On Optimal Testing of Linearity*

Vipul Arora[†1], Esty Kelman[‡2], and Uri Meir[3]

[1]School of Computing, National University of Singapore. vipul@comp.nus.edu.sg
[2]Boston University, Boston, MA, USA, and Massachusetts Institute of Technology, Cambridge, MA, USA. ekelman@mit.edu
[3]Blavatnik School of Computer Science, Tel-Aviv University, Tel-Aviv, Israel. urimeir.cs@gmail.com

Abstract

Linearity testing has been a focal problem in property testing of functions. We combine different known techniques and observations about Linearity testing in order to resolve two recent versions of this task.

First, we focus on the online-manipulation-resilient model introduced by Kalemaj, Raskhodnikova and Varma (Theory of Computing 2023). In this model, up to t data entries are adversarially manipulated after each query is answered. Ben-Eliezer, Kelman, Meir, and Raskhodnikova (ITCS 2024) showed an asymptotically optimal Linearity tester that is resilient to t manipulations per query, but fails if t is too large. We simplify their analysis for the regime of small t, and for larger values of t we instead use sample-based testers, as defined by Goldreich and Ron (ACM Transactions on Computation Theory 2016). A key observation is that sample-based testing is resilient to online manipulations but still achieves optimal query complexity for Linearity when t is large. We complement our result by showing that when t is *very* large any reasonable property, and in particular Linearity, cannot be tested at all.

Second, we consider Linearity over the reals with proximity parameter ε. Fleming and Yoshida (ITCS 2020) gave a tester using $O(1/\varepsilon \cdot log(1/\varepsilon))$ queries. We simplify their algorithms and modify the analysis accordingly, showing an optimal tester that only uses $O(1/\varepsilon)$ queries. This modification works for the low-degree testers presented in Arora, Bhattacharyya, Fleming, Kelman, and Yoshida (SODA 2023) as well, resulting in optimal testers for degree-d polynomials, for any constant d.

1 Introduction

In the field of *property testing* [RS96, GGR98], a randomized algorithm is given oracle access to a large object, and a promise that the object either has some property (YES instances), or is "far" from having it (NO instances), under some notion of distance. The goal of the algorithm is to distinguish between the two cases with success probability at least 2/3, where the complexity is measured by the number of oracle calls made (this is well justified, as the running time is typically polynomial in the query complexity).

Linearity testing. In this manuscript, we focus on testing functions over a field $\mathbb{F}$ (in particular $\mathbb{F} = \mathbb{F}_2$ and $\mathbb{F} = \mathbb{R}$). That is, the tester is allowed oracle access to some function $f : \mathbb{F}^n \to \mathbb{F}$ and distinguishes functions with a desired property from ones that are far from *any* function satisfying the property. The most studied property of functions is Linearity, for which a tester was first given in [BLR93], perhaps the earliest instance of a property tester in the literature. The test, dubbed "the 3 points test" simply draws two random points $x, y \in \mathbb{F}^n$ and accepts only if $f(x) + f(y) = f(x \oplus y)$, where $+$ denotes addition over $\mathbb{F}$, and $\oplus$ denotes point-wise addition. By definition, if f is linear, this equality must hold for any pair x, y. It was shown in various settings that if f is ε-far from linear, there is an *inequality* for at least an $\Omega(\varepsilon)$-fraction of the pairs x, y. [BLR93, BCH+96, FY20]. Thus, repeating the process $\theta(1/\varepsilon)$ times defines a tester for linearity with the same query complexity. We note this tester satisfies the additional requirement of *never* rejecting a linear function, making it a 1-sided error tester for linearity.

It is well known that $\Omega(1/\varepsilon)$ oracle calls are necessary to test linearity (and most reasonable properties [Fis24]), proving the 3 points test of [BLR93] is optimal in terms of query complexity.

*This work was done while the authors were visiting the Simons Institute for the Theory of Computing.

†Supported in part by NRF-AI Fellowship R-252-100-B13-281.

‡Supported in part by the National Science Foundation under Grant No. 2022446 and in part by NSF TRIPODS program (award DMS-2022448).

1.1 Online erasures over the Boolean field Linearity was studied extensively over the Boolean field $\mathbb{F}_2$ [BLR93,BS94,FGL+96,BCH+96,BGS98,Tre98,ST98,ST00,HW03,BSVW03,Sam07,ST09,SW06,KLX10,KRV23] and the survey in [RR16]. Before we present the erasure model, we first review the ideas from previous works that are used in our result.

k-point test. Recently, in the context of online erasures over $\mathbb{F} = \mathbb{F}_2$ (which we describe soon), the k-points test was proposed by [KRV23]. In this test, which generalizes the 3-point test, the algorithm chooses points $x_1, \ldots, x_{k-1}$ at random, and accepts only if $\sum_{i\in[k-1]} f(x_i) = f\left(\oplus_{i\in[k-1]} x_i\right)$. It was later shown by [BKMR24] that repeating the k-points test $\Theta(1/(k\varepsilon))$ times produces an optimal Linearity tester for any $k = O(1/\varepsilon)$.

sample-based testers. Linearity over $\mathbb{F}_2$ was also considered in [GR16] using a weaker oracle access, where the tester cannot retrieve $f(x)$ for a value x of its choice, but instead each oracle call outputs a pair $(x, f(x))$ with a uniformly random choice of $x \in_R \mathbb{F}_2^n$. A tester using such oracle access is called a *sample-based* tester [GGR98]. It was shown in [GR16] that $\Theta\left(n + 1/\varepsilon\right)$ samples are sufficient and necessary to test Linearity. The gist is that a linear function is determined by its value on a basis B of the entire space $\mathbb{F}_2^n$. After $\Theta(n)$ samples, one can extrapolate *some* linear function g, and use the remaining $1/\varepsilon$ samples to compare f and g. A formal argument for this setting, without online erasures, appears in Section 2.2.

The *online erasure model* was recently introduced by [KRV23]. In this model, after each query is answered, t data points go through manipulations. These can be either (1) erasures, where an entry $f(x)$ is replaced with $\perp$; or (2) corruptions, where an entry $f(x)$ can be changed to any value in the range of f.

In particular, they study Linearity testing over $\mathbb{F}_2$. First, they note the 3-points-test is fragile in the presence of an adversary: a tester that insists on querying a triplet of the form $(x, y, x + y)$ must make $\Omega\left(t\right)$ queries, as fewer queries allow the adversary to erase all (new) pairs that were created at each step. To overcome this, they devise the k-points test, which creates more combinations, and eventually leads to a tester with query complexity $O(\log t/\varepsilon)$. It was then pointed out in [BKMR24] that the k-point test not only preserves the probability of spotting a violation, but actually increases it by a factor of $\approx k$ — this is enough to reach query complexity of $\log t + 1/\varepsilon$, which is optimal. Indeed, a lower bound of $\Omega(\log t)$ was shown for this model by [KRV23], and an $\Omega(1/\varepsilon)$ holds even without erasures. Another recent work by Minzer and Zheng [MZ24] showed a striking $O(log^{\Theta(d)}(t))$ tester for any degree-d and any finite field. For the case of Linearity ($d = 1$), however, their result is suboptimal , achieving a query complexity of $O\left(\log^6(t/\varepsilon)/\varepsilon\right)$.

How many erasures can the tester handle? All the above testers share a similar limitation: they all require $t \leq 2^{n/c}$ for some $c \geq 2$ (See Table 1 for the precise values). It is natural to ask what is the highest value of t for which Linearity is still testable. Clearly, $t = 2^n$ is too much, but how close can we get?

In Section 2 we fully resolve this question. Our observation, roughly speaking, is that all previous testers include some queries that are "too predictable", making them susceptible to adversarial manipulations. However, this only occurs when the adversary has very large manipulation budget $t = 2^{\Omega(n)}$, in which case we can simply apply the sample-based tester instead, killing two birds at once. First, sample-based testers almost automatically overcome online manipulations. Second, the seemingly high sample complexity of the tester is actually optimal as in this regime we have $\Theta(\log(t) + 1/\varepsilon) = \Theta(n + 1/\varepsilon)$. Formally, we have the following.

THEOREM 1.1. (DOUBLY-EFFICIENT TESTER) *There exists a constant $c > 0$ such that for all $n \in \mathbb{N}, \varepsilon \in (0, 1/2]$, and t satisfying $t \leq c \min\left\{\varepsilon^2, 1/n^2\right\} \cdot 2^n$, there exists an ε-tester for linearity of functions $f : \{0,1\}^n \to \{0,1\}$ that works in the presence of a t-online erasure (or corruption) adversary and makes $O\left(\max\left\{1/\varepsilon, \log t\right\}\right)$ queries. In the case of erasure adversary, the tester has 1-sided error.*

Our tester can also overcome the stronger *budget-managing* adversary, which can "bank" their manipulation privilege and use it only when necessary. See [BKMR24] for more details.

Finally, we show our tester is optimal in both term of query complexity and the amount of manipulations it can handle, which makes it *doubly-optimal*. For even a slightly larger value of t, we show that Linearity is no longer testable in the model. For any n, t, ε where Linearity is testable, the tester has optimal query complexity.

THEOREM 1.2. *ε-testing linearity in the online erasure model is impossible whenever for any $\varepsilon \in (0, 1/2)$ provided that $t \geq 20\varepsilon^2 2^n$.*

1.2 Optimal Linearity tester over the Reals Over the reals, there is a distinction between

Result	Query Complexity	Range of Erasures	Reference
Lower bound	$\Omega(\log(t)+1/\varepsilon)$	any t	[KRV23]
Lower bound	untestable	$t \geq \tilde{\Omega}(2^n)$	Theorem 1.2
Algorithm	$O(\log(t)/\varepsilon)$	$t \leq O(2^{n/4})$	[KRV23]
Algorithm	$O(\log^6(t/\varepsilon)/\varepsilon)$	$t \leq O(2^{n/20})$	[MZ24]
Algorithm	$O(\log(t)+1/\varepsilon)$	$t \leq O(2^{n/2})$	[BKMR24]
New tester	$O(\log(t)+1/\varepsilon)$	$t \leq \tilde{O}(2^n)$	Theorem 1.1

Table 1: A comparison of all known testers for linearity in the online erasure model. For the sake of clarity, in the column for the range of erasures, we hide factors of $\mathsf{poly}(n/\varepsilon)$.

- *linearity*: $f(\boldsymbol{x}) \equiv \sum_{i=1}^{n} c_i x_i$, for some $\{c_i \in \mathbb{R}, i \in \{1,\ldots,n\}\}$, and
- *additivity*: for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, $f(\boldsymbol{x}) + f(\boldsymbol{y}) = f(\boldsymbol{x}+\boldsymbol{y})$.

In fact, the 3 point BLR test checks for additivity, and this alone does not guarantee linearity due to some pathological examples that are additive, but not linear [Ham05]. We therefore assume the function f is continuous, as was done in previous works, and focus on testing additivity instead.

The best known result [FY20, Theorem 1, Algorithms 3, 2] for distribution-free additivity tester uses $O(\frac{1}{\varepsilon}\log(\frac{1}{\varepsilon}))$ queries. Their algorithm uses the 'self-correct and test' approach. First, they test if f violates additivity over a sufficiently large ($\Theta(\frac{1}{\varepsilon})$) random set of points. They then define the self-correction function $g : \mathbb{R}^n \to \mathbb{R}$ of f as follows:

$$g(\boldsymbol{p}) \triangleq \kappa_{\boldsymbol{p}} \operatorname*{maj}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{0},I)} g_{\boldsymbol{x}}(\boldsymbol{p}), \tag{1.1}$$

$$\text{where } \kappa_{\boldsymbol{p}} \triangleq \begin{cases} 1, & \text{if } \|\boldsymbol{p}\|_2 \leq 1/50 \\ 50\|\boldsymbol{p}\|_2, & \text{if } \|\boldsymbol{p}\|_2 > 1/50 \end{cases}, \text{ and } g_{\boldsymbol{x}}(\boldsymbol{p}) \triangleq f\left(\frac{\boldsymbol{p}}{\kappa_{\boldsymbol{p}}} - \boldsymbol{x}\right) + f(\boldsymbol{x}). \tag{1.2}$$

Here, every $\boldsymbol{p} \in \mathbb{R}^n$ is radially contracted to the point $\boldsymbol{p}/\kappa_{\boldsymbol{p}} \in \mathrm{B}(\boldsymbol{0}, 1/50)$, and $g_{\boldsymbol{x}}(\boldsymbol{p})$ is the self-correction vote contributed by the direction $\boldsymbol{x} \in \mathbb{R}^n$, weighted as per the distribution $\mathcal{N}(\boldsymbol{0}, I)$. The majority of these votes decides the value of $g(\boldsymbol{p})$. If no violations of additivity are found, then the g is proven to be a well defined, additive function. Next, they build an $(O(\log\frac{1}{\varepsilon}))$-query access for g, and using it, they estimate the distance between f and g. This allows them to build a *distribution-free tester*, which is desirable as we may have no prior information about the underlying distribution. This thus avoids making any unrealizable assumptions about it.

For a measurable function $f : \mathbb{R}^n \to \mathbb{R}$, a distance parameter $\varepsilon > 0$, an unknown but samplable distribution $\mathcal{D}$ supported over $\mathbb{R}^n$, and a property P, f is ε-far from P with respect to $\mathcal{D}$ if

$$\delta_{\mathcal{D}}(f, \mathsf{P}) \triangleq \inf_{g\in\mathsf{P}} \left\{ \Pr_{\boldsymbol{x}\sim\mathcal{D}}[f(\boldsymbol{x}) \neq g(\boldsymbol{x})] \right\} > \varepsilon.$$

An algorithm is a *distribution-free tester* for P, if given query access to f, sample access to $\mathcal{D}$ and $\mathcal{N}(\boldsymbol{0}, I)$, and $\varepsilon > 0$, it accepts all $f \in \mathsf{P}$ with probability at least $2/3$, and rejects all f such that $\delta_{\mathcal{D}}(f, \mathsf{P}) > \varepsilon$, with probability at least $2/3$.

To shave off the $\log\frac{1}{\varepsilon}$ factor, we only need to modify the distance estimator part of the tester.

THEOREM 1.3. *There exists a distribution-free, one-sided error $O(1/\varepsilon)$-query additivity tester over the reals.*

We replace the $O(\log\frac{1}{\varepsilon})$-query access for g at any point $\boldsymbol{p} \in \mathbb{R}^n$ with $g_{\boldsymbol{x}}(\boldsymbol{p})$, where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, I)$ is a random direction. Analysis-wise, the proof for completeness remains the same. For soundness, using the approach of [HK04], we argue that $g(\boldsymbol{p})$ agrees with at least $9/10$ of the $g_{\boldsymbol{x}}(\boldsymbol{p})$'s (not just at least $1/2$ of them). Consequently, evaluating $g_{\boldsymbol{q}}(\boldsymbol{p})$ in a random direction $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, I)$, i.e. $g_{\boldsymbol{x}}(\boldsymbol{p})$ is sufficient to successfully query $g(\boldsymbol{p})$.

Low-degree testing over the reals. We observe that the same modification applies to the Distribution-free Low degree testers (Algorithms 2, 4, and 6), as well as the Approximate Additivity (Linearity) Tester (Algorithm 8) of [ABF+23], shaving off the $\log \frac{1}{\varepsilon}$ factor from their query complexity ($O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$ for constant d case), thus giving optimal query complexity for constant d as well. We state the results here. The proofs of these improvements are very similar to the proof of Theorem 1.3, and will appear in the full version of this paper.

THEOREM 1.4. *Let $d \in \mathbb{N}$, and for $L > 0$, suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a function that is bounded in the ball $\mathrm{B}(\mathbf{0}, L)$. Given $\varepsilon > 0$, query access to f, and sampling access to an unknown distribution $\mathcal{D}$, there exists a one-sided error, distribution-free, $O(d^5 + d^2/\varepsilon)$-query tester for testing whether f is a degree-d polynomial, or is ε-far from degree-d polynomials over $\mathcal{D}$.*

THEOREM 1.5. *Let $d \in \mathbb{N}$, for $L > 0$, suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a function that is bounded in $\mathrm{B}(\mathbf{0}, L)$, and for $\varepsilon \in (0, 1), R > 0$, let $\mathcal{D}$ be an $(\varepsilon/4, R)$-concentrated distribution, i.e., $\Pr_{\boldsymbol{p} \sim \mathcal{D}}[\boldsymbol{p} \in \mathrm{B}(\mathbf{0}, R)] \geq 1 - \varepsilon/4$. Given $\alpha > 0, \beta \geq 2^{(2n)^{O(d)}} (R/L)^d \alpha$, query access to f, and sampling access to $\mathcal{D}$, there is a one-sided error, $O(d^5 + d^2/\varepsilon)$-query tester which distinguishes between the cases when:*

- *f is pointwise α-close to some degree-d polynomial, say h, i.e., $|f(\boldsymbol{x}) - h(\boldsymbol{x})| \leq \alpha$, for every $\boldsymbol{x} \in \mathbb{R}^n$, and,*
- *for every degree-d polynomial $h \colon \mathbb{R}^n \to \mathbb{R}$, $\Pr_{\boldsymbol{p} \sim \mathcal{D}}[|f(\boldsymbol{p}) - h(\boldsymbol{p})| > \beta] > \varepsilon$.*

THEOREM 1.6. *For $d, B, R > 0$, let $B' \geq 16 \cdot \max\{n^{5/2+2d} d^{2d}, B^2 R^2/\sqrt{n}\}$ be a multiple of B. Let $\mathsf{L} = \frac{1}{B}\mathbb{Z}^n$ and $\mathsf{L}' = \frac{1}{B'}\mathbb{Z}^n$. Given $\varepsilon > 0$, query access to a function $f : \mathbb{R}^n \to \mathbb{R}$, and sample access to an unknown $(\varepsilon/4, R)$-concentrated distribution $\mathcal{D}$ supported on L, there is a one-sided error, $O(d^5 + d^2/\varepsilon)$-query tester for testing whether f is a degree-d polynomial, or is ε-far from every degree-d polynomials over $\mathcal{D}$. The tester queries f on points in L'.*

THEOREM 1.7. *Let $\alpha, \varepsilon > 0$ and $\mathcal{D}$ be an unknown $(R, \varepsilon/4)$-concentrated distribution. There exists a one-sided error, $O(1/\varepsilon)$-query tester for distinguishing between the case when f is pointwise α-close to some additive function and the case when, for every additive function h, $\Pr_{\boldsymbol{p} \sim \mathcal{D}}[|f(\boldsymbol{p}) - h(\boldsymbol{p})| > O(Rn^{1.5}\alpha)] > \varepsilon$.*

1.3 Notation We use $[n]$ for the set $\{1, 2, \ldots, n\}$ and log to denote logarithms with base 2. We use bold face, e.g. $\boldsymbol{p}, \boldsymbol{q}$, etc., to denote vectors/points over the reals.

Organization. Section 2 is devoted to the online model, wherein we provide preliminaries for online testing, as well as prove Theorem 1.1, and Theorem 1.2. Next, in Section 3, we prove Theorem 1.3. The proofs of Theorem 1.4, Theorem 1.5, Theorem 1.6, and Theorem 1.7, being similar to the proof of Theorem 1.3, will appear in the full version of this paper.

2 Testing over $\mathbb{F}_2$ with online manipulations

We start by introducing the model. The online manipulation-resilient testing model was introduced first by Kalemaj et al. [KRV23] and formally defined by Ben-Eliezer et al. [BKMR24]. We follow the definitions of the latter, as specified below.

The input is accessed via a sequence $\{\mathcal{O}_i\}_{i \in \mathbb{N}}$ of oracles, where $\mathcal{O}_i$ is used to answer the i-th query. The oracle $\mathcal{O}_1$ gives access to the original input (e.g., when the input is a function f, we have $\mathcal{O}_1 \equiv f$), and subsequent oracles are objects of the same type as the input (e.g., functions with the same domain and range). Each such oracle is obtained by the adversary by modifying the previous oracle to include a growing number of erasures/corruptions as i increases. We use $\mathrm{Dist}(\mathcal{O}, \mathcal{O}')$ for the Hamming distance between the two oracles (i.e., the number of queries for which they give different answers). We let $t \in \mathbb{N}$ denote the number of *erasures* (or *corruptions*) *per query*.

DEFINITION 2.1. (FIXED-RATE AND BUDGET-MANAGING ADVERSARIES) *Fix a parameter $t > 0$. A sequence[1] of oracles $\mathcal{O} = \{\mathcal{O}_i\}_{i \in \mathbb{N}}$ is induced by a t-*online fixed-rate adversary *if $\mathcal{O}_1$ is equal to the input and, for all $i \in \mathbb{N}$,*

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

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

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

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

Finally, we consider two types of manipulations to the input: erasures and corruptions.

DEFINITION 2.2. (ERASURE AND CORRUPTION ADVERSARIES) *Let $\perp$ represent the erasure symbol. A sequence of oracles $\mathcal{O} = \{\mathcal{O}_i\}_{i\in\mathbb{N}}$ is induced by an* erasure adversary *if for all $i \in \mathbb{N}$ and data points x,*

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

In contrast, a corruption adversary *can change answers to anything in the range, i.e., $\mathcal{O}_i$ can be any valid input for the computational task at hand.*

A property P denotes a set of objects (typically, a set of functions). Intuitively, it represents the set of positive instances for the testing problem. The (relative Hamming) distance between a function f and a property P, denoted $dist(f, \mathsf{P})$, is the smallest fraction of function values of f that must be changed to obtain a function in P. Given a proximity parameter $\varepsilon \in (0, 1)$, we say that f is *ε-far* from P if $dist(f, \mathsf{P}) \geq \varepsilon$. An online tester is given a proximity parameter ε and the rate of erasures (or corruptions) t.

DEFINITION 2.3. (ONLINE ε-TESTER) *Fix $\varepsilon \in (0, 1)$. An online ε-tester $\mathcal{T}$ for a property $\mathcal{P}$ that works in the presence of a specified adversary (e.g., t-online erasure budget-managing adversary) is given access to an input function f via a sequence of oracles $\mathcal{O} = \{\mathcal{O}_i\}_{i\in\mathbb{N}}$ induced by that type of adversary. For all adversarial strategies of the specified type,*

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

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

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

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

We are now ready to prove Theorem 1.1. Let us define an important parameter, according to which we choose our strategy against the adversary:

$$m := 4\lceil \log(t) + 10/\varepsilon \rceil.$$

When $m \leq n/3$ (Case I), meaning that ε is not too small and t is not too big, we can safely use the tester of [BKMR24] with the parameter m above. This allows us to significantly simplify the choice of m and leads to a cleaner analysis of the tester. When $m > n/3$ (Case II), however, we revert to sample-based testing, showing this strategy easily defeats the adversary and coincides with our desired query complexity, matching the lower bound. We analyze each of the two cases separately.

2.1 Case I: $m \leq n/3$

For this case we use the following primitive from [KRV23]:

Algorithm 1: $\textsc{XorTest}_k$

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

1 Query k points $x_1, \ldots, x_k \in \{0,1\}^n$ chosen uniformly at random (with replacement).

2 Query point $y = \bigoplus_{i\in[k]} x_i$.

3 **Accept** if $f(y) = \prod_{i\in[k]} f(x_i)$ (equivalently, if $f(y) \cdot \prod_{i\in[k]} f(x_i) = 1$); otherwise, **reject**.

An improved soundness guarantee for this test was shown by [BKMR24]:

LEMMA 2.1. (SOUNDNESS) *If f is ε-far from linear, and $k \geq 2$ is even, then*

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

The algorithm is as follows:

Algorithm 2: Online-Erasure-Resilient Linearity Tester

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

1 **Repeat** 6 **times**

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

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

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

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

6 **Reject** ▷ This implies no erasures in this iteration.

7 **Accept**

In this case, Algorithm 2 performs a simplified version of the tester of [BKMR24]. We upper bound the probability of seeing an erasure at any given iteration, simplifying the analysis of [BKMR24] for this restricted regime (in particular $\varepsilon \geq \Omega(1/n)$ and $t \leq 2^{O(n)}$).

LEMMA 2.2. *If $m \leq n/3$, the probability that one specific iteration of the loop in line 1 of Algorithm 2 queries an erased point is at most 3/64.*

Proof. If $m \leq n/3$, the algorithm enters the loop and queries a total of $6(m+1)$ points, which induce at most $6t(m+1)$ manipulations through the entire execution. We define three bad events and give upper bounds on their probabilities.

An erasure while querying X. Let B_1 be the event that $\mathcal{O}(x_j) = \perp$ for some point x_j sampled in this iteration, where $j \in [m]$. Each point x_j is sampled uniformly from $\{0,1\}^n$, so the probability it is erased is at most $6t(m+1)/2^n$. By a union bound over all m points, using $m \leq n$, we have

$$\Pr_X [B_1] \leq \frac{6t(m+1)m}{2^n} \leq \frac{6t(m+1)m}{2^m}. \tag{2.3}$$

X induces a bad distribution of y points. For any choice S denote $y_S = \oplus_{j \in S} x_j$, and let B_2 be the event that, in this iteration, there exist two different choices of S leading to the same choice $y \in \{0,1\}^n$. For any two distinct sets $T_1, T_2 \subset [m]$ of size $m/2$ w.l.o.g. there exists an element $\ell \in T_1 \setminus T_2$. Fix all entries in X besides x_ℓ. The value of y_{T_2} is now fixed, but over the random choice of $x_\ell \in \{0,1\}^n$, the vector y_{T_1} is uniform over $\{0,1\}^n$. Thus, $\Pr_{x_\ell} [y_{T_1} = y_{T_2}] = 2^{-n}$ and, consequently,

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

where both expectations are over all entries in X besides x_ℓ, which are drawn independently from x_ℓ. We use a union bound over all pairs of subsets T_1 and T_2, and the fact $m \leq n/3$ to get

$$\Pr [B_2] = \Pr_X \left[\exists T_1 \neq T_2 \text{ such that } y_{T_1} = y_{T_2} \right] \leq \frac{2^{2m}}{2^n} \leq 2^{-m}. \tag{2.4}$$

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

$$\Pr \left[B_3 \mid \overline{B_2} \right] \leq \frac{6t(m+1)}{\binom{m}{m/2}} \leq \frac{6t(m+1)\sqrt{2m}}{2^m}. \tag{2.5}$$

In terms of the bad events, we wish to bound $\Pr[B_1 \cup B_3]$. By using a union bound over B_1 and B_3 and then the law of total probability to compute $\Pr[B_3]$, we get

$$\Pr [B_1 \cup B_3] \leq \Pr [B_1] + \Pr \left[\overline{B_2} \right] \cdot \Pr \left[B_3 \mid \overline{B_2} \right] + \Pr [B_2] \cdot \Pr [B_3 \mid B_2]$$

$$\leq \Pr[B_1] + \Pr\left[B_3 \mid \overline{B_2}\right] + \Pr[B_2].$$

We next combine the bounds from (2.3), (2.4) and (2.5), and use $t \leq 2^{\frac{m}{4}-10}$ (which is implied by $\varepsilon \leq 1/2$).

$$\Pr[B_1 \cup B_3] \leq \frac{16t(m+1)m}{2^m} \leq \frac{(m+1)m}{64 \cdot 2^{\frac{3m}{4}}} \leq \frac{3}{64},$$

where the last transition holds since $\frac{(m+1)m}{2^{\frac{3m}{4}}} \leq 3$ for all positive values of m. ∎

We are ready to prove the correctness of Algorithm 2, showing Theorem 1.1 for case I.

Proof. If Algorithm 2 entered the loop in line 1, then it makes $6(m+1) = O(\log t + 1/\varepsilon)$ queries.

We next analyse the algorithm in the presence of erasures. It is easy to see the algorithm always accepts all linear functions. Now, fix an adversarial (budget-managing) strategy and suppose that the input function is ε-far from linear. By Lemma 2.1 and since $k = m/2 \geq 1/\varepsilon$ is even, the probability that one iteration of the loop in line 1 samples a witness of nonlinearity is at least $\min\{1/4\ ,\ k\varepsilon/2\} \geq \min\{1/4\ ,\ 1/2\} = 1/4$.

By Lemma 2.2, the probability that an erasure is seen in a specific iteration is at most $1/16$. By a union bound, the probability of a single iteration seeing an erasure or not selecting a witness of nonlinearity is at most $1 - \frac{1}{4} + \frac{1}{16} = 1 - \frac{3}{16}$. Algorithm 2 errs only if this occurs in all iterations. By independence of random choices in different iterations, the failure probability is at most

$$\left(1 - \frac{3}{16}\right)^6 \leq e^{-\frac{18}{16}} \leq \frac{1}{3},$$

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

Finally, we show that Algorithm 2 has two-sided error at most $1/3$ in the presence of corruptions. The soundness holds with the same analysis, as finding a single violation suffices for this case. For completeness, note that the algorithm can only err if it has seen a manipulation, and the probability of seeing a manipulation at any iteration is at most $3/64$ by Lemma 2.2. Using a union bound over all 6 iterations, the overall probability of seeing any manipulated entry during the entire execution is at most $6 \cdot (3/64) \leq 1/3$. ∎

2.2 Case II: $m > n/3$ In this case, we use [GR16, Theorem 5.1] for the domain $\{0,1\}^n$ and run an algorithm that only uses random samples, their algorithm is simple, and we bring it here for completeness:

Algorithm 3: Goldreich-Ron sampled based tester

Input : Parameters $\varepsilon \in (0, 1/2]$; sample access to $(x, f(x))$

1 $m = O(n)$.
2 Sample $X = (x_1, \ldots, x_m) \in (\{0,1\}^n)^m$ uniformly at random.
3 **Accept** if $span(X) \neq \{0,1\}^n$. ▷ w.h.p. this doesn't happen;
4 Let $Y \subset X$ be an arbitrary basis for $\{0,1\}^n$ and $g : \{0,1\}^n \to \{0,1\}$ the unique linear function that agrees with f on all points in Y.
5 **for** $O(1/\varepsilon)$ *times* **do**
6 Sample $z \sim \{0,1\}^n$.
7 **Reject** if $f(z) \neq g(z)$.
8 **Accept**.

LEMMA 2.3. ([GR16, THEOREM 5.1 (1)], FOR BOOLEAN FUNCTIONS) *There is a one-sided error sample-based tester of sample complexity $O(1/\varepsilon + n)$ for testing linearity of functions of the form $f : \{0,1\}^n \to \{0,1\}$.*

Next, we show that if the algorithm uses only uniform random samples, then the adversary loses its power, in the sense that we, most likely, won't see any manipulation.

THEOREM 2.1. *Let $\mathcal{T}$ be a sample-based tester (for some property $\mathcal{P}$) with input length N and distance parameter ε that uses q uniformly random samples and succeeds with probability $1-\delta$. Then the same tester with the same number of queries succeeds with probability $1-2\delta$ in the presence of t-online-manipulation adversary for any $t \leq \delta \cdot N/q^2$.*

Proof. To analyze the tester, we note that all queries in this test are random samples, and the total number of manipulations made is qt. Therefore, each sample has probability of at most qt/N to see an erasure. By a union bound over all the samples taken, the overall probability to sample any previously-manipulated points is at most $q^2t/N \leq \delta$. Adding this error to the original error of the tester $\mathcal{T}$ completes the proof. ∎

REMARK 2.1. *If $\mathcal{T}$ is a one-sided error tester, we can keep it one-sided against online-erasure adversary by accepting whenever we see an erasure. Otherwise, we get a two-sided error tester.*

COROLLARY 2.1. *There is a one-sided error sample-based tester for testing linearity against online-erasure adversaries that uses $q = O(1/\varepsilon + n)$ queries, succeeds with probability $2/3$ and works for all $\varepsilon \in (0, 1/2)$ and $t \leq O(\min\{2^n/n^2, \varepsilon^2 2^n\})$.*

We are now ready to prove Theorem 1.1 for Case II.

Proof. The query complexity is $q = O(n + 1/\varepsilon)$, and in our case, it is at most $O(m + 1/\varepsilon) = O(\log t + 1/\varepsilon)$. ∎

2.3 A testing impossibility result In this section we prove Theorem 1.2, that is that testing is impossible when t is too large, by showing a much more general statement. The argument below is formulated for properties of functions $f : \mathbb{F}_q^n \to \mathbb{F}_q$, but can be adjusted to other settings as well (e.g., properties of strings, Σ^n).

The proof leverages a recent elegant argument of [Fis24], that shows c/ε queries do not suffice to ε-test a property $\mathcal{P}$, provided the testing task is not trivial (the precise will appear shortly).

Consider any such non-trivial task in the online manipulation model with large enough t (roughly ε^2-fraction of the input size). On the one hand, after c/ε the tester still cannot distinguish certain YES and NO instances with high enough probability. On the other hand, at this point an online adversary can already manipulate an ε-fraction of the original input, enough to completely erase the initial difference between a YES and a NO instance. Hence, any additional query is useless, and the tester is doomed to fail.

We first review the argument of [Fis24] in the offline model and some of its notations, and later state and prove the impossibility result for the online model.

The offline model. We consider properties of functions[2] $f : \mathbb{F}_q^n \to \mathbb{F}_q$, where a property $\mathcal{P}$ is identified with a subset of all these functions. Fix a property $\mathcal{P}$ such that there exists two input functions $F_{\text{YES}} \in \mathcal{P}$ and F_{NO} that is α-far from $\mathcal{P}$ for some constant $\alpha > 0$. For simplicity, assume the proximity parameter satisfies $\varepsilon = \ell/q^n > 0$ for some integer $\ell \in \mathbb{N}$.

First, we consider a function $G \in \mathcal{P}$ that minimizes the distance to f_{NO} (if such G is not unique, choose one arbitrarily), and denote by $D \subseteq \mathbb{F}_q^n$ the set of inputs on which G and F_{NO} disagree. For any $A \subseteq D$, define

$$G_A(x) := \begin{cases} G(x), & \text{if } x \notin A \\ F_{\text{NO}}(x), & \text{if } x \in A \end{cases}$$

In particular, $G_D \equiv F_{\text{NO}}$, and $G_\emptyset \equiv G$, and evidently G_A has distance exactly $|A|/q^n$ from the property $\mathcal{P}$ (since G minimizes the distance of F_{NO} from $\mathcal{P}$).

The focus from now on is on a task we call (G, D, ℓ)-testing. An algorithm solves this task if it accept G with probability at least $2/3$, but accepts G_A with probability at most $1/3$ for any $A \subseteq D$ such that $|A| = \ell$. In particular, for $\varepsilon = \ell/q^n$, an ε-tester for $\mathcal{P}$ is a (G, D, ℓ)-tester.

It was shown in [Fis24] that w.l.o.g, a randomized (G, D, ℓ)-tester is non-adaptive and only queries points in D. Thus, any such tester simply queries the function on q points $x_1, \ldots x_q$, where each x_i is a random variable supported on D. Furthermore, for each point $z \in D$ we can define the event E_z that the point z was queried (i.e, $x_i = z$ for some $i \in [q]$).

[2]The original argument in [Fis24] is applied to properties of Σ^n, or equivalently functions $f : [n] \to \Sigma$. We use a slightly different to comply with linearity testing.

Lastly, fix a randomized (G, D, ℓ)-tester, and observe the probabilities for any point $z \in D$ to be queried (that is, the probability of E_Z). If the tester makes at most q queries, then

$$\sum_{z \in D} \Pr[E_z] = \sum_{z \in D} \mathbb{E}[\mathbb{1}_{E_z}] = \mathbb{E}\left[\sum_{z \in D} \mathbb{1}_{E_z}\right] \leq \mathbb{E}[q] = q.$$

Denoting by $D' = \{z_1, \ldots, z_\ell\}$ the set with ℓ points that has the smallest probabilities (break ties arbitrarily), we can bound the probability of the event $E_{D'}$ that any point from D' was queried. By a union bound, we have

$$\Pr[E_{D'}] \leq \sum_{z \in D'} \Pr[E_z] \leq \frac{\ell}{|D|} \cdot \sum_{z \in D} \Pr[E_z] \leq \frac{\ell \cdot q}{|D|}.$$

Since we started with f_{NO} that is α-far from $\mathcal{P}$, we have $|D| \geq \alpha \cdot q^n$ as well as $\ell = \varepsilon \cdot q^n$. Hence

$$\Pr[E_{D'}] \leq \frac{\varepsilon \cdot q}{\alpha}.$$

To finish up, consider two executions of the tester with the same random coins, one when fed values from G and another when fed values from $G_{D'}$. The answer can only differ if the event $E_{D'}$ occurred (otherwise, all the oracle answers are the same for both inputs). Thus, the probabilities of acceptance can differ by at most $\Pr[E_{D'}]$, which leads to a contradiction if $q < \alpha/(3\varepsilon)$.

The online model. For the online model, we similarly begin from a guaranteed function f_{NO} that is α-far, and a closest function $G \in \mathcal{P}$ (arbitrarily chosen if not unique) that differs on the set of inputs $D \subseteq \mathbb{F}_q^n$. We consider the same task of (U, D, ℓ)-testing, where we assumed $\varepsilon = \ell/q^n \in (0, \alpha)$.

We denote $\mathcal{F}_n := \left\{f : \mathbb{F}_q^n \to \mathbb{F}_q\right\}$ and $\mathcal{F} = \cup_{n \in \mathbb{N}} \mathcal{F}_n$. A property is simply $\mathcal{P} \subseteq \mathcal{F}$, and we use $\mathcal{P}_n = \mathcal{P} \cap \mathcal{F}_n$. We have the following formal statement:

THEOREM 2.2. *Fix a constant $\alpha > 0$, and a property $\mathcal{P} \subseteq \mathcal{F}$, such that for infinitely many values of n there exist $F_{YES}^n \in \mathcal{P}_n$ and f_{NO}^n that is α-far from $\mathcal{P}$. Fix $\varepsilon = \ell/q^n < \alpha$ for some integer $\ell \in \mathbb{N}$, and $t = (10/\alpha)\varepsilon^2 q^n$. Then there is not ε-tester for $\mathcal{P}$ that is resilient to t manipulations per query.*

Proof. The argument begins as before. For any n we define $G \in mathcalP_n$, and D that depend on f_{NO}^n. Any ε-tester for $\mathcal{P}$ is also a (U, D, ℓ)-tester as defined above. Assume such a tester exists and is resilient to t manipulations per query.

Similarly to before, we define an event for each point $z \in D$, but these events focus only on the first queries. Define E_z to be the event that z is queried within the first $q_0 = \alpha/(10\varepsilon)$ queries by the tester. As before, we can order the points z by the probabilities of E_z and define a set $D' \subseteq D$ consisting of the ℓ points with minimal probabilities. The event that any of these point was queried within the first q_0 queries is denoted by $E_{D'}$ and satisfies

$$\Pr[E_{D'}] \leq \frac{\varepsilon \cdot q_0}{\alpha} < \frac{1}{10}.$$

As before, fix the random coins of a tester, and consider one execution on G and another on $G_{D'}$. Only now, while the tester executes, an online adversary manipulates the inputs. The adversary will simply manipulate the points in D', of which there are exactly ℓ. Indeed, by the time q_0 queries has been made, the adversary can manipulate all entries in D' since

$$q_0 t = \frac{\alpha}{10\varepsilon} \cdot \frac{10\varepsilon^2 q^n}{\alpha} = \varepsilon q^n = \ell.$$

For both executions on G and on $G_{D'}$, the adversary acts the same. An erasing adversary will erase all entries in D', whereas a manipulation adversary will fix their values to those of G.

All in all, for any value of the random points, the tester can only see a difference between an execution on G and one on $G_{D'}$ if the event $E_{D'}$ occurred. Over all random choices, we have $\Pr[E_{D'}] \leq 1/(10)$, and since the tester must accept G with probability at least $2/3$ it also falsely accepts $G_{D'}$ with probability at least $2/3 - 1/(10) > 1/3$. In contradictions.

To summarize, the task of (U, D, ℓ)-testing is impossible against t-manipulation adversary for infinitely many values of n, and so is the task of ε-testing the property $\mathcal{P}$. ∎

To prove Theorem 1.2, we simply apply the theorem with $q = 2$ and $\alpha = 1/2$, as for any $n \geq 1$, the affine non-linear function $1 + x_1$ has distance $1/2$ from linearity.

3 Optimal Linearity Tester over the Reals

In this section, we show an optimal distribution-free additivity tester over the reals. The same tester works for testing linearity if the input function is guaranteed to be continuous.

Algorithm 4: Distribution-free Additivity Tester

Input : Query access to $f\colon \mathbb{R}^n \to \mathbb{R}$, sampling access to an unknown distribution $\mathcal{D}$, proximity parameter $\varepsilon > 0$;

Reject if TestAdditivity(f) returns **Reject**;
for $N_4 \leftarrow O(1/\varepsilon)$ *times* **do**
  Sample $\boldsymbol{p} \sim \mathcal{D}$ and $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, I)$;
  Reject if $f(\boldsymbol{p}) \neq g_{\boldsymbol{x}}(\boldsymbol{p})$ ▷ Where $g_{\boldsymbol{x}}(\boldsymbol{p}) = \kappa_{\boldsymbol{p}}\left(f(\boldsymbol{p}/\kappa_{\boldsymbol{p}} - \boldsymbol{x}) + f(\boldsymbol{x})\right)$. as in (1.2)
Accept.

Algorithm 5: Additivity Subroutine

Procedure TestAdditivity(f)
  Input : Query access to $f\colon \mathbb{R}^n \to \mathbb{R}$;
  for $N_5 \leftarrow O(1)$ *times* **do**
    Sample $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I)$;
    Reject if $f(-\boldsymbol{x}) \neq -f(\boldsymbol{x})$;
    Reject if $f(\boldsymbol{x} - \boldsymbol{y}) \neq f(\boldsymbol{x}) - f(\boldsymbol{y})$;
    Reject if $f\left(\frac{\boldsymbol{x}-\boldsymbol{y}}{\sqrt{2}}\right) \neq f\left(\frac{\boldsymbol{x}-\boldsymbol{z}}{\sqrt{2}}\right) + f\left(\frac{\boldsymbol{z}-\boldsymbol{y}}{\sqrt{2}}\right)$;
  Accept.

[FY20, Theorem 1] showed that $O\left(\frac{1}{\varepsilon}\log(\frac{1}{\varepsilon})\right)$ queries suffice for testing if $f : \mathbb{R}^n \to \mathbb{R}$ is linear. We simplify their algorithm, slightly modify the analysis accordingly, and show that $O(1/\varepsilon)$ queries suffice instead. Our simplified tester is given in Algorithm 4. First it runs Algorithm 5 (TestAddativity), which is exactly the same subroutine from [FY20], and then it proceeds similarly to their algorithm except:

- when we want to evaluate the self-corrected function $g(\boldsymbol{p})$ (as defined in (1.1)), we now use only one random direction ($\boldsymbol{x} \in \mathbb{R}^n$) and in only two queries to f we output $g_{\boldsymbol{x}}(\boldsymbol{p})$.
- Our approach differs from [FY20]'s at this juncture, as [FY20] designed a separate procedure, Query-g, wherein they evaluate $\{g_{\boldsymbol{x}_i}(\boldsymbol{p}), \boldsymbol{x}_i \sim \mathcal{N}(\mathbf{0}, I)\}$ in $O(\log(1/\varepsilon))$ random directions ($\boldsymbol{x}_i$'s), do a consistency check, i.e., ensure $g_{\boldsymbol{x}_i}(\boldsymbol{p})$'s are all the same, and then output $g_{\boldsymbol{x}_1}(\boldsymbol{p})$. Their idea was to get an accurate evaluation of (the majority definition of) g at all points.

We observe that the approach of [HK04] can be applied to this case as well, i.e., it is enough to correctly evaluate g with high (constant) probability at each point. Intuitively, one only needs to have a correct evaluation of $g(\boldsymbol{p})$ on one point $\boldsymbol{p}$ for which $f(\boldsymbol{p}) \neq g(\boldsymbol{p})$. We begin by noting down the relevant useful result from [FY20]:

Lemma 3.1. ([FY20, Lemma 8]) *If Algorithm 5 accepts with probability at least* $1/10$*, then g is a well-defined, additive function on $\mathbb{R}^n$, and furthermore, for every $\boldsymbol{p} \in \mathbb{R}^n$,*

$$\Pr_{\boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}\left[g(\boldsymbol{p}) \neq g_{\boldsymbol{x}}(\boldsymbol{p})\right] < 1/2.$$

We now prove Theorem 1.3.

Proof. The proof of the YES case remains the same as in [FY20]: if f is linear, Algorithm 5 always accepts, and moreover $g \equiv f$, ensuring Algorithm 4 also accepts f.

No case: f is ε-far from all additive functions. If Algorithm 5 rejects f w.p. $> 2/3$, we are done. Otherwise by Lemma 3.1, we have: g is a well defined, additive function on $\mathbb{R}^n$, so $\delta_{\mathcal{D}}(f,g) \geq \varepsilon$. And again by Lemma 3.1, for every $\boldsymbol{p} \in \mathbb{R}^n$ we have

$$\Pr_{\boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}[g(\boldsymbol{p}) = g_{\boldsymbol{x}}(\boldsymbol{p}) | f(\boldsymbol{p}) \neq g(\boldsymbol{p})] = \Pr_{\boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}[g(\boldsymbol{p}) = g_{\boldsymbol{x}}(\boldsymbol{p})] \geq 1/2. \tag{3.6}$$

$$\begin{aligned}
\implies \Pr_{\substack{\boldsymbol{p}\sim\mathcal{D}\\ \boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}}[f(\boldsymbol{p}) \neq g_{\boldsymbol{x}}(\boldsymbol{p})] &\geq \Pr_{\substack{\boldsymbol{p}\sim\mathcal{D}\\ \boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}}[f(\boldsymbol{p}) \neq g(\boldsymbol{p}) \wedge g(\boldsymbol{p}) = g_{\boldsymbol{x}}(\boldsymbol{p})] \\
&= \underbrace{\Pr_{\boldsymbol{p}\sim\mathcal{D}}[f(\boldsymbol{p}) \neq g(\boldsymbol{p})]}_{=\delta_{\mathcal{D}}(f,g)\geq\varepsilon} \cdot \Pr_{\boldsymbol{p}\sim\mathcal{D},\boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}[g(\boldsymbol{p}) = g_{\boldsymbol{x}}(\boldsymbol{p}) | f(\boldsymbol{p}) \neq g(\boldsymbol{p})] \\
&\geq \varepsilon \int_{\boldsymbol{p}\in\mathrm{supp}(\mathcal{D})\subseteq\mathbb{R}^n} \Pr_{X\sim\mathcal{D}}[X = \boldsymbol{p}] \underbrace{\Pr_{\boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}[g(\boldsymbol{p}) = g_{\boldsymbol{x}}(\boldsymbol{p})]}_{\geq 1/2, \forall \boldsymbol{p}\in\mathbb{R}^n, \text{ by (3.6)}} d\mu_{\mathcal{D}} \\
&\geq \frac{\varepsilon}{2} \underbrace{\int_{\boldsymbol{p}\in\mathrm{supp}(\mathcal{D})\subseteq\mathbb{R}^n} \Pr_{X\sim\mathcal{D}}[X = \boldsymbol{p}] d\mu_{\mathcal{D}}}_{=1} = \varepsilon/2.
\end{aligned}$$

$\implies \Pr_{\substack{\boldsymbol{p}\sim\mathcal{D}\\ \boldsymbol{x}\sim\mathcal{N}(\mathbf{0},I)}}[f(\boldsymbol{p}) = g_{\boldsymbol{x}}(\boldsymbol{p})] \leq 1 - \varepsilon/2$, which when applied over all N_4 iterations of Algorithm 4 gives us

$$\begin{aligned}
A &\triangleq \Pr[\text{Algorithm 4 doesn't reject } f \mid \text{ Algorithm 5 doesn't reject } f] \\
&= \Pr[\text{line 4 doesn't reject } f, \forall N_4 \text{ iterations} \mid \text{Algorithm 5 doesn't reject } f] \\
&\leq (1 - \varepsilon/2)^{N_4} \leq (1 - \varepsilon/2)^{O(1/\varepsilon)} \leq 1/3.
\end{aligned}$$

Therefore, we get $\Pr[\text{Algorithm 4 rejects } f] \geq \min\{\Pr[\text{Algorithm 5 rejects } f], 1 - A\} \geq 2/3$. ∎

References

[ABF+23] Vipul Arora, Arnab Bhattacharyya, Noah Fleming, Esty Kelman, and Yuichi Yoshida. Low degree testing over the reals. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 738–792. SIAM, 2023. 4

[BCH+96] Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996. 1, 2

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability-towards tight results. *SIAM Journal on Computing (SICOMP)*, 27(3):804–915, 1998. 2

[BKMR24] Omri Ben-Eliezer, Esty Kelman, Uri Meir, and Sofya Raskhodnikova. Property testing with online adversaries. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2024. 2, 3, 4, 5, 6

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993. 1, 2

[BS94] Mihir Bellare and Madhu Sudan. Improved non-approximability results. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 184–193, 1994. 2

[BSVW03] Eli Ben-Sasson, Madhu Sudan, Salil P. Vadhan, and Avi Wigderson. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 612–621, 2003. 2

[FGL+96] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. 2

[Fis24] Eldar Fischer. A basic lower bound for property testing. *arXiv preprint arXiv:2403.04999*, 2024. 1, 8

[FY20] Noah Fleming and Yuichi Yoshida. Distribution-Free Testing of Linear Functions on $\mathbb{R}^n$. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 22:1–22:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 1, 3, 10

[GGR98] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998. 1, 2

[GR16] Oded Goldreich and Dana Ron. On sample-based testers. *ACM Transactions on Computation Theory (TOCT)*, 8(2):1–54, 2016. 2, 7

[Ham05] Georg Hamel. Eine basis aller zahlen und die unstetigen lösungen der funktionalgleichung: $f(x+y) = f(x) + f(y)$. *Mathematische Annalen*, 60(3):459–462, 1905. 3

[HK04] Shirley Halevy and Eyal Kushilevitz. Testing monotonicity over graph products. *Proceedings, International Colloquium on Automata, Languages and Programming (ICALP)*, 2004. 3, 10

[HW03] Johan Håstad and Avi Wigderson. Simple analysis of graph tests for linearity and PCP. *Random Structures and Algorithms*, 22(2):139–160, 2003. 2

[KLX10] Tali Kaufman, Simon Litsyn, and Ning Xie. Breaking the epsilon-soundness bound of the linearity test over GF(2). *SIAM Journal on Computing (SICOMP)*, 39(5):1988–2003, 2010. 2

[KRV23] Iden Kalemaj, Sofya Raskhodnikova, and Nithin Varma. Sublinear-time computation in the presence of online erasures. *Theory of Computing*, 19(1):1–48, 2023. 2, 3, 4, 5

[MZ24] Dor Minzer and Kai Zhe Zheng. Adversarial low degree testing. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 4395–4409. SIAM, 2024. 2, 3

[RR16] Sofya Raskhodnikova and Ronitt Rubinfeld. Linearity testing/testing Hadamard codes. In *Encyclopedia of Algorithms*, pages 1107–1110. Springer, 2016. 2

[RS96] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing (SICOMP)*, 25:647–668, 1996. 1

[Sam07] Alex Samorodnitsky. Low-degree tests at large distances. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 506–515, 2007. 2

[ST98] Madhu Sudan and Luca Trevisan. Probabilistically checkable proofs with low amortized query complexity. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 18–27, 1998. 2

[ST00] Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 191–199, 2000. 2

[ST09] Alex Samorodnitsky and Luca Trevisan. Gowers uniformity, influence of variables, and PCPs. *SIAM Journal on Computing (SICOMP)*, 39(1):323–360, 2009. 2

[SW06] Amir Shpilka and Avi Wigderson. Derandomizing homomorphism testing in general groups. *SIAM Journal on Computing (SICOMP)*, 36(4):1215–1230, 2006. 2

[Tre98] Luca Trevisan. Recycling queries in PCPs and in linearity tests (extended abstract). In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 299–308, 1998. 2