of n-element sequences with at most r distinct values. The tester makes $O(\frac{\sqrt{r}}{\varepsilon})$ uniform and independent queries and works when $r < \frac{\varepsilon^2 n}{c_0 t}$, where $c_0 > 0$ is a specific constant.*

Proof. Recall that the online-erasure-resilient sortedness tester makes $\frac{2c\sqrt{r}}{\varepsilon}$ uniform and independent queries to $f : [n] \to [r]$. The tester always accepts when f is sorted. We can easily modify the tester, without affecting its asymptotic query complexity, to reject with probability at least 5/6 (as opposed to 2/3) when f is ε-far from sorted. Then, by Lemma 1.8, it suffices to show that, for all adversarial strategies, this tester queries an erased point with probability at most 1/6. The adversary performs at most $\frac{2c\sqrt{r}t}{\varepsilon}$ erasures during the execution of the algorithm. Since the tester performs its queries uniformly at random, the probability that a given query is erased is at most $\frac{2c\sqrt{r}t}{\varepsilon n}$. By a union bound, the probability that some query of the tester is erased is at most $\frac{4c^2 rt}{\varepsilon^2 n}$. Set $c_0 = 24c^2$. By our assumption that $r < \frac{\varepsilon^2 n}{24c^2 t}$, we obtain that the probability that the tester samples an erased point is at most $\frac{1}{6}$. ◻

8.4 Finding witnesses in the presence of online corruptions

In this section, we prove Lemma 1.9. Specifically, we show how to modify a nonadaptive, 1-sided error, online-erasure-resilient tester for a property $\mathcal{P}$ to get an algorithm that accesses the input via an online-corruption oracle and outputs a witness demonstrating the violation of $\mathcal{P}$.

Proof of Lemma 1.9. Let T be a nonadaptive, 1-sided error, t-online-erasure-resilient tester for $\mathcal{P}$. Let f be ε-far from $\mathcal{P}$. Fix a strategy of the t-online-corruption oracle, and let $\mathcal{O}^{(c)}$ denote the oracle running this strategy. Define a a t-online-erasure oracle $\mathcal{O}^{(e)}$ that follows the same strategy as $\mathcal{O}^{(e)}$, except that $\mathcal{O}^{(e)}$ erases the value at each point x that $\mathcal{O}^{(c)}$ modifies, i.e., it sets $\mathcal{O}^{(e)}(x) = \perp$.

If the tester T accesses f via $\mathcal{O}^{(e)}$, it rejects with probability at least 2/3. Fix the random coins of an execution of T for which it rejects. Let $T^{(e)}$ be the execution of T with those coins (where access to f is via $\mathcal{O}^{(e)}$). Similarly, let $T^{(c)}$ be the execution of T with the same coins when access to f is via $\mathcal{O}^{(c)}$. Since T is nonadaptive, $T^{(c)}$ and $T^{(e)}$ make the same queries. By definition of $\mathcal{O}^{(e)}$, if $T^{(e)}$ obtains $O^{(e)}(x) = f(x)$ for a query x, then $T^{(c)}$ obtains $O^{(c)}(x) = f(x)$ for the same query.

Since $T^{(e)}$ is a 1-sided error tester, it can reject only if it successfully obtains values of f on some queried points $x_1, \dots, x_k$ such that no $g \in \mathcal{P}$ has $g(x_i) = f(x_i)$ for all $i \in [k]$. Then, for the same queries, $T^{(c)}$ obtains $\mathcal{O}^{(c)}(x_1) = f(x_1), \dots, \mathcal{O}^{(c)}(x_k) = f(x_k)$. Therefore, $T^{(c)}$ finds a witness of $f \notin \mathcal{P}$. We modify T to output a witness if it finds one. (Note that $T^{(c)}$ and $T^{(e)}$ might output different sets of points violating $\mathcal{P}$, since $T^{(c)}$ will obtain additional values $\mathcal{O}^{(c)}(x) \in \mathbb{R}$ for the queries x for which $T^{(e)}$ obtained $\mathcal{O}^{(e)}(x) = \perp$.) Now, if we run the modified T with access to f via $\mathcal{O}^{(c)}$, it outputs a witness of $f \notin \mathcal{P}$ with probability at least 2/3. ◻

9 Yao's minimax principle for online erasure oracles

We extend Yao's minimax principle [72] to the setting of online erasures, so that it can be used to prove lower bounds on the query complexity of randomized online-erasure-resilient testers. In the standard property testing model, Yao's principle allows one to prove a lower bound on query complexity by describing a distribution on inputs that is hard to test, in the sense that a deterministic algorithm needs many queries to decide, with high probability over the input, whether an input sampled from the distribution has the property or is far from it. In the setting of online erasures, we show that one can prove a lower bound by describing a distribution on inputs as well as an oracle erasure strategy such that every deterministic algorithm that accesses the input selected from the distribution via the erasure oracle needs many queries to test a specified property with high probability.

Theorem 9.1. *To prove a lower bound q on the worst-case query complexity of online-erasure-resilient randomized algorithms that perform a specified computational task with error probability strictly less than $\frac{1}{3}$, it is enough to give a distribution $\mathcal{D}$ on inputs of size n and a (randomized) oracle erasure strategy $\mathcal{S}$, such that every deterministic algorithm making at most q queries to an input drawn from $\mathcal{D}$ via an online-erasure oracle that uses strategy $\mathcal{S}$, errs with probability at least $\frac{1}{3}$.*

Proof. We prove a min-max inequality similar to Yao's original min-max inequality [72]. The inequality relates the error complexities of the algorithms, rather than their query complexity, which we will fix beforehand. Consider a randomized algorithm for the specified task that, given access to an input object of size n via a t-online-erasure oracle, makes at most q queries, and may be incorrect with some probability. We can represent the randomized algorithm as a probability distribution τ on the set of all (possibly incorrect) deterministic algorithms A with query complexity at most q. Let $\mathcal{A}_q$ denote the set of such algorithms, and let $\tau(A)$ denote the probability of drawing algorithm A from the set $\mathcal{A}_q$.

Denote by X_n the set of inputs of size n for the computational task. Let $\mathcal{S}_{q,n}$ be the set of deterministic erasure strategies of the t-online-erasure oracle for inputs of size n against algorithms of query complexity at most q. An adversarial strategy can be represented by a decision tree that, based on the input and the queries of the algorithm, indicates which input entries to erase next. Since we are only considering inputs of fixed size and algorithms of bounded query complexity, the set of oracle strategies has finite cardinality. Oracle strategies can be randomized, so we consider a distribution on deterministic strategies. For ease of notation, we consider a joint distribution on inputs and oracle strategies, i.e., we let σ be a distribution on the input-strategy pairs $X_n \times \mathcal{S}_{q,n}$. Given an input $x \in X_n$, an adversarial strategy $s \in \mathcal{S}_{q,n}$, and an algorithm $A \in \mathcal{A}_q$, let $I(A, x, s)$ be the indicator function for A being incorrect on x, i.e., $I(A, x, s) = 1$ if A is incorrect on x against an oracle that uses strategy s, and $I(A, x, s) = 0$ otherwise.

We prove the following min-max inequality

$$\min_{\tau} \max_{(x,s)\in X_n\times\mathcal{S}_{q,n}} \sum_{A\in\mathcal{A}_q} I(A,x,s)\tau(A) \geq \max_{\sigma} \min_{A\in\mathcal{A}_q} \sum_{(x,s)\in X_n\times\mathcal{S}_{q,n}} I(A,x,s)\sigma(x,s). \tag{9.1}$$

We comment on how (9.1) implies Theorem 9.1. Suppose there exists some distribution σ on input-strategy pairs, such that every q-query deterministic algorithm errs with probability at least $1/3$ on an input-strategy pair drawn from σ. This implies that the right-hand side of (9.1) is at least $\frac{1}{3}$. (Note that we can express σ as a distribution $\mathcal{D}$ on the inputs and a randomized adversarial strategy $\mathcal{S}$). By (9.1), every q-query randomized algorithm errs with probability at least $\frac{1}{3}$ for every input and adversarial strategy. Therefore, an algorithm for the given computational task that errs with probability strictly less than $\frac{1}{3}$ has to make at least q queries, which implies Theorem 9.1.

We now prove (9.1). Fix distribution τ on algorithms and distribution σ on input-strategy pairs. Then

$$\begin{aligned}
\max_{(x,s)\in X_n\times\mathcal{S}_{q,n}} \sum_{A\in\mathcal{A}_q} I(A,x,s)\tau(A) &= \left(\sum_{(x,s)\in X_n\times\mathcal{S}_{q,n}} \sigma(x,s)\right) \max_{(x,s)\in X_n\times\mathcal{S}_{q,n}} \sum_{A\in\mathcal{A}_q} I(A,x,s)\tau(A) \\
&\geq \sum_{(x,s)\in X_n\times\mathcal{S}_{q,n}} \sigma(x,s) \sum_{A\in\mathcal{A}_q} I(A,x,s)\tau(A) \\
&= \sum_{A\in\mathcal{A}_q} \tau(A) \sum_{(x,s)\in X_n\times\mathcal{S}_{q,n}} I(A,x,s)\sigma(x,s) \qquad (9.2) \\
&\geq \sum_{A\in\mathcal{A}_q} \tau(A) \min_{A\in\mathcal{A}_q} \sum_{(x,s)\in X_n\times\mathcal{S}_{q,n}} I(A,x,s)\sigma(x,s) \\
&= \min_{A\in\mathcal{A}_q} \sum_{(x,s)\in X_n\times\mathcal{S}_{q,n}} I(A,x,s)\sigma(x,s).
\end{aligned}$$

Crucially, the fact that $\mathcal{A}_q$, X_n, and $\mathcal{S}_{q,n}$ have finite cardinalities allows us to exchange the order of the sums in (9.2). ◻

9.1 A version of Yao's minimax with two distributions

In this section, we prove a corollary to Theorem 9.1 which refers to the more common usage of Yao's minimax principle for proving lower bounds, where separate distributions on positive and negative instances are defined. The corollary is a generalization of Claim 5 in [60].

Definition 9.2. The *statistical distance* between two discrete distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ is

$$CD(\mathcal{D}_1,\mathcal{D}_2) = \max_{S\subseteq \text{support}(\mathcal{D}_1)\cup\text{support}(\mathcal{D}_2)} \left(\left| \Pr_{x\sim\mathcal{D}_1}[x\in S] - \Pr_{x\sim\mathcal{D}_2}[x\in S] \right| \right).$$

Definition 9.3. Given a q-query deterministic algorithm $\mathcal{A}$, let $a(x)$ be the string of q answers that $\mathcal{A}$ receives when making queries to input object x. For a distribution $\mathcal{D}$ and adversarial strategy $\mathcal{S}$, let $\mathcal{D}$-view be the distribution on strings $a(x)$ where the input object x is sampled from $\mathcal{D}$ and accessed via a t-online-erasure oracle using strategy $\mathcal{S}$.

Corollary 9.4. *To prove a lower bound q on the worst-case query complexity of online-erasure-resilient randomized algorithms for a promise decision problem, it is enough to give*

- *a randomized adversarial strategy* $\mathcal{S}$*,*
- *a distribution* $\mathcal{D}^+$ *on positive instances of size* n*, and*
- *a distribution* $\mathcal{D}^-$ *on instances of size* n *that are negative with probability at least* $\frac{6}{7}$*,*

such that $CD(\mathcal{D}^+\text{-view}, \mathcal{D}^-\text{-view}) < \frac{1}{6}$ *for every deterministic* q*-query algorithm that accesses its input via an online-erasure oracle using strategy* $\mathcal{S}$*.*

Our proof of Corollary 9.4 is similar to an argument in [60]. We rely on the following claim from [60].

Claim 9.5 (Claim 4 of [60])**.** *Let* E *be an event that happens with probability at least* $1-\delta$ *under the distribution* $\mathcal{D}$ *and let* $\mathcal{B}$ *be the distribution* $\mathcal{D}$ *conditioned on* E*. Then* $CD(\mathcal{D}, \mathcal{B}) \leq \delta'$*, where* $\delta' = \frac{1}{1-\delta} - 1$*.*

Proof of Corollary 9.4. Let A be a deterministic q-query algorithm for the promise decision problem. Given distributions $\mathcal{D}^+$ and $\mathcal{D}^-$, we define a distribution $\mathcal{D}$ satisfying the conditions of Theorem 9.1. Let E be the event that $x \sim \mathcal{D}^-$ is a negative instance, and let $\mathcal{B}$ be the distribution $\mathcal{D}^-$ conditioned on the event E. To obtain a sample from $\mathcal{D}$, with probability $1/2$ draw a sample from $\mathcal{D}^+$, and with probability $1/2$ draw a sample from $\mathcal{B}$. We show that A errs with probability at least $1/3$ when it accesses an input object $x \sim \mathcal{D}$ via the online-erasure oracle that uses strategy $\mathcal{S}$. Let Q be the set of query-answers strings a on which A accepts. By the law of total probability,

$$\begin{aligned}
\Pr_{x\sim\mathcal{D}}[A(x) \text{ is correct}] &= \frac{1}{2}\Pr_{x\sim\mathcal{D}^+}[A(x)\text{ accepts}] + \frac{1}{2}\Pr_{x\sim\mathcal{B}}[A(x)\text{ rejects}] \\
&= \frac{1}{2} + \frac{1}{2}\left(\Pr_{x\sim\mathcal{D}^+}[A(x)\text{ accepts}] - \Pr_{x\sim\mathcal{B}}[A(x)\text{ accepts}]\right) \\
&= \frac{1}{2} + \frac{1}{2}\left(\Pr_{a\sim\mathcal{D}^+\text{-view}}[a \in Q] - \Pr_{a\sim\mathcal{B}\text{-view}}[a \in Q]\right) \\
&\leq \frac{1}{2} + \frac{1}{2}CD(\mathcal{D}^+\text{-view}, \mathcal{B}\text{-view}).
\end{aligned}$$

Note that $\mathcal{B}$-view is the same as the distribution $\mathcal{D}^-$-view conditioned on the event E. By Claim 9.5 and the assumption that $\Pr_{x\sim\mathcal{D}^-}[E] \geq \frac{6}{7}$, we have $CD(\mathcal{B}\text{-view}, \mathcal{D}^-\text{-view}) \leq \frac{1}{6}$. By the triangle inequality,

$$CD(\mathcal{D}^+\text{-view}, \mathcal{B}\text{-view}) \leq CD(\mathcal{D}^+\text{-view}, \mathcal{D}^-\text{-view}) + CD(\mathcal{B}\text{-view}, \mathcal{D}^-\text{-view},) < \frac{1}{6} + \frac{1}{6} = \frac{1}{3}.$$

Therefore, $\Pr_{x\sim\mathcal{D}}[A(x) \text{ is correct}] < \frac{1}{2} + \frac{1}{2}\cdot\frac{1}{3} = \frac{2}{3}$. ☐

References

[1] NIR AILON, BERNARD CHAZELLE, SESHADHRI COMANDUR, AND DING LIU: Estimating the distance to a monotone function. *Random Struct. Algor.*, 31(3):371–383, 2007. [doi:10.1002/rsa.20167] 4

[2] NOGA ALON, TALI KAUFMAN, MICHAEL KRIVELEVICH, SIMON LITSYN, AND DANA RON: Testing Reed-Muller codes. *IEEE Trans. Inform. Theory*, 51(11):4032–4039, 2005. [doi:10.1109/TIT.2005.856958] 3, 6, 7, 11, 24, 27

[3] 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] 6

[4] 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] 7

[5] LÁSZLÓ BABAI, LANCE FORTNOW, LEONID A. LEVIN, AND MARIO SZEGEDY: Checking computations in polylogarithmic time. In *Proc. 23rd STOC*, pp. 21–31. ACM Press, 1991. [doi:10.1145/103418.103428] 6

[6] LÁSZLÓ BABAI, LANCE FORTNOW, AND CARSTEN LUND: Non-deterministic exponential time has two-prover interactive protocols. *Comput. Complexity*, 1(1):3–40, 1991. [doi:10.1007/BF01200056] 6

[7] AMOS BEIMEL, HAIM KAPLAN, YISHAY MANSOUR, KOBBI NISSIM, THATCHAPHOL SARANURAK, AND URI STEMMER: Dynamic algorithms against an adaptive adversary: generic constructions and lower bounds. In *Proc. 54th STOC*, pp. 1671–1684. ACM Press, 2022. [doi:10.1145/3519935.3520064] 4

[8] MIHIR BELLARE, DON COPPERSMITH, JOHAN HÅSTAD, MARCOS A. KIWI, AND MADHU SUDAN: Linearity testing in characteristic two. *IEEE Trans. Inform. Theory*, 42(6):1781–1795, 1996. [doi:10.1109/18.556674] 3, 5, 12, 19, 20

[9] MIHIR BELLARE, ODED GOLDREICH, AND MADHU SUDAN: Free bits, PCPs, and nonapproximability–towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998. [doi:10.1137/S0097539796302531] 5

[10] MIHIR BELLARE, SHAFI GOLDWASSER, CARSTEN LUND, AND ALEXANDER RUSSELL: Efficient probabilistically checkable proofs and applications to approximations. In *Proc. 25th STOC*, pp. 294–304. ACM Press, 1993. [doi:10.1145/167088.167174] 5

[11] MIHIR BELLARE AND MADHU SUDAN: Improved non-approximability results. In *Proc. 26th STOC*, pp. 184–193. ACM Press, 1994. [doi:10.1145/195058.195129] 5

[12] ALEKSANDRS BELOVS: Adaptive lower bound for testing monotonicity on the line. In *Proc. 22nd Internat. Conf. on Randomization and Computation (RANDOM'18)*, pp. 31:1–10. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. [doi:10.4230/LIPIcs.APPROX-RANDOM.2018.31] 4, 6, 7

[13] OMRI BEN-ELIEZER, ELDAR FISCHER, AMIT LEVI, AND RON D. ROTHBLUM: Hard properties with (very) short PCPPs and their applications. In *Proc. 11th Innovations in Theoret. Comp. Sci. Conf. (ITCS'20)*, pp. 9:1–27. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2020. [doi:10.4230/LIPIcs.ITCS.2020.9] 4

[14] OMRI BEN-ELIEZER, RAJESH JAYARAM, DAVID P. WOODRUFF, AND EYLON YOGEV: A framework for adversarially robust streaming algorithms. *J. ACM*, 69(2):17:1–33, 2022. [doi:10.1145/3498334] 4

[15] OMRI BEN-ELIEZER AND EYLON YOGEV: The adversarial robustness of sampling. In *Proc. 39th Symp. on Principles of Database Systems (PODS'20)*, pp. 49–62. ACM Press, 2020. [doi:10.1145/3375395.3387643] 4

[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 *Proc. 35th STOC*, pp. 612–621. ACM Press, 2003. [doi:10.1145/780542.780631] 5

[17] PIOTR BERMAN, SOFYA RASKHODNIKOVA, AND GRIGORY YAROSLAVTSEV: L_p-testing. In *Proc. 46th STOC*, pp. 164–173. ACM Press, 2014. [doi:10.1145/2591796.2591887] 6, 7, 14, 15

[18] ARNAB BHATTACHARYYA, ELENA GRIGORESCU, KYOMIN JUNG, SOFYA RASKHODNIKOVA, AND DAVID P. WOODRUFF: Transitive-closure spanners. *SIAM J. Comput.*, 41(6):1380–1425, 2012. [doi:10.1137/110826655] 4

[19] ARNAB BHATTACHARYYA, SWASTIK KOPPARTY, GRANT SCHOENEBECK, MADHU SUDAN, AND DAVID ZUCKERMAN: Optimal testing of Reed-Muller codes. In *Proc. 51st FOCS*, pp. 488–497. IEEE Comp. Soc., 2010. A version of this paper, by the same title and authors, appeared as a chapter in "Property Testing: Current Research and Surveys" (Oded Goldreich, ed.), 2010, pp. 269–275, Springer LNCS vol. 6390. [doi:10.1109/FOCS.2010.54] 6, 11

[20] ERIC BLAIS, SOFYA RASKHODNIKOVA, AND GRIGORY YAROSLAVTSEV: Lower bounds for testing properties of functions over hypergrid domains. In *Proc. 29th IEEE Conf. on Comput. Complexity (CCC'14)*, pp. 309–320. IEEE Comp. Soc., 2014. [doi:10.1109/CCC.2014.38] 7

[21] MANUEL BLUM, MICHAEL LUBY, AND RONITT RUBINFELD: Self-testing/correcting with applications to numerical problems. *J. Comput. System Sci.*, 47(3):549–595, 1993. [doi:10.1016/0022-0000(93)90044-W] 3, 5

[22] DEEPARNAB CHAKRABARTY, KASHYAP DIXIT, MADHAV JHA, AND C. SESHADHRI: Property testing on product distributions: Optimal testers for bounded derivative properties. *ACM Trans. Algorithms*, 13(2):20:1–30, 2017. [doi:10.1145/3039241] 7

[23] DEEPARNAB CHAKRABARTY AND C. SESHADHRI: Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proc. 45th STOC*, pp. 419–428. ACM Press, 2013. [doi:10.1145/2488608.2488661] 4, 7

[24] DEEPARNAB CHAKRABARTY AND C. SESHADHRI: An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10(17):453–464, 2014. [doi:10.4086/toc.2014.v010a017] 6

[25] IRIT DINUR AND VENKATESAN GURUSWAMI: PCPs via low-degree long code and hardness for constrained hypergraph coloring. In *Proc. 54th FOCS*, pp. 340–349. IEEE Comp. Soc., 2013. [doi:10.1109/FOCS.2013.44] 6

[26] KASHYAP DIXIT, MADHAV JHA, SOFYA RASKHODNIKOVA, AND ABHRADEEP THAKURTA: Testing the Lipschitz property over product distributions with applications to data privacy. In *Proc. Theory of Cryptography Conf. (TCC'13)*, pp. 418–436. Springer, 2013. [doi:10.1007/978-3-642-36594-2_24] 7

[27] KASHYAP DIXIT, SOFYA RASKHODNIKOVA, ABHRADEEP THAKURTA, AND NITHIN VARMA: Erasure-resilient property testing. *SIAM J. Comput.*, 47(2):295–329, 2018. [doi:10.1137/16M1075661] 4, 11

[28] YEVGENIY DODIS, ODED GOLDREICH, ERIC LEHMAN, SOFYA RASKHODNIKOVA, DANA RON, AND ALEX SAMORODNITSKY: Improved testing algorithms for monotonicity. In *Proc. 3rd Internat. Workshop on Randomization and Computation (RANDOM'99)*, pp. 97–108. Springer, 1999. [doi:10.1007/978-3-540-48413-4_10] 4, 32

[29] FUNDA ERGÜN, SAMPATH KANNAN, RAVI KUMAR, RONITT RUBINFELD, AND MAHESH VISWANATHAN: Spot-checkers. *J. Comput. System Sci.*, 60(3):717–751, 2000. [doi:10.1006/jcss.1999.1692] 4, 6

[30] URIEL FEIGE, SHAFI GOLDWASSER, LÁSZLÓ LOVÁSZ, SHMUEL SAFRA, AND MARIO SZEGEDY: Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, 1996. [doi:10.1145/226643.226652] 5, 6

[31] ELDAR FISCHER: On the strength of comparisons in property testing. *Inform. Comput.*, 189(1):107–116, 2004. [doi:10.1016/j.ic.2003.09.003] 6

[32] ELDAR FISCHER, ERIC LEHMAN, ILAN NEWMAN, SOFYA RASKHODNIKOVA, RONITT RUBINFELD, AND ALEX SAMORODNITSKY: Monotonicity testing over general poset domains. In *Proc. 34th STOC*, pp. 474–483. ACM Press, 2002. [doi:10.1145/509907.509977] 4, 7

[33] KATALIN FRIEDL AND MADHU SUDAN: Some improvements to total degree tests. In *Proc. 3rd Isr. Symp. Theory Comp. Sys. (ISTCS'95)*, pp. 190–198. IEEE Comp. Soc., 1995. [doi:10.1109/ISTCS.1995.377032] 6

[34] PETER GEMMELL, RICHARD J. LIPTON, RONITT RUBINFELD, MADHU SUDAN, AND AVI WIGDERSON: Self-testing/correcting for polynomials and for approximate functions. In *Proc. 23rd STOC*, pp. 33–42. ACM Press, 1991. [doi:10.1145/103418.103429] 6

[35] ODED GOLDREICH: On multiple input problems in property testing. In *Proc. 18th Internat. Workshop on Randomization and Computation (RANDOM'14)*, pp. 704–720. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014. [doi:10.4230/LIPIcs.APPROX-RANDOM.2014.704] 6

[36] ODED GOLDREICH, SHAFI GOLDWASSER, ERIC LEHMAN, DANA RON, AND ALEX SAMORODNITSKY: Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000. [doi:10.1007/s004930070011] 33

[37] ODED GOLDREICH, SHAFI GOLDWASSER, AND DANA RON: Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. [doi:10.1145/285055.285060] 3

[38] VENKATESAN GURUSWAMI AND ATRI RUDRA: Tolerant locally testable codes. In *Proc. 9th Internat. Workshop on Randomization and Computation (RANDOM'05)*, pp. 306–317. Springer, 2005. [doi:10.1007/11538462_26] 4

[39] ELAD HARAMATY, AMIR SHPILKA, AND MADHU SUDAN: Optimal testing of multivariate polynomials over small prime fields. *SIAM J. Comput.*, 42(2):536–562, 2013. [doi:10.1137/120879257] 6

[40] JOHAN HÅSTAD AND AVI WIGDERSON: Simple analysis of graph tests for linearity and PCP. *Random Struct. Algor.*, 22(2):139–160, 2003. [doi:10.1002/rsa.10068] 5

[41] MADHAV JHA AND SOFYA RASKHODNIKOVA: Testing and reconstruction of Lipschitz functions with applications to data privacy. *SIAM J. Comput.*, 42(2):700–731, 2013. [doi:10.1137/110840741] 7

[42] CHARANJIT S. JUTLA, ANINDYA C. PATTHAK, ATRI RUDRA, AND DAVID ZUCKERMAN: Testing low-degree polynomials over prime fields. *Random Struct. Algor.*, 35(2):163–193, 2009. [doi:10.1002/rsa.20262] 6

[43] IDEN KALEMAJ, SOFYA RASKHODNIKOVA, AND NITHIN VARMA: Sublinear-time computation in the presence of online erasures. In *Proc. 13th Innovations in Theoret. Comp. Sci. Conf. (ITCS'22)*, volume 215, pp. 90:1–25. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2022. [doi:10.4230/LIPIcs.ITCS.2022.90] 1

[44] TALI KAUFMAN, SIMON LITSYN, AND NING XIE: Breaking the ϵ-soundness bound of the linearity test over GF(2). *SIAM J. Comput.*, 39(5):1988–2003, 2010. [doi:10.1137/080715548] 5

[45] TALI KAUFMAN AND DANA RON: Testing polynomials over general fields. *SIAM J. Comput.*, 36(3):779–802, 2006. [doi:10.1137/S0097539704445615] 6

[46] SWASTIK KOPPARTY AND SHUBHANGI SARAF: Tolerant linearity testing and locally testable codes. In *Proc. 13th Internat. Workshop on Randomization and Computation (RANDOM'09)*, pp. 601–614. Springer, 2009. [doi:10.1007/978-3-642-03685-9_45] 4

[47] AMIT LEVI, RAMESH KRISHNAN S. PALLAVOOR, SOFYA RASKHODNIKOVA, AND NITHIN VARMA: Erasure-resilient sublinear-time graph algorithms. *ACM Trans. Comput. Theory*, 14(1):1–22, 2021. Preliminary version in ITCS'21. [doi:10.1145/3488250] 4

[48] ALESSANDRO MANTELERO: The EU proposal for a general data protection regulation and the roots of the 'right to be forgotten'. *Computer Law & Security Review*, 29(3):229–235, 2013. [doi:10.1016/j.clsr.2013.03.010] 2

[49] DANA MOSHKOVITZ: Low-degree test with polynomially small error. *Comput. Complexity*, 26(3):531–582, 2017. [doi:10.1007/s00037-016-0149-4] 6

[50] DANA MOSHKOVITZ AND RAN RAZ: Sub-constant error low degree test of almost-linear size. *SIAM J. Comput.*, 38(1):140–180, 2008. [doi:10.1137/060656838] 6

[51] ILAN NEWMAN AND NITHIN VARMA: New sublinear algorithms and lower bounds for LIS estimation. In *Proc. 48th Internat. Colloq. on Automata, Languages, and Programming (ICALP'21)*, pp. 100:1–20. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2021. [doi:10.4230/LIPIcs.ICALP.2021.100] 4

[52] RYAN O'DONNELL: *Analysis of Boolean Functions*. Cambridge Univ. Press, 2014. [doi:10.1017/CBO9781139814782, arXiv:2105.10386] 12

[53] RAMESH KRISHNAN S. PALLAVOOR, SOFYA RASKHODNIKOVA, AND NITHIN VARMA: Parameterized property testing of functions. *ACM Trans. Comput. Theory*, 9(4):17:1–19, 2018. [doi:10.1145/3155296] 7, 32

[54] RAMESH KRISHNAN S. PALLAVOOR, SOFYA RASKHODNIKOVA, AND ERIK WAINGARTEN: Approximating the distance to monotonicity of boolean functions. *Random Struct. Algor.*, 60(2):233–260, 2022. [doi:10.1002/rsa.21029] 4

[55] RAMESH KRISHNAN PALLAVOOR SURESH: *Improved Algorithms and New Models in Property Testing*. Ph. D. thesis, Boston University, 2020. OpenBU. 7

[56] MICHAL PARNAS, DANA RON, AND RONITT RUBINFELD: Tolerant property testing and distance approximation. *J. Comput. System Sci.*, 72(6):1012–1042, 2006. [doi:10.1016/j.jcss.2006.03.002] 4

[57] SOFYA RASKHODNIKOVA: Testing if an array is sorted. In MING-YANG KAO, editor, *Encyclopedia of Algorithms*, pp. 2219–2222. Springer, 2016. [doi:10.1007/978-1-4939-2864-4_700] 6

[58] SOFYA RASKHODNIKOVA, NOGA RON-ZEWI, AND NITHIN VARMA: Erasures versus errors in local decoding and property testing. *Random Struct. Algor.*, 59(4):640–670, 2021. [doi:10.1002/rsa.21031] 4

[59] SOFYA RASKHODNIKOVA AND RONITT RUBINFELD: Linearity testing/Testing Hadamard codes. In MING-YANG KAO, editor, *Encyclopedia of Algorithms*, pp. 1107–1110. Springer, 2016. [doi:10.1007/978-1-4939-2864-4_202] 5

[60] SOFYA RASKHODNIKOVA AND ADAM D. SMITH: A note on adaptivity in testing properties of bounded degree graphs. *Electron. Colloq. Comput. Complexity*, TR06-089, 2006. [ECCC] 39, 40

[61] SOFYA RASKHODNIKOVA AND NITHIN VARMA: Brief announcement: Erasure-resilience versus tolerance to errors. In *Proc. 45th Internat. Colloq. on Automata, Languages, and Programming (ICALP'18)*, pp. 111:1–3. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. [doi:10.4230/LIPIcs.ICALP.2018.111] 4

[62] RAN RAZ AND SHMUEL SAFRA: A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proc. 29th STOC*, pp. 475–484. ACM Press, 1997. [doi:10.1145/258533.258641] 6

[63] NOGA RON-ZEWI AND MADHU SUDAN: A new upper bound on the query complexity of testing generalized Reed-Muller codes. *Theory of Computing*, 9(25):783–807, 2013. [doi:10.4086/toc.2013.v009a025] 6

[64] RONITT RUBINFELD AND MADHU SUDAN: Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996. [doi:10.1137/S0097539793255151] 3, 6

[65] MICHAEL E. SAKS AND C. SESHADHRI: Estimating the longest increasing sequence in polylogarithmic time. *SIAM J. Comput.*, 46(2):774–823, 2017. [doi:10.1137/130942152] 4

[66] ALEX SAMORODNITSKY: Low-degree tests at large distances. In *Proc. 39th STOC*, pp. 506–515. ACM Press, 2007. [doi:10.1145/1250790.1250864] 5, 6

[67] ALEX SAMORODNITSKY AND LUCA TREVISAN: A PCP characterization of NP with optimal amortized query complexity. In *Proc. 32nd STOC*, pp. 191–199. ACM Press, 2000. [doi:10.1145/335305.335329] 5

[68] ALEX SAMORODNITSKY AND LUCA TREVISAN: Gowers uniformity, influence of variables, and PCPs. *SIAM J. Comput.*, 39(1):323–360, 2009. [doi:10.1137/070681612] 5, 6

[69] AMIR SHPILKA AND AVI WIGDERSON: Derandomizing homomorphism testing in general groups. *SIAM J. Comput.*, 36(4):1215–1230, 2006. [doi:10.1137/S009753970444658X] 5

[70] MADHU SUDAN AND LUCA TREVISAN: Probabilistically checkable proofs with low amortized query complexity. In *Proc. 39th FOCS*, pp. 18–27. IEEE Comp. Soc., 1998. [doi:10.1109/SFCS.1998.743425] 5

[71] LUCA TREVISAN: Recycling queries in PCPs and in linearity tests (extended abstract). In *Proc. 30th STOC*, pp. 299–308. ACM Press, 1998. [doi:10.1145/276698.276769] 5

[72] ANDREW CHI-CHIH YAO: Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *Proc. 18th FOCS*, pp. 222–227. IEEE Comp. Soc., 1977. [doi:10.1109/SFCS.1977.24] 38

AUTHORS

Iden Kalemaj
Graduate student
Department of Computer Science
Boston University
Boston, MA, USA
ikalemaj@bu.edu
https://cs-people.bu.edu/ikalemaj/

Sofya Raskhodnikova
Professor
Department of Computer Science
Boston University
Boston, MA, USA
sofya@bu.edu
https://cs-people.bu.edu/sofya/

Nithin Varma
Assistant Professor
Chennai Mathematical Institute
Chennai, Tamil Nadu, India
nithvarma@gmail.com
https://www.cmi.ac.in/~nithinvarma/

ABOUT THE AUTHORS

IDEN KALEMAJ is a Ph. D. student in Computer Science at Boston University, working under the wonderful supervision of Sofya Raskhodnikova, in the fields of Sublinear Algorithms and Differential Privacy.

Iden grew up in Vlorë, Albania. Her math teacher and school master, Dafina Brokaj, with a rare sense of humour and wisdom, instilled in Iden the confidence to pursue mathematics.

Iden completed her undergraduate studies in Mathematics at Princeton University, and started to explore her interest in theoretical computer science via classes in graph theory and complexity theory. In her free time she enjoys climbing in the many Boston gyms and taking dance classes.

SOFYA RASKHODNIKOVA is a professor of Computer Science at Boston University. Her office is located in the newly constructed Center for Computing and Data Sciences, colloquially known as the Jenga building. She received all her degrees (S. B., S. M., and Ph. D.) from MIT, and then was a postdoctoral fellow at the Hebrew University of Jerusalem and the Weizmann Institute of Science. She was a Professor of Computer Science and Engineering at Penn State and held visiting positions at the Institute for Pure and Applied Mathematics at UCLA, Boston University, Harvard University, and at the Simons Institute for the Theory of Computing at Berkeley. Sofya works in the areas of randomized and approximation algorithms. Her main interest is the design and analysis of sublinear-time algorithms for combinatorial problems. She has also made important contributions to data privacy. Sofya has been a faculty member at Sigma Camp, a residential summer science camp for kids, for several years. As far as her hobbies go, recall that she works on privacy.

NITHIN VARMA is a researcher working in the fields of Sublinear Algorithms and Approximation Algorithms. He got his Ph. D. from the Computer Science Department at Boston University, where he was fortunate to be advised by Sofya Raskhodnikova. After that, he was a postdoctoral researcher at the Department of Computer Science, University of Haifa, where he was hosted and mentored by Ilan Newman and Noga Ron-Zewi. He was introduced to Theoretical Computer Science during his undergraduate years at the Department of Computer Science and Engineering at the National Institute of Technology Calicut through the wonderful courses offered by Prof. K Muralikrishnan. In his free time, Nithin enjoys meditation, singing Indian classical music and practicing the Mohiniyattam dance.