The Pennsylvania State University
The Graduate School
College of Engineering

ROBUST MODELS FOR PROPERTY TESTING

A Dissertation in
Computer Science and Engineering
by
Kashyap Dixit

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2015

The dissertation of Kashyap Dixit was reviewed and approved* by the following:

Martin Fürer
Professor of Computer Science
Dissertation Co-Advisor
Co-Chair of Committee

Sofya Raskhodnikova
Professor of Computer Science
Dissertation Co-Advisor
Co-Chair of Committee

Piotr Berman
Professor of Computer Science

Jason Morton
Professor of Mathematics

Raj Acharya
Professor of Computer Science
Head of Department

*Signatures are on file in the Graduate School.

Abstract

Property testing [Rubinfeld Sudan 96,Goldreich Goldwasser Ron 98] is a formal framework for studying approximate sublinear time randomized algorithms for decision problems. These algorithms have black-box query access to the input functions. Property testers are required to distinguish between functions that satisfy a given property from those that are 'far' from satisfying the property. Informally, a function is far from satisfying a given property if many function values need to be changed to satisfy the property. The notion of distance from a property is central to property testing. The distance of a function $f : \mathcal{D} \to \mathbb{R}$ from a property $\mathcal{P}$ is the smallest distance between f and any function $g : \mathcal{D} \to \mathbb{R}$ that satisfies $\mathcal{P}$. One of the most widely studied model [Rubinfeld Sudan 96,Goldreich Goldwasser Ron 98] uses the relative *Hamming distance* as the notion of distance between two functions. That is, the distance between two functions $f : \mathcal{D} \to \mathbb{R}$ and $g : \mathcal{D} \to \mathbb{R}$ is the fraction of domain points on which f and g differ. A long line of work has been dedicated to the design and study of sublinear time algorithms for a number of function properties using this model.

This works focuses on studying practical generalizations of the aforementioned model. In the first part, we work with generalizing the notion of distance between functions. The relative Hamming distance can also be viewed as the distance between functions with respect to the uniform distribution. As part of our study, we design *optimal* testers for a large class of functions properties, called the bounded derivative properties, when the distance is measured with respect to a known or an unknown product distribution. The class of bounded derivative properties include well studied function properties like monotonicity and the Lipschitz property.

The second part of our work focuses on a generalization of the input access model. In many practical scenarios, a black-box access to the whole input function might not be possible. Some of the function values might not be available to the tester due to privacy requirements or adversarial erasures.We refer to such domain points as *erased*. The location of an erasure becomes known to the tester only upon querying the respective domain point. We design efficient erasure-resilient sublinear time testers for a large number of properties including the bounded derivative properties and convexity. Some of our testers are almost optimal, even in the case when a constant fraction of points are erased.

Table of Contents

Acknowledgments

I would like to express my heartiest gratitude to my advisors Martin and Sofya. Martin has always given me a lot of encouragement, in all aspects of my life. He gave me total freedom to work on the topics I like to. The discussions with him were always constructive. Every conversation with him has made me feel better and encouraged me to work and prove results. Sofya has introduced me to the beautiful area of property testing, where I could produce new results which are a part of this thesis. I would like extend my gratitude towards all the instructors of Penn State whose courses have been instrumental in enriching my knowledge and shaping my research. I would also like to acknowledge my collaborators who have been an important part of all the research I have been a part of during my PhD: Deeparnab Chakrabarty, Martin Furer, Madhav Jha, C. Seshadhri, Sofya Raskhodnikova, Abhradeep Thakurta and Nithin Varma. Finally, I would like to thank my family members for standing by all my decisions and endeavors.

I also extend my humble gratitude towards the National Science Foundation whose grants, namely, CCF-0964655, CCF-1320814 and NSF career award CCF-0845701 supported my research.

Dedication

To my parents (Mr. Harish Kumar Dixit and Mrs. Alka Dixit)

Chapter 1

Introduction

Big data is a broad term for data so huge that traditional data processing algorithms perform poorly on it. In some cases, the studied data might be so massive that it does not fit into memory. For such data, even the linear time computation may be too slow or infeasible. In this thesis, we study algorithms that run in time sublinear in the size of the input data. In particular, they access only a small portion of the entire data and output the correct answer with high probability. More specifically, we investigate sublinear time algorithms that determine whether the data approximately satisfies some desired property.

Formally, we view a dataset as a function over some discrete domain $\mathcal{D}$. For example, the classical problem of testing whether an array of n numbers is sorted in nondecreasing order can be viewed as a problem of testing whether a function $f : [n] \to \mathbb{R}$ is monotone (nondecreasing), where $[n]$ denotes the set $\{1, 2, \ldots, n\}$. In this case, data at position i in the list corresponds to the value of function f at index i. We can further generalize the property of monotonicity to two dimensional arrays or grids. In this case, entries in every row and column must be sorted in a non-decreasing order. In general, for any number of dimensions say d, we can test if function $f : [n]^d \to \mathbb{R}$ is monotone. The set $[n]^d$ is referred to as *hypergrid*. Another important property to test is *convexity*. A convex function $f : [n] \to \mathbb{R}$ is amenable to many efficient optimization algorithms used in machine learning. In general, we study properties of real-valued functions on hypergrid domains.

Property testing [RS96, GGR98] is a formal framework for studying approximate sublinear time algorithms for decision problems. Property testers are required to distinguish between functions that satisfy a given property from those that are 'far' from satisfying the property. Informally, a function is far from satisfying a given property if many function values need to be changed to satisfy the property. The notion of distance from a property is central to property testing. The distance of a function $f : \mathcal{D} \to \mathbb{R}$ from a property $\mathcal{P}$ is the smallest distance between f and any function $g : \mathcal{D} \to \mathbb{R}$ that satisfies $\mathcal{P}$. One of the most widely studied model [RS96, GGR98] uses the relative

Hamming distance as the notion of distance between two functions. That is, the distance between two functions $f : \mathcal{D} \to \mathbb{R}$ and $g : \mathcal{D} \to \mathbb{R}$ is the fraction of domain points on which f and g differ.

Property testers have found applications in many areas. Testers for monotonicity have applications in query optimization [BKF+11]. Property testers have also been used in program checking [RS96], vision [KRTA13, RT14], data privacy [JR13, DJRT13] and PCP [ALM+98].

The goal of this work is to study practical generalizations of this model to overcome its current limitations. The first limitation is the use of relative Hamming distance as notion of distance of a function from a given property. The relative Hamming distance between two functions $f : \mathcal{D} \to \mathbb{R}$ and $g : \mathcal{D} \to \mathbb{R}$ can be rewritten as $\Pr_{x \in_{\mathcal{U}} \mathcal{D}}[f(x) \neq g(x)]$. Here, $\mathcal{U}$ denotes the uniform distribution over $\mathcal{D}$ and the notation $x \in_{\mathcal{U}} \mathcal{D}$ stands for an x sampled uniformly randomly from $\mathcal{D}$. The use of uniform distribution to define distance makes the model rather restrictive. A natural generalization of this notion of distance would be $\Pr_{x \in_{\Gamma} \mathcal{D}}[f(x) \neq g(x)]$, where Γ is an arbitrary distribution defined over $\mathcal{D}$. Many learning algorithms use this notion of distance. The authors in [GGR98] note that this formulation of distance is inspired by PAC learning model [Val84]. Shirley and Halevy [HK08a, HK08b] have designed efficient testers for this model for many problems. But for monotonicity (an important property we study) over hypergrid, they give strong hardness results in [HK05]. Therefore, instead of arbitrary distributions, we work with distances defined with respect to an important class of distributions called product distributions.

A product distribution Π over domain $[n]^d$ is defined as follows. Each coordinate x_i of a domain point $\{x_1, x_2, \ldots, x_d\}$ ($1 \leq i \leq d$) is a random variable coming from a distribution Π_i that is statistically independent from the distribution Π_j for $j \neq i$. We note that [AC06] were first to design monotonicity testers with respect to product distributions. Our monotonicity testers [CDJS15] are strictly better than [AC06] in the cases where underlying product distribution on $[n]^d$ is known or unknown. In [DJRT13], we define a new model for testing privacy and show that the Lipschitz property testers with respect to product distributions are important for designing efficient privacy testers. Our optimal Lipschitz testers from [CDJS15] further improve our privacy testers in [DJRT13]. We study this model in the first part of thesis, where we design testers for a large class of properties called *bounded derivative properties* (BDPs) that includes monotonicity and the Lipschitz property. [1].

Another line of work that we pursue is to generalize property testing to scenarios where oracles have limitations. Most algorithms studied in property testing until now assume the oracle to return function values at all queried domain points. However, in many applications, this assumption

---

[1] We note that apart from models discussed henceforth, we have also worked on testing bounded derivative properties in L_p-testing model defined by [BRY14a]. have not included it in this thesis to make it coherent.

is unrealistic. The oracle may be unable to reveal parts of the data due to privacy concerns, or when some of the values get erased by mistake or by an adversary. Motivated by these scenarios, we propose to study sublinear algorithms that work with partially erased data. The tester has to check whether the function restricted to the accessible (non-erased) part of the domain satisfies a given property or is ‘far’ from satisfying it. The distance between two functions in this case is the Hamming distance between the functions restricted to the non-erased domain points. Our testers can handle a large fraction of erasures, while still having sublinear query complexity.

The rest of this chapter is organized as follows. In Section 1.1, we give an overview of testing bounded derivative properties with respect to product distribution. In Section 1.2, we discuss erasure-resilient property testing and give a glance of the results for a large number of properties including BDPs, convexity and linearity.

1.1 Testing Bounded Derivative Properties With Respect To Product Distributions

Bounded derivative properties (BDPs) capture a large class of well studied properties. For example, monotonicity, one of the first and most studied properties in the literature, is a bounded derivative property as discussed in Chapter 3. Another well studied BDP is the Lipschitz property of functions. With its application to privacy [JR13, DJRT13], it has gained a lot of attention recently. In particular, we show in [DJRT13] how a Lipschitz-tester can be used to test the *privacy preserving property* of an algorithm that makes queries on a sensitive database. These properties are generally studied for the functions defined over hypergrid domains $[n]^d$ (Definition 2.0.1). In this work, we give optimal testers for functions $f : [n]^d \to \mathbb{R}$ when distance from a property is measured with respect to a known or an unknown product distribution (see Definition 2.0.4). Our work subsumes a plethora of work on monotonicity and the Lipschitz property for uniform and product distribution in the past.

One of the main contributions of this work is an *optimal dimension reduction* lemma. Roughly, it says that if an input function is "far" from satisfying a given BDP P, then it is likely to be far from P when restricted to a randomly sampled axis-parallel line in $[n]^d$. Dimension-reduction has been studied in the previous works [GGL$^+$00, DGL$^+$99, AC06, HK08a, JR13] and might be of independent interest. Our dimension reduction reduces the problem of testing a given BDP on $[n]^d$ to testing the function on $O(d/\varepsilon)$ randomly sampled axis-parallel lines. Therefore, if we have an optimal tester for the functions $f : [n] \to \mathbb{R}$, we can use dimension reduction to get an optimal tester for functions $f : [n]^d \to \mathbb{R}$.

To get optimal tester for $f : [n] \to \mathbb{R}$, we use binary search tree based tester. First we show that if f is ε-far from satisfying a given property P, then we can find a violation on function restricted the points that lie on a randomly chosen path in any binary search tree with probability at least ε. Note that the query complexity in an iteration is at most the length of the longest search path in the chosen binary search tree. Therefore, to minimize the query complexity, we choose the optimal binary search tree for a given distribution (see Definition 2.0.5). This is the tree which minimizes the expected depth of a path with respect to the input distribution. Note that the smallest expected depth with respect to a given distribution is at most $O(\log n)$, which is the depth of a balanced binary tree. We can find the minimum expected depth with respect to a distribution using Knuth's algorithm [Knu73].

We complement this result by giving matching lower bounds for the worst query complexity of any (even adaptive, two-sided) tester for bounded derivative properties. To this end, first we give a reduction from testing monotonicity to testing any bounded derivative property. Therefore, giving the lower bound on query-complexity for testing monotonicity suffices for all BDPs. Then we give a lower bound for testing monotonicity with respect to a product distribution by using the framework from [Fis04, CS13b]. All the details are present in Chapter 3.

1.2 Erasure-Resilient Property Testers

Most previously studied models assume a black-box oracle access to the input function they test. But this assumption may not be true in many realistic situations. Some subset of domain might be inaccessible to the tester due to privacy reasons. Also, the the function value might be accidentally or adversarially erased. A tester may not know about an erasure unless it queries that particular point. Most of the testers in standard property testing model [RS96, GGR98] are not very useful in these situations. We define a more robust property testing model called *erasure model* in order to design useful testers in this setting. We give efficient testers for a large number of properties including bounded derivative properties (BDPs), convexity and linearity, etc. We study algorithms that work in the presence of *adversarial erasures*. In other words, the query complexity of an algorithm is the number of queries it makes in the *worst case* over all α-erased input functions.

In Chapter 4, we design new erasure-resilient testers for the bounded derivative properties and convexity. Our testers can handle a large fraction of erasures and still output correct answer with high probability by making poly-logarithmic (in domain size) number of queries. For functions $f : [n] \to \mathbb{R}$, our BDP tester has optimal dependence on n, yet it can handle a large fraction of erasures. Our convexity tester for functions $f : [n] \to \mathbb{R}$ can also handle a large fraction of erasures while still having poly-logarithmic query complexity. Note that all our testers are optimal when there are no erasures. Our testers are based on traversing random binary search paths over non-erased points in $[n]$. In order to bound the query complexity and prove correctness of our testers, we required new structural lemmas about randomized binary search paths.

First we give a brief overview of our monotonicity tester for functions $f : [n] \to \mathbb{R}$. We can get a tester for any BDP P of $f : [n] \to \mathbb{R}$ by reducing the property of testing BDP to the problem of testing monotonicity. For monotoncity, we traverse a randomized binary search path over non-erased points as follows. In the beginning, we keep querying the randomly sampled domain points in the $[n]$ to get a non-erased point called *target point* denoted by t. Then, we construct a randomized search path to t. First, we randomly sample and query points in $[n]$ until we get a non-erased point a_1 in $[n]$. We stop if $a_1 = t$. We reject if the pair a_1, t is violated with respect to monotonicity. Else we choose the one of the sets $\{1, 2, \ldots, a_1 - 1\}$ or $\{a_1 + 1, a_1 + 2, \ldots, n\}$, that contains t. Then we recursively proceed in the appropriate subset. We do this until we reach the point s. Note that we traverse a randomized binary search path in the above process. The above process of traversing a randomized search path is repeated $O(\varepsilon^{-1})$ times. [Dev86] proved that the length of a randomized search path is $O(\log n)$ with high probability. Using our new combinatorial Lemma 4.2.2, we prove that with high probability, all the sets encountered while creating the randomized search paths (that is, the ones containing the target point t) have sufficiently high density of non-erased points. Since the probability of finding a violation in a randomized search path is at least ε, we get a tester with logarithmic complexity.

The tester for convexity of functions $f : [n] \to \mathbb{R}$ is slightly more involved. This tester is also based on the construction of a randomized binary search path. Here too, we test for a *goodness condition* in each interval that we encounter while randomized path construction. Note that the best known tester for convexity of functions $f[n] \to \mathbb{R}$ by [PRR03] also test for a goodness condition like us. But their goodness condition is deterministic and hence not very useful in the erasure-model. First we come up a more general *goodness condition* that is erasure-resilient. This also helps us proving a more general fact. If a function is ε-far from convexity, then with probability at least ε, it violates the goodness condition in at least one of the intervals on every randomly chosen binary search path in a given binary search tree. In particular, this is true even for a randomly constructed binary search path. Therefore, here we check for the goodness condition on $O(\varepsilon^{-1})$ randomly constructed binary search paths. In the first step we check for the goodness condition in the set $[n]$. Eventually, we have to check goodness condition in $O(\varepsilon^{-1} \log n)$ contiguous sets that we encounter in the tester. There is another complication. The goodness condition needs to query consecutive non-erased points. This requires the tester to perform a linear search in every interval that it encounters. This could result in querying a linear number of points in some interval encountered by the tester. To overcome this issue, we prove another combinatorial Lemma 4.4.1 about every randomized binary search path. Corollary 4.4.6 of Lemma 4.4.1 is shows that with high probability, the number of queries made during the linear search in each interval is at most $O(\log n)$.

1.3 Organization

We give some preliminary definitions in Chapter 2. In Chapter 3, we present our optimal testers for bounded derivative properties over product distributions. An important characterization of bounded derivative properties is given using a quasi-metric $\mathfrak{m} : [n]^d \times [n]^d \to \mathbb{R}$ defined in Section 3.2. Tester for functions $f : [n] \to \mathbb{R}$ is presented in Section 3.5 along with the analysis. Then we present our optimal dimension reduction in Section 3.4. First we prove the dimension reduction lemma with respect uniform distribution over hypergrid $[n]^d$ in Section 3.4.1. Then in Section 3.4.2, we give a generalization of quasi-metric $\mathfrak{m}$ which directly extends the proof of dimension reduction for uniform distribution to any product distribution. In Section 3.7, we give the matching lower bounds for testing BDPs over product distributions. First, in Section 3.7.1 we show that testing monotonicity of functions $f : [n]^d \to \mathbb{R}$ can be reduced to testing any bounded derivative property with respect to a product distribution. Therefore, we can focus on getting the lower bounds for monotonicity testing. We show a construction for a bunch of “hard to distinguish” functions for domains $[n]$ and $[n]^d$. In other words, we show that any *deterministic* tester whose query complexity is strictly smaller than our testers, fails to detect a violation on one of the hard functions. This, from Theorem 3.7.5 implies our lower bounds. In Section 3.7.3, we show the construction of such functions on domain $[n]$. Then, in Section 3.7.5, we generalize this construction for domains $[n]^d$.

In Chapter 4, we propose our new erasure-resilient model for testing a large class of properties, including bounded derivative properties, convexity and linearity. In Section 4.1.1, we formally define our model. Then we give our erasure-resilient property testers for monotonicity of functions $f : [n] \to \mathbb{R}$ in Section 4.2, which we analyze in Section 4.2.2. Then in Section 4.3.1, we give a reduction from which we derive a tester for any given BDP P using the monotonicity tester as a subroutine. In Section 4.3.2, we give prove a dimension reduction for BDPs in our erasure-model, using which we analyze our tester for functions $f : [n]^d \to \mathbb{R}$ given in Section 4.3.3. We conclude this chapter by giving an erasure-resilient tester for convexity of functions $f : [n] \to \mathbb{R}$.

Chapter 2

Preliminaries

Property testing [GGR98, RS96] is concerned with designing algorithms that check whether a given function satisfies a given property. Formally, a *property* $\mathcal{P}$ is a subset of functions. A tester solves the relaxed membership problem of distinguishing functions in $\mathcal{P}$ from those 'far' from $\mathcal{P}$. We define 'farness' and tester in Definition 2.0.2 and Definition 2.0.3 respectively.

In this work, we consider the functions defined over hypergrid domains which we define below.

Definition 2.0.1 (Hypergrid, line)**.** *Given $n, d \in \mathbb{N}$, the hypergrid is of size n and dimension d is the set $[n]^d$ associated with an order relation $\preceq$, such that $x \preceq y$ for all $x, y \in [n]^d$ iff $x_i \leq y_i$ for all $i \in [d]$, where x_i (respectively y_i) denote the i^{th} coordinate of x (respectively, y). The special case $[n]$ is called a* line.

A function f is *ε-far from* $\mathcal{P}$ if $\mathsf{dist}(f, g) \geq \varepsilon$ for all functions $g \in \mathcal{P}$. When $\mathcal{P}$ is a property of functions $f : [n]^d \to \mathbb{R}$, distance to the property is usually measured in terms of the fraction of points in the domain $[n]^d$ on which f must be modified in order to satisfy the property. We work with this notion of farness in Chapter 4. In Chapter 3, we use a more general notion of distance, first formulated by [GGR98], defined with respect to a probability distribution on the domain $[n]^d$.

Definition 2.0.2 (Distance to a property)**.** *Let $\mathcal{P}$ be a property (i.e., a set) of functions $f : [n]^d \to \mathbb{R}$. Let Π be a distribution on $[n]^d$. The* distance *$dist_\Pi(f, g)$ between functions $f, g : [n]^d \to \mathbb{R}$ (with respect to the distribution Π) is $\Pr_{x \sim \Pi}[f(x) \neq g(x)]$. The* distance *$dist_\Pi(f, \mathcal{P})$ of a function f to the property $\mathcal{P}$ is $\min_{g \in \mathcal{P}} dist_\Pi(f, g)$. For convenience, we use ϵ_f as a shorthand for $dist_\Pi(f, \mathcal{P})$. We say that f is ϵ-*far *from property $\mathcal{P}$ if $\epsilon_f \geq \epsilon$.*

Definition 2.0.3 (Property tester)**.** *Consider a function $f : [n]^d \to \mathbb{R}$, a property $\mathcal{P}$ and a distribution Π on $[n]^d$. Let $\varepsilon \in (0, 1]$ be the* proximity parameter*. A* Property tester *T with respect to distribution Π on $[n]^d$ is a randomized algorithm that gets oracle access to function $f : [n]^d \to \mathbb{R}$ and oracle access to independent samples from distribution Π. The tester satisfies the following:*

1. *If f satisfies $\mathcal{P}$, then T accepts*[1].

2. *If f is ε-far from $\mathcal{P}$ with respect to distribution Π, then T rejects with probability at least 2/3.*

In Chapter 3, we focus on *product distributions* Π on $[n]^d$.

Definition 2.0.4 (Product distribution)**.** *A distribution Π on $[n]^d$ is called a* product distribution *if its marginal distributions are mutually independent. In other words, $\Pi = \Pi_1 \times \Pi_2 \cdots \times \Pi_d$, where Π_i is a distribution on $[n]$.*

For stating our bounds, we need the notion of *optimal binary search trees* over $[n]$, defined with respect to a distribution Π.

Definition 2.0.5 (Optimal Binary Search Trees [Knu73])**.** *Given a binary search tree T over the points in $[n]$ and a distribution Π, let $\Delta_\Pi(T)$ denote the expected depth of T with respect of Π, that is, $\Delta_\Pi(T) = \mathbf{E}_{x\sim\Pi}[\mathit{depth}(x)]$. Let $\mathcal{T}$ denote the set of all binary search trees over $[n]$. An optimal binary search tree, denoted by T_{opt} satisfies the following equation $\Delta_\Pi(T_{opt}) = \min_{T\in\mathcal{T}} \Delta_\Pi(T)$. $\Delta_\Pi(T_{opt})$ is denoted by $\Delta^*(\Pi)$.*

2.1 Properties We Consider

Next we define properties of real-valued functions considered in this article and summarize previous work on testing them. Most properties of real-valued functions studied in the property testing framework are for functions over the hypergrid domains.
We consider domains that are subsets of $[n]^d$ to be able to handle arbitrary erasures on $[n]^d$.

2.1.1 Monotonicity.

Monotonicity of functions, first studied in the context of property testing in [GGL$^+$00], is one of the most widely investigated properties in this model [EKK$^+$00, DGL$^+$99, LR01, FLN$^+$02, AC06, Fis04, HK08a, BRW05, PRR06a, ACCL07, BGJ$^+$12, BCGSM12, BBM12, CS13a, CS13b, BRY14b, CDJS15]. A function $f : \mathcal{D} \mapsto \mathbb{R}$, defined on a partially ordered domain $\mathcal{D}$ with order $\preceq$, is monotone if $x \preceq y$ implies $f(x) \leq f(y)$ for all $x, y \in \mathcal{D}$. The query complexity of testing monotonicity of functions $f : [n] \mapsto \mathbb{R}$ is $\Theta(\log n/\varepsilon)$ [EKK$^+$00, Fis04]; for functions $f : [n]^d \mapsto \mathbb{R}$, it is

---

[1] This requirement precludes the tester from making an error on Lipschitz functions, i.e., as defined, the tester must have one-sided error. More general, two-sided error testers, must accept Lipschitz functions with probability at least 2/3. Currently, the fastest known Lipschitz testers have one-sided error.

$\Theta(d \log n/\varepsilon)$ [CS13a, CS13b], and for functions over arbitrary partially ordered domains $\mathcal{D}$, it is $O\left(\sqrt{|\mathcal{D}|/\varepsilon}\right)$ [FLN+02].

2.1.2 The Lipschitz Property.

Lipschitz continuity is defined for functions between arbitrary metric spaces, but was specifically studied for real-valued functions on hypergrid domains [JR13, AJMR12, CS13a, DJRT13, BRY14b, CDJS15] because of applications to privacy [JR13, DJRT13]. For $\mathcal{D} \subseteq [n]^d$ and $c \in \mathbb{R}$, a function $f : \mathcal{D} \mapsto \mathbb{R}$ is c-Lipschitz if $|f(x) - f(y)| \leq c \cdot ||x - y||_1$ for all $x, y \in \mathcal{D}$, where $||x - y||_1$ is the L_1 distance between x and y. More generally, f is (α, β)-Lipschitz, where $\alpha < \beta$, if $\alpha \cdot ||x - y||_1 \leq |f(x) - f(y)| \leq \beta \cdot ||x - y||_1$ for all $x, y \in [n]^d$. All (α, β)-Lipschitz properties can be tested with $O(d \log n/\varepsilon)$ queries [CS13a].

2.1.3 Bounded Derivative Properties.

We defined the class of bounded derivative properties (BDPs) in [CDJS15]. This is a natural generalization of monotonicity and the Lipschitz property.

An ordered set $\mathbf{B}$ of $2d$ functions $l_1, u_1, l_2, u_2, \ldots, l_d, u_d : [n-1] \mapsto \mathbb{R} \cup \{\pm\infty\}$ is a *bounding family* if for all $r \in [d]$ and $y \in [n-1]$, $l_r(y) < u_r(y)$. Let $\mathbf{B}$ be a bounding family of functions and let $\mathbf{e}_r$ be the unit vector along dimension r. Let $\partial_r f(x) = f(x + \mathbf{e}_r) - f(x)$. The property $\mathcal{P}(\mathbf{B})$ of being $\mathbf{B}$-*derivative bounded* is the set of functions $f : [n]^d \mapsto \mathbb{R}$ such that for all $r \in [d]$ and $x \in [n]^d$ with $x_r \neq n$

$$l_r(x_r) \leq \partial_r f(x) \leq u_r(x_r). \tag{2.1}$$

The class of BDPs includes monotonicity and the c-Lipschitz property. The bounding family for monotonicity is obtained by setting $l_r(y) = 0$ and $u_r(y) = \infty$ for all $r \in [d]$, and for the c-Lipschitz property, by setting $l_r(y) = -c$ and $u_r(y) = c$ for all $r \in [d]$. In general, different bounding families allow a function to be monotone in one dimension, c-Lipschitz in another dimension and so on. In [CDJS15], we showed that the complexity of testing BDPs of functions $f : [n]^d \mapsto \mathbb{R}$ is $\Theta(d \log n/\varepsilon)$ for uniform distribution. For arbitrary product distributions, our query complexity is $O(\varepsilon^{-1} \sum_{i=1}^{d} \Delta^*(\Pi_i))$. Here $\Delta^*(\Pi)$ is as defined in Definition 2.0.5. We show that this is *optimal*.

A bounding family $\mathbf{B} = \{l_1, u_1, \ldots, l_d, u_d\}$ defines a quasi-metric $\mathfrak{m}_{\mathbf{B}}(x, y) := \sum_{r: x_r > y_r} \sum_{t=y_r}^{x_r - 1} u_r(t) - \sum_{r: x_r < y_r} \sum_{t=x_r}^{y_r - 1} l_r(t)$ over the points in $x, y \in [n]^d$. We investigate this further in Chapter 3 and use a characterization of BDPs from [CDJS15] for defining BDPs on functions over domains $\mathcal{D} \subseteq [n]^d$ in Chapter 4.

2.1.4 Convexity of Functions.

A function $f : \mathcal{D} \mapsto \mathbb{R}$ is convex if $f(t\mathbf{x} + (1-t)\mathbf{y}) \leq tf(\mathbf{x}) + (1-t)f(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{D}$ and $t \in [0, 1]$. If $\mathcal{D} \subseteq [n]$, equivalently, f is convex if $\frac{f(y)-f(x)}{y-x} \leq \frac{f(z)-f(y)}{z-y}$ for all $x < y < z$. Parnas et al. [PRR06a] gave a convexity tester for functions $f : [n] \mapsto \mathbb{R}$ with query complexity $O(\log n/\varepsilon)$. Blais et al. [BRY14b] gave an $\Omega(\log n)$ bound for nonadaptive testers for this problem.

Chapter 3

Optimal Property Testing on Product Distributions

3.1 Introduction

In this chapter, we give the testers for Bounded Derivative Properties (BDPs) of functions $f : [n]^d \to \mathbb{R}$. As noted in Chapter 2, these cover well studied properties like monotonicity and the Lipschitz property. Our testers always accept if the input function f satisfies a given BDP P. They reject with high probability if f is ε-far from satisfying P. As against most of the previous works, the farness is measured with respect to a (known or unknown) product distribution. In a previous work [DJRT13], we have shown that fast testers for the Lipschitz property of functions $f : [n]^d \to \mathbb{R}$ will in turn result in fast privacy testers. We further elaborate on this below.

Consider an algorithm $\mathcal{A}$ that works runs on a d-dimensional data-set $\mathcal{D}$, where each attribute or entry of a particular row is an element in $[n]$. We can test whether the output of $\mathcal{A}$ preserves the privacy of every individual entry of $\mathcal{D}$. The notion of privacy that we test is a distributional relaxation of well-studied notion of *Differential Privacy* [DMNS06]. We call it Differential Privacy over Typical Databases (DPTD) [DJRT13]. Since this thesis is focused on property-testing, we to urge the reader to read [DJRT13] for further details. In Section 3.1.1, we summarize our results and technical contributions in designing the optimal testers on a larger class of properties called Bounded Derivative Properties (BDP) (see Section 2.1 for the definition).

3.1.1 Main Results

Here we present the results from [CDJS15]. Our primary result is a property tester for all bounded-derivative properties over any product distribution. The formal theorem requires some definitions of search trees. Consider any binary search tree (BST) T over the universe $[n]$, and let the depth of a node denote the number of *edges* from it to the root. For a distribution $\mathcal{D}_r$ over $[n]$, the *optimal BST for* $\mathcal{D}_r$ is the BST minimizing the expected depth of vertices drawn from $\mathcal{D}_r$. Let $\Delta^*(\mathcal{D}_r)$ be this optimal depth (see Definition 2.0.5): a classic dynamic programming solution finds this optimal tree [Knu73, Yao82] in polynomial time. Given a product distribution $\mathcal{D} = \prod_{r \leq d} \mathcal{D}_r$, we abuse notation and let $\Delta^*(\mathcal{D})$ denote the sum $\sum_{r=1}^{d} \Delta^*(\mathcal{D}_r)$.

Theorem 3.1.1. [Main upper bound] *Consider functions* $f : [n]^d \mapsto \mathbb{R}$. *Let* $\mathbf{B}$ *be a bounding family and* $\mathcal{D}$ *be a product distribution. There is a tester for* $\mathcal{P}(\mathbf{B})$ *w.r.t.* $\mathcal{D}$ *making* $100\varepsilon^{-1}\Delta^*(\mathcal{D})$ *queries.*

The tester is non-adaptive with one-sided error, that is, the queries don't depend on the answers, and the tester always accepts functions satisfying the property. Furthermore, the *same* tester works for all bounding families, that is, the set of queries made by the tester doesn't depend on $\mathbf{B}$. Interestingly, the "worst" distribution is the uniform distribution, where $\Delta^*(\mathcal{D})$ is maximized to $\Theta(d \log n)$. We remark that the class of bounded derivative properties was not known to be testable even under uniform distributions. Results were known [CS13a] (only under the uniform distribution) for the subclass where all l_r (and u_r) are the same, constant function. To give perspective on the above result, it is instructive to focus on say just monotonicity (one can repeat this for Lipschitz). Let $H(\mathcal{D})$ denote the Shannon entropy of distribution $\mathcal{D}$ over the hypergrid. It is well-known that $\Delta^*(\mathcal{D}_r) \leq H(\mathcal{D}_r)$ (see [Meh75] for a proof), so $\Delta^*(\mathcal{D}) \leq H(\mathcal{D})$ for product distribution $\mathcal{D}$.

Corollary 3.1.2. *Consider functions* $f : [n]^d \mapsto \mathbb{R}$. *Monotonicity testing over a product distribution* $\mathcal{D}$ *can be done with* $100H(\mathcal{D})/\varepsilon$ *queries.*

This is an *exponential* improvement over the previous best result of Ailon and Chazelle [AC06], who give a monotonicity tester with query complexity $O(2^d H(\mathcal{D})/\varepsilon)$. Observe that for uniform distributions, $H(\mathcal{D}) = \Theta(d \log n)$, and therefore the above result subsumes the optimal testers of [CS13a]. Now consider the monotonicity testing over the boolean hypercube.

Corollary 3.1.3. *Consider functions* $f : \{0,1\}^d \mapsto \mathbb{R}$. *Monotonicity testing over any product distribution* $\mathcal{D} = \prod_{r=1}^{d} \mathcal{D}_r$*, where each* $\mathcal{D}_r = (\mu_r, 1-\mu_r)$*, can be done with* $100\varepsilon^{-1} \sum_{r=1}^{d} \min(\mu_r, 1-\mu_r)$ *queries.*

Given that monotonicity testing over the hypercube has received much attention [GGL$^+$00, DGL$^+$99, LR01, FLN$^+$02, BBM12, CS13a, CS13b], it is somewhat surprising that *nothing non-trivial* was known even over the p-biased distribution for $p \neq 1/2$; our result implies an $O(\epsilon^{-1}pd)$-query tester. The above corollary also asserts that entropy of a distribution *doesn't capture the complexity of monotonicity testing* since the entropy, $\sum_r \mu_r \log(1/\mu_r) + (1-\mu_r)\log(1/(1-\mu_r))$, can be larger than the query complexity described above by a logarithmic factor. For example, if each $\mu_r = 1/\sqrt{d}$, the tester of Corollary 3.1.3 requires $O(\sqrt{d}/\varepsilon)$ queries, while $H(\mathcal{D}) = \Theta(\sqrt{d}\log d)$. We complement Theorem 3.1.1 with a matching lower bound, cementing the connection between testing of bounded-derivative properties and optimal search tree depths. This requires a technical definition of stable distributions, which is necessary for the lower bound. To see this consider a distribution $\mathcal{D}$ for which there exists a product distribution $\mathcal{D}'$ such that $||\mathcal{D}' - \mathcal{D}||_{\mathsf{TV}} \leq \epsilon/2$ but $\Delta^*(\mathcal{D}') \ll \Delta^*(\mathcal{D})$. One could simply apply Theorem 3.1.1 with $\mathcal{D}'$ to obtain a tester with a much better query complexity than $\Delta^*(\mathcal{D})$. $\mathcal{D}$ is called (ϵ', ρ)-stable if $\|\mathcal{D} - \mathcal{D}\| \leq \epsilon'$ implies $\Delta^*(\mathcal{D}') \geq \rho\Delta^*(\mathcal{D})$, for any product distribution $\mathcal{D}'$.

Theorem 3.1.4. [Main lower bound] *For any parameter ϵ, there exists $\epsilon' = \Theta(\epsilon)$ such that for any bounding family $\mathbf{B}$ and (ε', ρ)-stable, product distribution $\mathcal{D}$, any (even adaptive, two-sided) tester for $\mathcal{P}(\mathbf{B})$ w.r.t. $\mathcal{D}$ with proximity parameter ε requires $\Omega(\rho\Delta^*(\mathcal{D}))$ queries.*

This lower bound is new even for monotonicity testing over one dimension. Ailon and Chazelle [AC06] explicitly ask for lower bounds for monotonicity testing for domain $[n]$ over arbitrary distributions. Our upper and lower bounds completely resolve this problem. For Lipschitz testing, the state of the art was a *non-adaptive* lower bound of $\Omega(d\log n)$ for the uniform distribution [BRY14b]. Since the uniform distribution is stable, the previous theorem implies an optimal $\Omega(d\log n)$ lower bound even for adaptive, two-sided testers over the uniform distribution. The previous upper bounds are in the setting where the tester knows the distribution $\mathcal{D}$. In the *distribution-free* setting, the tester only gets random samples from $\mathcal{D}$ although it is free to query any point of the domain. As a byproduct of our approach, we also get results for this setting. The previous best bound was an $O(\varepsilon^{-1}d2^d\log n)$ query tester [AC06].

Theorem 3.1.5. *Consider functions $f : [n]^d \mapsto \mathbb{R}$. There is a distribution-free (non-adaptive, one-sided) tester for $\mathcal{P}(\mathbf{B})$ w.r.t. $\mathcal{D}$ making $100\varepsilon^{-1}d\log n$ queries.*

3.1.2 Technical Highlights

Optimal dimension reduction. The main engine running the upper bounds is an optimal dimension reduction theorem. Focus on just the uniform distribution. Given $f : [n]^d \mapsto \mathbb{R}$ that is ε-far from $\mathcal{P}(\mathbf{B})$, what is the expected distance of the function restricted to a uniform random line in $[n]^d$? This natural combinatorial question has been at the heart of various monotonicity testing results [GGL+00, DGL+99, AC06, HK08a]. The best known bounds are that this expected distance is at least $\varepsilon/(d2^d)$ [AC06, HK08a]. Weaker results are known for the Lipschitz property [JR13, AJMR12]. We given an optimal resolution (up to constant factors) to this problem not only for the uniform distribution, but for any arbitrary product distribution, and for any bounded derivative property. In $[n]^d$, an r-line is a combinatorial line parallel to the r-axis. Fix some bounding family $\mathbf{B}$ and product distribution $\mathcal{D} = \prod_r \mathcal{D}_r$. Note that $\mathcal{D}_{-r} = \prod_{i \neq r} \mathcal{D}_i$ is a distribution on r-lines. If we restrict f to an r-line ℓ, we get a function $f|_\ell : [n] \mapsto \mathbb{R}$. It is meaningful to look at the distance of $f|_\ell$ to $\mathcal{P}(\mathbf{B})$ (though this only involves the bounds of $l_r, u_r \in \mathbf{B}$). Let $\mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P}(\mathbf{B})) := \mathbf{E}_{\ell \sim \mathcal{D}_{-r}}[\mathsf{dist}_{\mathcal{D}_r}(f|_\ell, \mathcal{P}(\mathbf{B}))]$.

Theorem 3.1.6. [Optimal Dimension Reduction] *Fix bounding family* $\mathbf{B}$ *and product distribution* $\mathcal{D}$. *For any function* f,

$$\sum_{r=1}^{d} \mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P}(\mathbf{B})) \geq \mathsf{dist}_{\mathcal{D}}(f, \mathcal{P}(\mathbf{B}))/4.$$

Let us give a short synopsis of previous methods used to tackle the case of monotonicity in the uniform distribution case. For brevity's sake, let ϵ^r_f denote $\mathsf{dist}^r_{\mathcal{U}}(f, \mathtt{MON})$ and ϵ_f denote $\mathsf{dist}_{\mathcal{U}}(f, \mathtt{MON})$. That is, $\epsilon^r_f n^d$ modifications makes the function monotone along the r-dimension, and the theorem above states that $4\sum_r \epsilon^r_f n^d$ modifications suffice to make the whole function monotone. Either explicitly or implicitly, previous attempts have taken a constructive approach: they use the modifications along the rth dimensions to correct the whole function. Although in principle a good idea, a bottleneck to the above approach is that correcting the function along one dimension may potentially introduce significantly larger errors along other dimensions. Thus, one can't just "add up" the corrections in a naive manner. The process is even more daunting when one tries this approach for the Lipschitz property. Our approach is completely different, and is 'non-constructive', and looks at all bounded-derivative properties in a uniform manner. We begin by proving Theorem 3.4.1 for $\mathcal{P}(\mathbf{B})$ over the uniform distribution. The starting point is to consider a weighted violation graph G, where any two domains point forming a violation to $\mathcal{P}(\mathbf{B})$ are connected (the weight is a "magnitude" of violation). It is well-known that the size of a maximum matching M in G is at least $\varepsilon_f n^d/2$. The main insight is to use different matchings to get handles on the distance ϵ^r_f rather than using modifications that correct the function. More precisely, we construct a sequence of special matchings $M = M_0, M_1, \ldots, M_d = \emptyset$, such that the drop in size $|M_{r-1}| - |M_r|$ is at most $2\epsilon^r_f n^d$, which proves the above theorem. This requires structural properties on the M_r's proven using the *alternating path machinery* developed in [CS13a]. What about a general product distribution $\mathcal{D}$? Suppose we 'stretch' every point in every direction proportional to its marginal. This leads to a 'bloated' hypergrid $[N]^d$ where each point in the original hypergrid corresponds to a high-dimensional cuboid. By the obvious association of function values, one obtains a $f_{\texttt{ext}} : [N]^d \mapsto \mathbb{R}$. If $\mathcal{P}(\mathbf{B})$ is monotonicity, then it is not hard to show that $\mathsf{dist}_{\mathcal{D}}(f) = \mathsf{dist}_{\mathcal{U}}(f_{\texttt{ext}})$. So we can apply dimension reduction for $f_{\texttt{ext}}$ over the uniform distribution and map it back to f over $\mathcal{D}$. However, such an argument breaks down for Lipschitz (let alone general $\mathbf{B}$), since $\mathsf{dist}_{\mathcal{U}}(f')$ can be much smaller than $\mathsf{dist}_{\mathcal{D}}(f)$. The optimal fix for $f_{\texttt{ext}}$ could perform non-trivial changes within the cuboidal regions, and this cannot be mapped back to a fix for the original f. This is where the generality of the bounded-derivative properties saves the day. For any $\mathbf{B}$ and $\mathcal{D}$, we can define a new bounding family $\mathbf{B}_{\texttt{ext}}$ over $[N]^d$, such that $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{P}(\mathbf{B})) = \mathsf{dist}_{\mathcal{U}}(f_{\texttt{ext}}, \mathcal{P}(\mathbf{B}_{\texttt{ext}}))$. Now, dimension reduction is applied to $f_{\texttt{ext}}$ for $\mathcal{P}(\mathbf{B}_{\texttt{ext}})$ over $\mathcal{U}$ and translated back to the original setting.

3.1.3 Search Trees and Monotonicity

An appealing aspect of our results is the tight connection between optimal search trees over product distributions to bounded-derivative properties. The dimension reduction lemma allows us (for the upper bounds) to focus on just the line domain $[n]$. For monotonicity testing on $[n]$ over an arbitrary distribution $\mathcal{D}$, Halevy and Kushilevitz gave an $O(\varepsilon^{-1} \log n)$-query distribution free tester [HK08a], and Ailon and Chazelle gave an $O(\varepsilon^{-1} H(\mathcal{D}))$-query tester [AC06]. Pretty much every single result for monotonicity testing on $[n]$ involves some analogue of binary search [EKK$^+$00, BRW05, ACCL07, PRR06b, HK08a, AC06, BGJ$^+$12]. But we make this connection extremely precise. We show that *any* binary search tree can be used to get a tester with respect to an arbitrary distribution, whose expected query complexity is the expected depth of the tree with respect to the distribution. This argument is extremely simple in hindsight, but it is a significant conceptual insight. Firstly, it greatly simplifies earlier results – using the completely balanced BST, we get an $O(\varepsilon^{-1} \log n)$-distribution free tester; with the optimal BST, we get $O(\varepsilon^{-1} H(\mathcal{D}))$-queries. The BST tester along with the dimension reduction, provides a tester for $[n]^d$ whose running time can be better than $H(\mathcal{D})$ (especially for the hypercube). Most importantly, optimal BSTs are a crucial ingredient for our lower bound construction.

3.1.4 Lower Bounds for Product Distributions.

The first step to general lower bounds is a simple reduction from monotonicity testing to any bounded-derivative property. Again, the reduction may seem trivial in hindsight, but note that special sophisticated constructions were used for existing Lipschitz lower bounds [JR13, BRY14b]. For monotonicity, we use the framework developed in [Fis04, CS13b] that allows us to focus on comparison based testers. The lower bound for $[n]$ uses a convenient near-optimal BST. For each level of this tree we construct a 'hard' non-monotone function, leading to (roughly) $\Delta^*(\mathcal{D})$ such functions in case of stable distributions. These functions have violations to monotonicity lying in 'different regions' of the line, and any bonafide tester must make a different query to catch each function. In going to higher dimensions, we face a significant technical hurdle. The line lower bound easily generalizes to the hypergrid *if each marginal distribution is individually stable.* However, this may not be the case – there are stable product distributions whose marginals are unstable. As a result, each dimension may give 'hard' functions with very small distance. Our main technical contribution is to show how to *aggregate* functions from various dimensions together to obtain hard functions for the hypergrid in such a way that the distances add up. This is rather delicate, and is perhaps the most technical portion of this paper. In summary, we show that for stable distributions, the total search-tree depth is indeed the lower bound for testing monotonicity, and via the reduction mentioned above, for any bounded-derivative property.

3.1.5 Organization.

We give a brief outline of remainder of this chapter. In Section 3.2, we define a particular *quasi-metric* corresponding to a bounding family $\mathbf{B}$ and give an equivalent definition of the bounded-derivative property with respect to it. This definition is convenient and will be the one used for the rest of the paper. In Section 3.3, we describe the tester when the domain is just the line, that uses optimal binary search tree to get the optimal query-complexity. The dimension reduction theorem is presented in its full glory in Section 3.4. The tester for hypergrid is an easy generalization of the line tester via dimension reduction as presented in Section 3.6. For lower bounds, we prove the reduction to monotonicity in Section 3.7.1, and describe the approach to montonicity lower bounds in Section 3.7.2. The hard families for the line is given in Section 3.7.3, for the hypercube in Section 3.7.4, and the general hypergrid lowerbound is described in Section 3.7.5.

3.2 Quasimetric Induced by a Bounding Family

It is convenient to abstract out $\mathcal{P}(\mathbf{B})$ in terms of a *metric-bounded property*. Such ideas was used in [CS13a] to give a unified proof for monotonicity and Lipschitz for the uniform distribution. The treatment here is much more general. We define a quasimetric depending on $\mathbf{B}$ denoted by $\mathfrak{m}(x, y)$.

Definition 3.2.1. *Given bounding family* $\mathbf{B}$*, construct the weighted directed hypergrid* $[n]^d$*, where all adjacent pairs are connected by two edges in opposite directions. The weight of* $(x + \mathbf{e}_r, x)$ *is*

$u_r(x_r)$ *and the weight of* $(x, x + \mathbf{e}_r)$ *is* $-l_r(x_r)$. $\mathfrak{m}(x, y)$ *is the shortest path weight from* x *to* y.

Note that $\mathfrak{m}$ is asymmetric, can take negative values, and $\mathfrak{m}(x, y) = 0$ does not necessarily imply $x = y$. For these reasons, it is really a possibly-negative-pseudo-quasi-metric, although we will refer to it simply as a metric in the remainder of the paper. Since $\mathbf{B}$ is a bounding family, any cycle in the $[n]^d$ digraph has positive weight, and $\mathfrak{m}(x, y)$ is well-defined. Therefore, a shortest path from x to y is given by the rectilinear path obtained by decreasing the coordinates r with $x_r > y_r$ and increasing the coordinates r with $x_r < y_r$. A simple calculation yields

$$\mathfrak{m}(x, y) := \sum_{r: x_r > y_r} \sum_{t=y_r}^{x_r - 1} u_r(t) - \sum_{r: x_r < y_r} \sum_{t=x_r}^{y_r - 1} l_r(t) \tag{3.1}$$

If a function $f \in \mathcal{P}(\mathbf{B})$, then applying Eq. (2.1) on every edge of the path described above (the upper bound when we decrement a coordinate and the lower bound when we increment a coordinate), we get $f(x) - f(y) \leq \mathfrak{m}(x, y)$ for any pair (x, y). Conversely, if $\forall x, y, f(x) - f(y) \leq \mathfrak{m}(x, y)$, then considering neighboring pairs gives $f \in \mathcal{P}(\mathbf{B})$. This argument is encapsulated in the following lemma.

Lemma 3.2.2. $f \in \mathcal{P}(\mathbf{B})$ *iff* $\forall x, y \in [n]^d$, $f(x) - f(y) \leq \mathfrak{m}(x, y)$.

When $\mathcal{P}(\mathbf{B})$ is monotonicity, $\mathfrak{m}(x, y) = 0$ if $x \prec y$ and ∞ otherwise. For the c-Lipschitz property, $\mathfrak{m}(x, y) = c\|x - y\|_1$. The salient properties of $\mathfrak{m}(x, y)$ are documented below and can be easily checked.

Lemma 3.2.3. $\mathfrak{m}(x, y)$ *satisfies the following properties.*

1. (Triangle Inequality.) *For any* x, y, z, $\mathfrak{m}(x, z) \leq \mathfrak{m}(x, y) + \mathfrak{m}(y, z)$.
2. (Linearity.) *If* x, y, z *are such that for every* $1 \leq r \leq d$, *either* $x_r \leq y_r \leq z_r$ *or* $x_r \geq y_r \geq z_r$, *then* $\mathfrak{m}(x, z) = \mathfrak{m}(x, y) + \mathfrak{m}(y, z)$.
3. (Projection.) *Fix any dimension* r. *Let* x, y *be two points with* $x_r = y_r$. *Let* x' *and* y' *be the projection of* x, y *onto some other* r*-hyperplane. That is,* $x'_r = y'_r$, *and* $x'_j = x_j$, $y'_j = y_j$ *for* $j \neq r$. *Then,* $\mathfrak{m}(x, y) = \mathfrak{m}(x', y')$ *and* $\mathfrak{m}(x, x') = \mathfrak{m}(y, y')$.

Proof. $\mathfrak{m}(x, x) = 0$ follows since the RHS of Eq. (3.1) is empty. Triangle inequality holds because $\mathfrak{m}(x, y)$ is a shortest path weight. Linearity follows by noting $\sum_{t=y_r}^{x_r-1} u_r(t) = \sum_{t=y_r}^{z_r-1} u_r(t) + \sum_{t=z_r}^{x_r-1} u_r(t)$. For projection, note that if $x_r = y_r$, the RHS of Eq. (3.1) has no term corresponding to r. Thus, $\mathfrak{m}(x, y) = \mathfrak{m}(x', y')$. Suppose $x'_r > x_r$. Then, $\mathfrak{m}(x, x') = \sum_{t=x_r}^{x'_r} u_i(t) = \mathfrak{m}(y, y')$. A similar proof holds when $x'_r < x_r$. □

Henceforth, all we need is Lemma 3.2.2 and Lemma 3.2.3. We will interchangably use the terms $\mathcal{P}(\mathbf{B})$ and $\mathcal{P}(\mathfrak{m})$ where $\mathfrak{m}$ is as defined in Eq. (3.1). In fact, since $\mathbf{B}$ and therefore $\mathfrak{m}$ will be fixed in most of our discussion, we will simply use $\mathcal{P}$ including the parametrization wherever necessary.

Definition 3.2.4 (Violation Graph). *The violation graph of a function f with respect to property $\mathcal{P}$, denoted as $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$, has $[n]^d$ as vertices, and edge (x, y) if it forms a violation to $\mathcal{P}$, that is either $f(x) - f(y) > \mathfrak{m}(x, y)$ or $f(y) - f(x) > \mathfrak{m}(y, x)$.*

The triangle inequality of $\mathfrak{m}$ suffices to prove the following version of a classic lemma [FLN$^+$02] relating the distance of a function to $\mathcal{P}$ to the vertex cover of the violation graph.

Lemma 3.2.5. *For any distribution $\mathcal{D}$ on $[n]^d$, any bounded-derivative property $\mathcal{P}$, and any function f, $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{P}) = \min_X \mu_{\mathcal{D}}(X)$ where the minimum is over all vertex covers of $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$. Thus, if M is* any *maximal matching in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$, then for the* uniform distribution, $|M| \geq \mathsf{dist}_{\mathcal{U}}(f, \mathcal{P}) n^d / 2$.

3.3 Testers For the Line $[n]$

Let T be *any* binary search tree (BST) with respect to the totally ordered domain $[n]$. Every node of T is labeled with a unique entry in $[n]$, and the left (resp. right) child, if it exists, has a smaller (resp. larger) entry. The *depth* of a node v in the tree T, denoted as $\mathsf{depth}_T(v)$, is the number of *edges* on its path to the root. So the root has depth 0. Given a distribution $\mathcal{D}$ on $[n]$, the expected depth of T w.r.t. $\mathcal{D}$ is denoted as $\Delta(T; \mathcal{D}) = \mathbf{E}_{v \sim \mathcal{D}}[\mathsf{depth}_T(v)]$. The depth of the optimal BST w.r.t. $\mathcal{D}$ is denoted by $\Delta^*(\mathcal{D})$. It has long been observed that the transitivity of violations is the key property required for monotonicity testing on $[n]$ [BRW05, EKK$^+$00, ACCL07, JR13]. We distill this argument down to a key insight: Given *any* BST T, there exists the following tester BST(T) for $\mathcal{P}$ on the line.

It is clear that the tester never rejects a function satisfying $\mathcal{P}$. (To connect with previous work, observe that the list of ancestor-descendant pairs forms a 2-Transitive Closure spanner [BGJ$^+$12].)

Lemma 3.3.1. *For any bounded derivative property $\mathcal{P}$,* $\Pr[$*BST tester rejects*$] \geq \mathsf{dist}_{\mathcal{D}}(f, \mathcal{P})$.

Proof. Let X be the set of non-root nodes v of T with the following property: (u, v) is a violation to $\mathcal{P}$ for some node u on the path from v to the root of T. The probability of rejection of the BST tester is precisely $\mu_{\mathcal{D}}(X)$. We claim that X is a vertex cover of $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$ which proves the lemma using Lemma 3.2.5. Pick any violation (x, y) and assume without loss of generality $f(x) - f(y) > \mathfrak{m}(x, y)$. Let z be the lowest common ancestor of x and y in T. By the BST property, either $x < z < y$ of $x > z > y$. By the linearity property of $\mathfrak{m}$, we get $\mathfrak{m}(x,y) = \mathfrak{m}(x,z) + \mathfrak{m}(z,y)$. This implies either $f(x) - f(z) > \mathfrak{m}(x,z)$ or $f(z) - f(y) > \mathfrak{m}(z,y)$, that is, either (x,z) or (y,z) is a violation implying one of them is in X. ☐

Lemma 3.3.2. *For any BST T, there is a $24\varepsilon^{-1}\Delta(T;\mathcal{D})$-query line monotonicity-tester.*

Proof. The expected number of queries made by the BST tester is $\sum_{v\text{: non-root}} \Pr[v] \cdot (\mathsf{depth}_T(v) + 1) = (1 - \Pr[\text{root}]) + \Delta(T;\mathcal{D}) \leq 2 \cdot \Delta(T;\mathcal{D})$.

$$\sum_{v\text{: non-root}} \Pr[v] \cdot (\mathsf{depth}_T(v) + 1) = (1 - \Pr[\text{root}]) + \Delta(T;\mathcal{D}) \leq 2 \cdot \Delta(T;\mathcal{D})$$

The expected depth is at least $(1 - \Pr[\text{root}])$ since non-roots have depth at least 1. To get a bonafide tester with deterministic query bounds, run the BST tester $2/\varepsilon$ times, aborting (and accepting) if the total number of queries exceeds $24\Delta(T;\mathcal{D})/\varepsilon$. The expected total number of queries is at most $4\Delta(T;\mathcal{D})/\varepsilon$. By Markov's inequality, the probability that the tester aborts is $\leq 1/6$. By Lemma 3.3.1, if $\mathsf{dist}_{\mathcal{D}}(f;\mathcal{P}) > \varepsilon$, the probability that this tester does not find a violation is at most $(1-\varepsilon)^{2/\varepsilon} \leq 1/6$. With probability $\geq (1-1/6-1/6) = 2/3$, the tester rejects an ε-far function. ☐

Choose T to be the optimal BST to get the following theorem.

Theorem 3.3.3. *There exists a $24\varepsilon^{-1}\Delta^*(\mathcal{D})$-query tester for any bounded derivative property over the line.*

Note that once the tree is fixed, the BST tester only needs random samples from the distribution. Pick T to be the balanced binary tree of depth $O(\log n)$ to get a *distribution-free* tester.

Theorem 3.3.4. *There exists a $24\varepsilon^{-1}\log n$-query* distribution free *tester for any bounded-derivative property over the line.*

3.4 The Dimension Reduction Theorem

For any combinatorial line ℓ in $[n]^d$, $f|_\ell : [n] \mapsto \mathbb{R}$ is f restricted to ℓ. It is natural to talk of $\mathcal{P}$ for any restriction of f, so $\mathsf{dist}_{\mathcal{D}_r}(f_{|\ell}, \mathcal{P})$ is well-defined for any r-line ℓ. For any $1 \leq r \leq d$, define the r-distance of the function:

$$\mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P}) := \mathbf{E}_{\ell \sim \mathcal{D}_{-r}}[\mathsf{dist}_{\mathcal{D}_r}(f_{|\ell}, \mathcal{P})] \tag{3.2}$$

Call a function f r-good if there are no violations along r-lines, that is, for any x and y on the same r-line, we have $f(x) - f(y) \leq \mathfrak{m}(x, y)$. Observe that $\mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P})$ is the minimum $\mu_{\mathcal{D}}$-mass of points on which f needs to be modified to make it r-good. The following is the optimal dimension reduction theorem which connects the r-distances to the real distance.

Theorem 3.4.1 (Dimension Reduction)**.** *For any function f, any bounded-derivative property $\mathcal{P}$, and any product distribution $\mathcal{D} = \prod_{1 \leq r \leq d} \mathcal{D}_i$,*

$$\sum_{r=1}^{d} \mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P}) \geq \mathsf{dist}_{\mathcal{D}}(f, \mathcal{P})/4.$$

(It can be easily shown that $\sum_{r=1}^{d} \mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P}) \leq \mathsf{dist}_{\mathcal{D}}(f, \mathcal{P})$, by simply putting the same 1D function of all, say, 1-lines.) We first prove the above theorem for the uniform distribution. Recall the violation graph $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$ whose edges are violation to $\mathcal{P}$. We define weights on the edges (x, y).

$$w(x, y) := \mathsf{max}(f(x) - f(y) - \mathfrak{m}(x, y), f(y) - f(x) - \mathfrak{m}(y, x)) \tag{3.3}$$

Note that $w(x, y) > 0$ for all edges in the violation graph. Let M be a maximum weight matching of minimum cardinality (MWmC). (Introduce an arbitrary tie-breaking rule to ensure this is unique.) A pair $(x, y) \in M$ is an r-*cross pair* if $x_r \neq y_r$. The following theorem (proof defered to Section 3.4.1) establishes the crucial structural result about these MWmC matchings in violated graphs of r-good functions.

Theorem 3.4.2 (No r-violations $\Rightarrow$ no r-cross pairs)**.** *Let f be an r-good function. Then there exists an MWmC matching M in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$ with no r-cross pairs.*

We proceed with the proof of Theorem 3.4.1 over the uniform distribution starting with some definitions.

Definition 3.4.3 (*Hypergrid slices*)**.** *Given an r-dimensional vector $\mathbf{a} \in [n]^r$, the a-*slice *is $S_{\mathbf{a}} := \{x \in [n]^d : x_j = \mathbf{a}_j,\ 1 \leq j \leq r\}$.*

Each $\mathbf{a}$-slice is a $(d - r)$-dimensional hypergrid, and the various $\mathbf{a}$-slices for $\mathbf{a} \in [n]^r$ partition $[n]^d$. Let $f_{|\mathbf{a}}$ denote the restriction of f to the slice $S_{\mathbf{a}}$. For two functions f, g we use $\Delta(f, g) := |\{x : f(x) \neq g(x)\}| = \mathsf{dist}_{\mathcal{U}}(f, g) \cdot n^d$. The following claim relates the sizes of MWmC matchings to $\Delta(f, g)$.

Claim 3.4.4. *Let $f, g : [n]^d \mapsto \mathbb{R}$. Let M and N be the MWmC matchings in the violation graphs for f and g, respectively. Then, $||M| - |N|| \leq \Delta(f, g)$.*

Proof. The symmetric difference of M and N is a collection of alternating paths and cycles. $||M| - |N||$ is at most the number of alternating paths. Each alternating path must contain a point at which f and g differ, for otherwise we can improve either M or N, either in weight or cardinality. $\square$

Define a sequence of $d+1$ matchings $(M_0, M_1, \ldots, M_d)$ in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$ in non-increasing order of cardinality as follows. For $0 \leq r \leq d$, M_r is the MWmC matching in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$ among matchings that *do not contain any i-cross pairs for* $1 \leq i \leq r$. By Lemma 3.2.5, we have $|M_0| \geq \mathsf{dist}_{\mathcal{U}}(f, \mathcal{P}) n^d/2$. The last matching M_d is empty and thus has cardinality 0.

Lemma 3.4.5. *For all* $1 \leq r \leq d$, *we have* $|M_{r-1}| - |M_r| \leq 2 \cdot \mathsf{dist}^r_{\mathcal{U}}(f, \mathcal{P}) \cdot n^d$.

Adding the inequalities in the statement of Lemma 3.4.5 for all r, we get $\mathsf{dist}_{\mathcal{U}}(f, \mathcal{P}) n^d/2 \leq |M_0| - |M_d| \leq 2 \sum_{r=1}^{d} \mathsf{dist}^r_{\mathcal{U}}(f, \mathcal{P}) \cdot n^d$. This completes the proof of Theorem 3.4.1 for the uniform distribution. Now we prove Lemma 3.4.5.

Proof. Since M_{r-1} has no j-cross pairs for $1 \leq j \leq r-1$, all pairs of M_{r-1} have both endpoints in the same slice $S_{\mathbf{a}}$ for some $\mathbf{a} \in [n]^{r-1}$. Thus, M_{r-1} partitions into sub-matchings in each $S_{\mathbf{a}}$. Let $M^{\mathbf{a}}_{r-1}$ be the pairs of M_{r-1} with both endpoints in slice $S_{\mathbf{a}}$, so $|M_{r-1}| = \sum_{\mathbf{a} \in [n]^{r-1}} |M^{\mathbf{a}}_{r-1}|$. Similarly, $M^{\mathbf{a}}_r$ is defined. Since M_r has no r-cross pairs either, $\forall \mathbf{a} \in [n]^{r-1}$, $|M^{\mathbf{a}}_r| = \sum_{i=1}^{n} |M^{(\mathbf{a} \circ i)}_r|$, where $(\mathbf{a} \circ i)$ is the r-dimensional vector obtained by concatenating i to the end of $\mathbf{a}$. Observe that for any $\mathbf{a} \in [n]^{r-1}$, $M^{\mathbf{a}}_{r-1}$ is an MWmC matching in $S_{\mathbf{a}}$ w.r.t. $f_{|\mathbf{a}}$. Furthermore, for any $i \in [n]$, $M^{(\mathbf{a} \circ i)}_r$ is an MWmC matching in $S_{(\mathbf{a} \circ i)}$ w.r.t. $f_{|(\mathbf{a} \circ i)}$. Let $f^{(r)}$ be the closest function to f with no violations along dimension r. By definition, $\Delta(f, f^{(r)}) = \mathsf{dist}^r(f, \mathcal{P}) \cdot n^d$. Now comes the crucial part of the proof. Fix $\mathbf{a} \in [n]^{r-1}$ and focus on the $\mathbf{a}$-slice $S_{\mathbf{a}}$. Since $f^{(r)}$ has no violations along the r-lines, neither does ${f^{(r)}}_{|\mathbf{a}}$. By Theorem 3.4.2, there exists an MWmC matching $N^{\mathbf{a}}$ in $S_{\mathbf{a}}$ w.r.t. ${f^{(r)}}_{|\mathbf{a}}$ which has no r-cross pairs. Therefore, $N^{\mathbf{a}}$ partitions as $N^{\mathbf{a}} = \bigcup_{i=1}^{n} N^{(\mathbf{a} \circ i)}$. Furthermore, each matching $N^{(\mathbf{a} \circ i)}$ is an MWmC matching in $S_{(\mathbf{a} \circ i)}$ with respect to the weights corresponding to the function $f^{(r)}_{|(\mathbf{a} \circ i)}$. Since $M^{\mathbf{a}}_{r-1}$ is an MWmC matching w.r.t. $f_{|\mathbf{a}}$ and $N^{\mathbf{a}}$ is an MWmC matching w.r.t. $f^{(r)}_{|\mathbf{a}}$ in $S_{\mathbf{a}}$, Claim 3.4.4 gives

$$|N^{\mathbf{a}}| \geq |M^{\mathbf{a}}_{r-1}| - \Delta(f_{|\mathbf{a}}, {f^{(r)}}_{|\mathbf{a}}) \tag{3.4}$$

Since $M^{(\mathbf{a} \circ i)}_r$ is an MWmC matching w.r.t. $f_{|(\mathbf{a} \circ i)}$ and $N^{(\mathbf{a} \circ j)}$ is an MWmC matching w.r.t. $f^{(r)}_{|(\mathbf{a} \circ i)}$ in $S_{(\mathbf{a} \circ i)}$, Claim 3.4.4 gives us $|M^{(\mathbf{a} \circ i)}_r| \geq |N^{(\mathbf{a} \circ i)}| - \Delta(f_{|(\mathbf{a} \circ i)}, {f^{(r)}}_{|(\mathbf{a} \circ i)})$. Summing over all $1 \leq i \leq n$,

$$|M^{\mathbf{a}}_r| \geq |N^{\mathbf{a}}| - \Delta(f_{|\mathbf{a}}, {f^{(r)}}_{|\mathbf{a}}) \tag{3.5}$$

Adding Eq. (3.4), Eq. (3.5) over all $\mathbf{a} \in [n]^{r-1}$, $|M_r| \geq |M_{r-1}| - 2\sum_{\mathbf{a}\in[n]^{r-1}} \Delta(f_{|\mathbf{a}}, f^{(r)}{}_{|\mathbf{a}})$ $= |M_{r-1}| - 2 \cdot \mathsf{dist}^r(f, \mathcal{P}) \cdot n^d$. ☐

3.4.1 No r-Violations Imply No r-Cross Pairs

In this subsection we prove Theorem 3.4.2.

Theorem 3.4.2 (No r-violations $\Rightarrow$ no r-cross pairs)**.** *Let f be an r-good function. Then there exists an MWmC matching M in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P})$ with no r-cross pairs.*

This requires the alternating path setup of [CS13a]. Recall the weight function $w(x, y) = \max(f(x) - f(y) - \mathfrak{m}(x, y), f(y) - f(x) - \mathfrak{m}(y, x))$ defined on pairs of the domain. Note that (x, y) is a violation iff $w(x, y) > 0$. Let M be a maximum weight minimum cardinality (MWmC) matching of $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P}(\mathfrak{m}))$ with the *minimum number* of r-cross pairs. Recall an r-cross pair (x, y) has $x_r \neq y_r$. We will prove that this minimum value is 0. Let $\mathsf{cr}(M)$ be the set of r-cross pairs in M. Let $\mathsf{st}(M) := M \setminus \mathsf{cr}(M)$. For contradiction's sake, assume $\mathsf{cr}(M)$ is nonempty. Let $(x, y) \in \mathsf{cr}(M)$ be an arbitrary r-cross pair with $x_r = a$ and $y_r = b$ with $a \neq b$. Define matching $H := \{(u, v) : u_r = a, v_r = b, u_j = v_j, j \neq i\}$. This is a matching by projection between points with rth coordinate a and b. For convenience, we denote the points with rth coordinate a (resp. b) as the *a-plane* (resp. b-plane). Consider the alternating paths and cycles in $H\Delta$ $\mathsf{st}(M)$. The vertex y is incident to only an H-pair, since $(x, y) \in \mathsf{cr}(M)$. Let $y = s_1, s_2, \ldots, s_t$ be the alternating path starting from y, collectively denoted by S. We let $s_0 := x$. The end of S, s_t, may be either M-unmatched or $\mathsf{cr}(M)$-matched. In the latter case, we define s_{t+1} to be such that $(s_t, s_{t+1}) \in \mathsf{cr}(M)$. For even i, (s_{i-1}, s_i) is an H-pair and (s_i, s_{i+1}) is an M-pair. We list out some basic claims about S.

Claim 3.4.6. *If strictly positive $j \equiv 0, 1 \mod 4$, then s_j is in the b-plane. Otherwise, s_j is in the a-plane.*

Claim 3.4.7. *For strictly positive even i, $f(s_{i-1}) - f(s_i) - \mathfrak{m}(s_{i-1}, s_i) \leq 0$ and $f(s_i) - f(s_{i-1}) - \mathfrak{m}(s_i, s_{i-1}) \leq 0$.*

Proof. Since f is r-good and H-pairs differ only in the rth coordinate, $w(s_{i-1}, s_i) \leq 0$ for all even i. The definition of $w(s_{i-1}, s_i)$ completes the proof. ☐

Claim 3.4.8. *For strictly positive $i \equiv 0 \mod 4$, $\mathfrak{m}(s_{i-1}, s_i) = \mathfrak{m}(s_2, s_1)$. For $i \equiv 2 \mod 4$, $\mathfrak{m}(s_i, s_{i-1}) = \mathfrak{m}(s_2, s_1)$.*

Proof. The point s_0 (which is x) lies in the a-plane. Hence, for any $i \equiv 2 \mod 4$, s_i lies in the b-plane. Similarly, for $i \equiv 0 \mod 4$, s_i lies in the a-plan. For strictly positive even i, (s_{i-1}, s_i) is an H-pair. An application of the projection property completes the proof. ☐

Claim 3.4.9. *For strictly positive even* i, $\mathfrak{m}(s_i, s_{i+1}) = \mathfrak{m}(s_{i-1}, s_{i+2})$ *and* $\mathfrak{m}(s_{i+1}, s_i) = \mathfrak{m}(s_{i+2}, s_{i-1})$.

Proof. Consider $\mathsf{st}(M)$-pair (s_i, s_{i+1}). Both points are on the same (a or b-)plane. Observe that s_{i-1} is the projection of s_i and s_{i+2} is the projection of s_{i+1} onto the other plane. Apply the projection property of d to complete the proof. ☐

Now we have all the ingredients to prove the theorem. The strategy is to find another matching M' such that either $w(M') > w(M)$ or $w(M') = w(M)$ and M' has strictly fewer cross pairs. Let us identify certain subsets of pairs to this end. For even k, define

$$E_-(k) := (s_0, s_1), (s_2, s_3), \ldots, (s_k, s_{k+1}) = \{(s_j, s_{j+1}) : j \text{ even}, 0 \le j \le k\}$$

These are precisely the $\mathsf{st}(M)$-pairs in S in the first k-steps. Note that $|E_-(k)| = k/2 + 1$. Now we define $E_+(k)$. In English: first pick pair (s_0, s_2); subsequently pick the first unpaired s_i and pair it with the next unpaired s_j of the *opposite* parity. More precisely, for even k,

$$E_+(k) := (s_0, s_2), (s_1, s_4), (s_3, s_6), \ldots, (s_{k-3}, s_k) = (s_0, s_2) \cup \{(s_{j-3}, s_j) : j \text{ even}, 4 \le j \le k\}$$

Note that $|E_+(k)| = k/2$. Wlog, assume that $w(x, y) = f(x) - f(y) - \mathfrak{m}(x, y)$. It turns out the weights of all other M-pairs in S are determined. We will assert that the pattern is as follows.

$$w(s_i, s_{i+1}) = \begin{cases} f(s_i) - f(s_{i+1}) - \mathfrak{m}(s_i, s_{i+1}) & \text{if } i \equiv 0 \mod 4 \\ f(s_{i+1}) - f(s_i) - \mathfrak{m}(s_{i+1}, s_i) & \text{if } i \equiv 2 \mod 4 \end{cases} \qquad (\clubsuit)$$

The following lemma determines the weights of all other M-edges in the alternating path S. Recall $(s_i, s_{i+1}) \in \mathsf{st}(M)$ for even i.

Lemma 3.4.10. *Suppose* s_i *exists. If Eq. (♣) holds for all even indices* $< i$, *then* s_i *is matched in* M.

Proof. Assume $i \equiv 2 \mod 4$. (The other case is analogous and omitted.) We prove by contradiction, so suppose s_i is not matched in M. We set $M' := M - E_-(i-2) + E_+(i)$. Note that M' is a valid matching, since s_i is not matched. We compare $w(M')$ and $w(M)$. By Eq. (♣), we can express $w(E_-(i-2))$ exactly.

$$\begin{aligned} w(E_-(i-2)) &= \sum_{j\text{:even},\, 0\leq j\leq i-2} w(s_j, s_{j+1}) \\ &= [f(s_0) - f(s_1) - \mathfrak{m}(s_0, s_1)] + [f(s_3) - f(s_2) - \mathfrak{m}(s_3, s_2)] + \\ &\quad [f(s_4) - f(s_5) - \mathfrak{m}(s_4, s_5)] + [f(s_7) - f(s_6) - \mathfrak{m}(s_7, s_6)] + \cdots \\ &\quad [f(s_{i-2}) - f(s_{i-1}) - \mathfrak{m}(s_{i-2}, s_{i-1})] \end{aligned} \tag{3.6}$$

We lower bound $w(E_+(i))$. Since each individual weight term is a maximum of two expressions, we can choose either. We set the expression up to match $w(E_-(i-2))$ as best as possible.

$$\begin{aligned} w(E_+(i)) &\geq [f(s_0) - f(s_2) - \mathfrak{m}(s_0, s_2)] + [f(s_4) - f(s_1) - \mathfrak{m}(s_4, s_1)] + \\ &\quad [f(s_3) - f(s_6) - \mathfrak{m}(s_3, s_6)] + [f(s_8) - f(s_5) - \mathfrak{m}(s_8, s_5)] + \\ &\quad [f(s_{i-3}) - f(s_i) - \mathfrak{m}(s_{i-3}, s_i)] \end{aligned} \tag{3.7}$$

Note that $w(M') - w(M) = w(E_+(i)) - w(E_-(i-2))$. Observe that any f term that occurs in both Eq. (3.6) and Eq. (3.7) has the same coefficient. By Claim 3.4.9, $\mathfrak{m}(s_3, s_2) = \mathfrak{m}(s_4, s_1)$, $\mathfrak{m}(s_4, s_5) = \mathfrak{m}(s_3, s_6)$, etc.

$$w(E_+(i)) - w(E_-(i-2)) \geq f(s_{i-1}) - f(s_i) - \mathfrak{m}(s_0, s_2) + \mathfrak{m}(s_0, s_1)$$

The points s_0 and s_1 lie is different planes, and $(s_1, s_2) \in H$. We can apply the linearity property to get $\mathfrak{m}(s_0, s_1) = \mathfrak{m}(s_0, s_2) + \mathfrak{m}(s_2, s_1)$. Plugging this in, applying Claim 3.4.7 and Claim 3.4.8 for i,

$$w(E_+(i)) - w(E_-(i-2)) \geq f(s_{i-1}) - f(s_i) + \mathfrak{m}(s_2, s_1) = -[f(s_i) - f(s_{i-1}) - \mathfrak{m}(s_i, s_{i-1})] \geq 0$$

Hence $w(M') \geq w(M)$. Note that $|M'| - |M| = |E_+(i)| - |E_-(i-2)| = i/2 - ((i-2)/2 + 1) = 0$. Finally, observe that $E_+(i)$ has no r-cross pairs, but $E_-(i-2)$ has one (pair (s_0, s_1)). This contradicts the choice of M as a MWmC matching with the least r-cross pairs. $\square$

Claim 3.4.11. *If Eq. (♣) holds for all even indices $< i$, then $s_0, s_1, \ldots, s_{i+1}$ are all distinct.*

Proof. (This is trivial if $i < t$. The non-trivial case if when S ends as s_i.) The points $s_1, \ldots, s_i$ are all distinct. If $s_i \neq x$, the claim holds. So assume $s_i = x = s_0$. By Claim 3.4.6, $i \equiv 2 \mod 4$. Replace pairs $A = \{(s_0, s_1), (s_{i-2}, s_{i-1})\}$ by (s_{i-2}, s_1). Note that $\mathfrak{m}(s_0, s_1) = \mathfrak{m}(s_0, s_{i-1}) + \mathfrak{m}(s_{i-1}, s_1)$.

By Eq. (♣),

$$\begin{aligned} w(A) &= [f(s_0) - f(s_1) - \mathfrak{m}(s_0, s_1)] + [f(s_{i-2}) - f(s_{i-1}) - \mathfrak{m}(s_{i-2}, s_{i-1})] \\ &= [f(s_{i-2}) - f(s_1) - \mathfrak{m}(s_{i-2}, s_{i-1}) - \mathfrak{m}(s_{i-1}, s_1)] + [f(s_0) - f(s_{i-1}) - \mathfrak{m}(s_0, s_{i-1})] \\ &\leq [f(s_{i-2}) - f(s_1) - \mathfrak{m}(s_{i-2}, s_1)] \leq w(s_{i-2}, s_1) \end{aligned}$$

The total number of pairs has decreased, so we complete the contradiction. □

Lemma 3.4.12. *Suppose s_i exists. If Eq. (♣) holds for all even indices $< i$, then Eq. (♣) holds for i.*

Proof. We prove by contradiction, so Eq. (♣) is false for i. (Again, assume $i \equiv 2 \mod 4$. The other case is omitted.) By Claim 3.4.11, $E_+(i-2) \cup (s_{i-3}, s_{i+1})$ is a valid set of matched pairs. Let $M' := M - E_-(i) + (E_+(i-2) \cup (s_{i-3}, s_{i+1}))$. Observe that $|M'| = |M| - 1$ and the vertices s_{i-1} and s_i are left unmatched in M'. By Eq. (♣) for even indices $< i$ and the opposite of Eq. (♣) for i,

$$\begin{aligned} w(E_-(i)) &= [f(s_0) - f(s_1) - \mathfrak{m}(s_0, s_1)] + [f(s_3) - f(s_2) - \mathfrak{m}(s_3, s_2)] + \\ &\quad [f(s_4) - f(s_5) - \mathfrak{m}(s_4, s_5)] + [f(s_7) - f(s_6) - \mathfrak{m}(s_7, s_6)] + \cdots \\ &\quad [f(s_{i-2}) - f(s_{i-1}) - \mathfrak{m}(s_{i-2}, s_{i-1})] + [f(s_i) - f(s_{i+1}) - \mathfrak{m}(s_i, s_{i+1})] \end{aligned} \tag{3.8}$$

We stress that the last weight is "switched". We lower bound $w(E_+(i-2) \cup (s_{i-3}, s_{i+1}))$ appropriately.

$$\begin{aligned} w(E_+(i-2) \cup (s_{i-3}, s_{i+1})) \geq\ & [f(s_0) - f(s_2) - \mathfrak{m}(s_0, s_2)] + [f(s_4) - f(s_1) - \mathfrak{m}(s_4, s_1)] + \\ & [f(s_3) - f(s_6) - \mathfrak{m}(s_3, s_6)] + [f(s_8) - f(s_5) - \mathfrak{m}(s_8, s_5)] + \cdots \\ & [f(s_{i-7}) - f(s_{i-4}) - \mathfrak{m}(s_{i-7}, s_{i-4})] + [f(s_{i-2}) - f(s_{i-5}) - \\ & \mathfrak{m}(s_{i-2}, s_{i-5})] + [f(s_{i-3}) - f(s_{i+1}) - \mathfrak{m}(s_{i-3}, s_{i+1})] \end{aligned} \tag{3.9}$$

As before, we subtract Eq. (3.8) from Eq. (3.9). All function terms from Eq. (3.9) cancel out. By Claim 3.4.9, all $\mathfrak{m}$-terms except the first and last cancel out.

$$w(M') - w(M) \geq f(s_{i-1}) - f(s_i) - \mathfrak{m}(s_0, s_2) - \mathfrak{m}(s_{i-3}, s_{i+1}) + \mathfrak{m}(s_0, s_1) + \mathfrak{m}(s_{i-2}, s_{i-1}) + \mathfrak{m}(s_i, s_{i+1})$$

By linearity, $\mathfrak{m}(s_0, s_1) = \mathfrak{m}(s_0, s_2) + \mathfrak{m}(s_2, s_1)$. Furthermore, by Claim 3.4.8, $\mathfrak{m}(s_2, s_1) = \mathfrak{m}(s_i, s_{i-1})$. By Claim 3.4.9, $\mathfrak{m}(s_{i-2}, s_{i-1}) = \mathfrak{m}(s_{i-3}, s_i)$. By triangle inequality, $-\mathfrak{m}(s_{i-3}, s_{i+1}) +$

$\mathfrak{m}(s_{i-3}, s_i) + \mathfrak{m}(s_i, s_{i+1}) \geq 0$. Putting it all together and applying Claim 3.4.7,

$$w(M') - w(M) \geq -[f(s_i) - f(s_{i-1}) - \mathfrak{m}(s_i, s_{i-1})] \geq 0$$

So M' has at least the same weight but lower cardinality than M. Contradiction. $\square$

Lemma 3.4.13. *Suppose s_i exists. If Eq. (♣) holds for all even indices $< i$, then s_i is matched in* $\mathsf{st}(M)$.

Proof. Suppose not. (Again, assume $i \equiv 2 \mod 4$.) By Lemma 3.4.10, s_i is matched in M, so $(s_i, s_{i+1}) \in \mathsf{cr}(M)$. We set $M' = M - E_-(i) + (E_+(i) \cup (s_{i-1}, s_{i+1}))$. By Claim 3.4.11, M' is a valid matching. We have $|M'| = |M|$. M has two r-cross pairs (s_0, s_1) and (s_i, s_{i+1}), but M' has at most one (s_{i-1}, s_{i+1}). It suffices to show that $w(M') \geq w(M)$ to complete the contradiction. By Lemma 3.4.12 and Eq. (♣),

$$\begin{aligned} w(E_-(i)) &= [f(s_0) - f(s_1) - \mathfrak{m}(s_0, s_1)] + [f(s_3) - f(s_2) - \mathfrak{m}(s_3, s_2)] + \\ &\quad [f(s_4) - f(s_5) - \mathfrak{m}(s_4, s_5)] + [f(s_7) - f(s_6) - \mathfrak{m}(s_7, s_6)] + \cdots \\ &\quad [f(s_{i-2}) - f(s_{i-1}) - \mathfrak{m}(s_{i-2}, s_{i-1})] + [f(s_{i+1}) - f(s_i) - \mathfrak{m}(s_{i+1}, s_i)] \end{aligned}$$

$$\begin{aligned} w(E_+(i) \cup (s_{i-1}, s_{i+1})) &\geq [f(s_0) - f(s_2) - \mathfrak{m}(s_0, s_2)] + [f(s_4) - f(s_1) - \mathfrak{m}(s_4, s_1)] + \\ &\quad [f(s_3) - f(s_6) - \mathfrak{m}(s_3, s_6)] + [f(s_8) - f(s_5) - \mathfrak{m}(s_8, s_5)] + \cdots \\ &\quad [f(s_{i-3}) - f(s_i) - \mathfrak{m}(s_{i-3}, s_i)] + [f(s_{i+1}) - f(s_{i-1}) - \mathfrak{m}(s_{i+1}, s_{i-1})] \end{aligned}$$

All function terms and all but the first and last $\mathfrak{m}$-terms cancel out. The second inequality below holds by linearity and triangle inequality. The last equality is an application of Claim 3.4.8.

$$\begin{aligned} w(M') - w(M) &\geq \mathfrak{m}(s_0, s_1) - \mathfrak{m}(s_0, s_2) + \mathfrak{m}(s_{i+1}, s_i) - \mathfrak{m}(s_{i+1}, s_{i-1}) \\ &\geq \mathfrak{m}(s_2, s_1) - \mathfrak{m}(s_i, s_{i-1}) = 0 \end{aligned}$$

$\square$

Finally, we prove Theorem 3.4.2.

Proof. We started with a MWmC matching M with the minimum number of r-cross pairs. If there exists at least one such cross pair (x, y), we can define the alternating path sequence S. Wlog, we assumed Eq. (♣) holds for $i = 0$. Applications of Lemma 3.4.12 and Lemma 3.4.13 imply that S can never terminate. Contradiction. $\square$

3.4.2 Reducing From Arbitrary Product Distributions

We reduce arbitrary product distributions to uniform distributions on what we call the bloated hypergrid. Assume without loss of generality that all $\mu_{\mathcal{D}_r}(j) = q_r(j)/N$, for some integers $q_r(j)$ and N. Consider the d-dimensional N-hypergrid $[N]^d$. There is a natural many-to-one mapping from $\Phi : [N]^d \mapsto [n]^d$ defined as follows. First fix a dimension r. Given an integer $1 \leq t \leq N$, let $\phi_r(t)$ denote the index $\ell \in [1, n]$ such that $\sum_{j<\ell} q_r(j) < t \leq \sum_{j\leq \ell} q_r(j)$. That is, partition $[N]$ into n contiguous segments of lengths $q_r(1), \ldots, q_r(n)$. Then $\phi_r(t)$ is the index of the segment where t lies. The mapping $\Phi : [N]^d \mapsto [n]^d$ is defined as

$$\Phi(x_1, x_2 \ldots, x_d) = (\phi_1(x_1), \phi_2(x_2), \ldots, \phi_{\mathfrak{m}}(x_d)) \,.$$

We use Φ^{-1} to define the set of preimages, so Φ^{-1} maps a point in $[n]^d$ to a 'cuboid' in $[N]^d$. Observe that for any $x \in [n]^d$,

$$|\Phi^{-1}(x)| = N^d \prod_{r=1}^{d} \mu_{\mathcal{D}_r}(x) = N^d \mu_{\mathcal{D}}(x). \tag{3.10}$$

Claim 3.4.14. *For any set $X \subseteq [n]^d$, define $Z \subseteq [N]^d$ as $Z := \bigcup_{x\in X} \Phi^{-1}(x)$. Then $\mu_{\mathcal{D}}(X) = \mu_{\mathcal{U}}(Z)$.*

Proof. The set $Z = \bigcup_{x\in X} \Phi^{-1}(x)$ is the union of all the preimages of Φ over the elements of X. Since preimages are disjoint, we get $|Z| = \sum_{x\in X} |\Phi^{-1}(x)| = N^d \mu_{\mathcal{D}}(X)$. Therefore, $\mu_{\mathcal{U}}(Z) = \mu_{\mathcal{D}}(X)$. $\square$

Given $f : [n]^d \mapsto \mathbb{R}$, we define its extension $f_{\texttt{ext}} : [N]^d \mapsto \mathbb{R}$:

$$f_{\texttt{ext}}(x_1, \ldots, x_d) = f(\Phi(x_1, \ldots, x_d)). \tag{3.11}$$

Thus, $f_{\texttt{ext}}$ is constant on the cuboids in the bloated hypergrid corresponding to a point in the original hypergrid. Define the following metric on $[N]^d$.

$$\text{For } x, y \in [N]^d, \quad d_{\texttt{ext}}(x, y) = \mathfrak{m}(\Phi(x), \Phi(y)) \tag{3.12}$$

The following statements establish the utility of the bloated hypergrid, and the proof of the dimension reduction of f over $[n]^d$ w.r.t. $\mathcal{D}$ follows easily from these and the proof for the uniform distribution.

Lemma 3.4.15. *If $\mathfrak{m}$ satisfies the conditions of Lemma 3.2.3 over $[n]^d$, then so does $d_{\texttt{ext}}$ over $[N]^d$.*

Proof. Consider $x, y, z \in [N]^d$. Triangle inequality and well-definedness immediately follow from the validity of $\mathfrak{m}$. Now for linearity. If $x_r \leq y_r \leq z_r$, then so is $\phi_r(x_r) \leq \phi_r(y_r) \leq \phi_r(z_r)$. Thus, $\Phi(x), \Phi(y), \Phi(z)$ satisfy linearity w.r.t. $\mathfrak{m}$. So, $d_{\texttt{ext}}(x, z) = \mathfrak{m}(\Phi(x), \Phi(z)) = \mathfrak{m}(\Phi(x), \Phi(y)) + \mathfrak{m}(\Phi(y), \Phi(z)) = d_{\texttt{ext}}(x, y) + d_{\texttt{ext}}(y, z)$. Now for projection. Suppose $x_r = y_r$ and $x'_r = y'_r$. Note that $\Phi(x)$ and $\Phi(y)$ have same rth coordinate, and so do $\Phi(x')$ and $\Phi(y')$. Furthermore, $\Phi(x')$ (resp. $\Phi(y')$) is the projection of $\Phi(x)$ (resp. $\Phi(x)$). Thus we get $d_{\texttt{ext}}(x, y) = \mathfrak{m}(\Phi(x), \Phi(y)) = \mathfrak{m}(\Phi(x'), \Phi(y')) = d_{\texttt{ext}}(x', y')$, and similarly $d_{\texttt{ext}}(x, x') = d_{\texttt{ext}}(y, y')$. ☐

Theorem 3.4.16. $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{P}(\mathfrak{m})) = \mathsf{dist}_{\mathcal{U}}(f_{\texttt{ext}}, \mathcal{P}(d_{\texttt{ext}}))$.

Proof. ($\geq$). Let $X \subseteq [n]^d$ be a vertex cover in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P}(\mathfrak{m}))$ minimizing $\mu_{\mathcal{D}}(X)$. From Lemma 3.2.5, $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{P}(\mathfrak{m})) = \mu_{\mathcal{D}}(X)$. We claim $Z = \bigcup_{x \in X} \Phi^{-1}(x)$ is a vertex cover of $\mathcal{G}_{\mathsf{viol}}(f_{\texttt{ext}}, \mathcal{P}(d_{\texttt{ext}}))$. This implies $\mathsf{dist}_{\mathcal{U}}(f_{\texttt{ext}}, \mathcal{P}(d_{\texttt{ext}})) \leq \mu_{\mathcal{U}}(Z) = \mu_{\mathcal{D}}(X) = \mathsf{dist}_{\mathcal{D}}(f, \mathcal{P}(\mathfrak{m}))$, where the first equality follows from Claim 3.4.14. Consider a violated pair (u, v) in this graph and so wlog $f_{\texttt{ext}}(u) - f_{\texttt{ext}}(v) > d_{\texttt{ext}}(u, v)$. Hence, $f(\Phi(u)) - f(\Phi(v)) > \mathfrak{m}(\Phi(u), \Phi(v))$ implying $(\Phi(u), \Phi(v))$ is an edge in $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P}(\mathfrak{m}))$. Thus, either $\Phi(u)$ or $\Phi(v)$ lies in X implying either u or v lies in Z.($\leq$). Let $Z \subseteq [N]^d$ be a vertex cover in $\mathcal{G}_{\mathsf{viol}}(f_{\texttt{ext}}, \mathcal{P}(d_{\texttt{ext}}))$ minimizing $\mu_{\mathcal{U}}(Z)$. Therefore, $\mathsf{dist}_{\mathcal{U}}(f_{\texttt{ext}}, \mathcal{P}(d_{\texttt{ext}})) = \mu_{\mathcal{U}}(Z)$. Define $X \subseteq [n]^d$ as $X = \{x \in [n]^d : \Phi^{-1}(x) \subseteq Z\}$. Therefore, $Z \supseteq \bigcup_{x \in X} \Phi^{-1}(x)$ and from Claim 3.4.14 we get $\mu_{\mathcal{U}}(Z) \geq \mu_{\mathcal{D}}(X)$. It suffices to show that X is a vertex cover of $\mathcal{G}_{\mathsf{viol}}(f, \mathcal{P}(\mathfrak{m}))$. Consider a violated edge (x, y) in this graph such that $f(x) - f(y) > \mathfrak{m}(x, y)$. Suppose neither x nor y are in X. Hence, there exists $u \in \Phi^{-1}(x) \setminus Z$ and $v \in \Phi^{-1}(y) \setminus Z$. So $f_{\texttt{ext}}(u) - f_{\texttt{ext}}(v) = f(\Phi(u)) - f(\Phi(v)) = f(x) - f(y) > \mathfrak{m}(x, y) = d_{\texttt{ext}}(\Phi(u), \Phi(v))$, implying (u, v) is a violation in $\mathcal{G}_{\mathsf{viol}}(f_{\texttt{ext}}, \mathcal{P}(d_{\texttt{ext}}))$. This contradicts the fact that Z is a vertex cover. ☐

Fix a dimension r and r-line ℓ. Abusing notation, let $\Phi^{-1}(\ell)$ denote the collection of r-lines in $[N]^d$ that are mapped to ℓ by Φ. Note that $|\Phi^{-1}(\ell)| = N^{d-1}\mu_{\mathcal{D}_{-r}}(\ell)$. A proof identical to one above yields the following theorem.

Theorem 3.4.17. *For any r-line,* $\mathsf{dist}_{\mathcal{D}_r}(f_{|\ell}, \mathcal{P}(\mathfrak{m})) = \mathsf{dist}_{\mathcal{U}_r}(f_{\texttt{ext}|\ell'}, \mathcal{P}(d_{\texttt{ext}}))$ *for all $\ell' \in \Phi^{-1}(\ell)$.*

Now we can complete the proof of Theorem 3.4.1.

$$\begin{aligned}
\mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P}(\mathfrak{m})) &= \sum_{r\text{-line } \ell} \mu_{\mathcal{D}_{-r}}(\ell) \cdot \mathsf{dist}_{\mathcal{D}_r}(f_{|\ell}, \mathcal{P}(\mathfrak{m})) \\
&= \frac{1}{N^{d-1}} \sum_{r\text{-line } \ell} |\Phi^{-1}(\ell)| \cdot \mathsf{dist}_{\mathcal{D}_r}(f_{|\ell}, \mathcal{P}(\mathfrak{m})) \\
&= \frac{1}{N^{d-1}} \sum_{r\text{-line } \ell} \sum_{\ell' \in \Phi^{-1}(\ell)} \mathsf{dist}_{\mathcal{U}_r}(f_{\mathtt{ext}\,|\ell'}, \mathcal{P}(d_{\mathtt{ext}})) \\
&= \mathbf{E}_{\ell' \sim \mathcal{U}_{-r}}[\mathsf{dist}_{\mathcal{U}_r}(f_{\mathtt{ext}\,|\ell'}, \mathcal{P}(d_{\mathtt{ext}}))] = \mathsf{dist}^r_{\mathcal{U}}(f_{\mathtt{ext}}, \mathcal{P}(d_{\mathtt{ext}})).
\end{aligned}$$

We can apply the dimension reduction to $f_{\mathtt{ext}}$ for property $\mathcal{P}(d_{\mathtt{ext}})$ over the uniform distribution. The proof of Theorem 3.4.1 for f follows directly.

3.5 Search Trees and BDP Testing

As a result of dimension reduction, we can focus on designing testers for the line $[n]$. Our analysis is simple, but highlights the connection between bounded-derivative property testing and optimal search trees.

3.6 Testers For the Hypergrid

Given a series of BSTs $T_1, T_2, \ldots, T_d$ corresponding to each dimension, we have the following hypergrid BST tester.

Lemma 3.6.1. *For any set of BSTs $T_1, T_2, \ldots, T_d$, the probability of rejection is at least $\mathsf{dist}_{\mathcal{D}}(f, \mathcal{P})/4d$.*

Proof. Condition on an r-line being chosen. The probability distribution over r-lines for this tester is $\mathcal{D}_{-r}$. By Lemma 3.3.1, the rejection probability is at least $\mathbf{E}_{\ell \sim \mathcal{D}_{-r}}[\mathsf{dist}_{\mathcal{D}_r}(f_{|\ell}, \mathcal{P})] = \mathsf{dist}^r_{\mathcal{D}}(f, \mathcal{P})$. The overall rejection probability is at least $\sum_{r=1}^{d} \frac{\mathsf{dist}^i_{\mathcal{D}}(f,\mathcal{P})}{d} \geq \frac{\mathsf{dist}_{\mathcal{D}}(f,\mathcal{P})}{4d}$, by Theorem 3.4.1. ☐

The expected number of queries made by this procedure is at most $\frac{1}{d} \cdot \sum_{r=1}^{d} 2\Delta(T_r; \mathcal{D}_r)$. Repeating it $O(d/\varepsilon)$ times to get the desired tester. The proof of the following is identical to that of Lemma 3.3.2 and is omitted.

Lemma 3.6.2. *For any collections of BSTs $(T_1, \ldots, T_d)$, there is a $100\varepsilon^{-1} \sum_{i=1}^{d} \Delta(T; \mathcal{D})$-query tester for any bounded derivative property.*

As in the case of the line we get the following as corollaries.

Theorem 3.1.1. [Main upper bound] *Consider functions* $f : [n]^d \mapsto \mathbb{R}$. *Let* $\mathbf{B}$ *be a bounding family and* $\mathcal{D}$ *be a product distribution. There is a tester for* $\mathcal{P}(\mathbf{B})$ *w.r.t.* $\mathcal{D}$ *making* $100\varepsilon^{-1}\Delta^*(\mathcal{D})$ *queries.*

Theorem 3.1.5. *Consider functions* $f : [n]^d \mapsto \mathbb{R}$. *There is a distribution-free (non-adaptive, one-sided) tester for* $\mathcal{P}(\mathbf{B})$ *w.r.t.* $\mathcal{D}$ *making* $100\varepsilon^{-1}d\log n$ *queries.*

The upper bound $\sum_{r=1}^{d} \Delta^*(\mathcal{D}_r)$ is at most $H(\mathcal{D})$, but can be much smaller, and it is clearest in the case of the hypercube. In the hypercube, each $\mathcal{D}_r$ is given by $(\mu_r, 1-\mu_r)$. Set $\theta_r := \min(\mu_r, 1-\mu_r)$. The optimal BST places the point of larger mass on the root and has expected depth θ_r.

Corollary 3.1.3. *Consider functions* $f : \{0,1\}^d \mapsto \mathbb{R}$. *Monotonicity testing over any product distribution* $\mathcal{D} = \prod_{r=1}^{d} \mathcal{D}_r$, *where each* $\mathcal{D}_r = (\mu_r, 1-\mu_r)$, *can be done with* $100\varepsilon^{-1}\sum_{r=1}^{d} \min(\mu_r, 1-\mu_r)$ *queries.*

It is instructive to open up this tester. It samples a point x from the distribution $\mathcal{D}$ and picks a dimension r uniformly at random. With probability θ_r, it queries both endpoints of $(x, x \oplus \mathbf{e}_r)$. With probability $(1-\theta_r)$, it does *nothing*. This process is repeated $O(d/\varepsilon)$ times. When $\theta_r = \mu_r = 1/2$, this is the standard edge tester.

3.7 Lower Bounds

We prove that the upper bounds of Section 3.5 are tight up to the dependence on the distance parameter ε. As alluded to in Section 3.1.2, we can only prove lower bounds for *stable* product distributions. These are distributions where small perturbations to the mass function do not change Δ^* drastically.

Definition 3.7.1 (Stable Distributions). *A product distribution* $\mathcal{D}$ *is said to be* (ϵ, ρ)*-stable if for all product distributions* $\mathcal{D}'$ *with* $||\mathcal{D} - \mathcal{D}'||_{\mathsf{TV}} \leq \varepsilon$, $\Delta^*(\mathcal{D}') \geq \rho\Delta^*(\mathcal{D})$.

The uniform distribution on $[n]^d$ is $(\varepsilon, 1-o(1))$-stable, for any constant $\varepsilon < 1$. The Gaussian distribution also shares the same stability. An example of an unstable distribution is the following. Consider $\mathcal{D}$ on $[n]$, where the probability on the first $k = \log n$ elements is $(1-\varepsilon)/k$, and is $\varepsilon/(n-k)$ for all other elements. Let $\mathcal{D}'$ have all its mass uniformly spread on the first k elements. We have $||\mathcal{D} - \mathcal{D}'||_{\mathsf{TV}} = \epsilon$ but $\Delta^*(\mathcal{D}) \approx \varepsilon \log n$ and $\Delta^*(\mathcal{D}') \approx \log k = \log\log n$.

Theorem 3.1.4. [Main lower bound] *For any parameter* ϵ, *there exists* $\epsilon' = \Theta(\epsilon)$ *such that for any bounding family* $\mathbf{B}$ *and* (ε', ρ)*-stable, product distribution* $\mathcal{D}$, *any (even adaptive, two-sided) tester for* $\mathcal{P}(\mathbf{B})$ *w.r.t.* $\mathcal{D}$ *with proximity parameter* ε *requires* $\Omega(\rho\Delta^*(\mathcal{D}))$ *queries.*

3.7.1 Reduction from Monotonicity to a Bounded-Derivative Property

Consider a function $f : [n]^d \mapsto [R]$ with where $R \in \mathbb{N}$. Let $\mathfrak{m}$ be the distance function obtained by bounding family $\mathbf{B}$. We let $\mathbf{0} \in [n]^d$ be $(0, 0, \ldots, 0)$. We use $\prec$ to denote the natural partial order in $[n]^d$, and let $\mathtt{hcd}(x, y)$ be the highest common descendant of $x, y \in [n]^d$. We first prove an observation about triangle equality.

Observation 3.7.2. *If* $\mathfrak{m}(\mathbf{0}, x) + \mathfrak{m}(x, y) = \mathfrak{m}(\mathbf{0}, y)$*, then* $x \prec y$.

Proof. By linearity, $\mathfrak{m}(x, y) = \mathfrak{m}(x, \mathtt{hcd}(x, y)) + \mathfrak{m}(\mathtt{hcd}(x, y), y)$. Since $\mathtt{hcd}(x, y) \prec x$, by linearity again, $\mathfrak{m}(\mathbf{0}, x) = \mathfrak{m}(\mathbf{0}, \mathtt{hcd}(x, y)) + \mathfrak{m}(\mathtt{hcd}(x, y), x)$. (Similarly for y.) Putting it all into the 'if' condition,

$$\mathfrak{m}(\mathbf{0}, \mathtt{hcd}(x, y)) + \mathfrak{m}(\mathtt{hcd}(x, y), x) + \mathfrak{m}(x, \mathtt{hcd}(x, y)) + \mathfrak{m}(\mathtt{hcd}(x, y), y) = \mathfrak{m}(\mathbf{0}, \mathtt{hcd}(x, y)) + \mathfrak{m}(\mathtt{hcd}(x, y), y)$$

This yields $\mathfrak{m}(\mathtt{hcd}(x, y), x) + \mathfrak{m}(x, \mathtt{hcd}(x, y)) = 0$. Suppose $\mathtt{hcd}(x, y) \neq x$. The length (in terms of $\mathbf{B}$) of the path from $\mathtt{hcd}(x, y)$ to x involves a sum of $u_i(t)$ terms, and the reverse path involves corresponding $-l_i(t)$ terms. Since $u_i(t) > l_i(t)$, the total path length from $\mathtt{hcd}(x, y)$ to x and back is strictly positive. Therefore, $\mathtt{hcd}(x, y) = x$ and $x \prec y$. ☐

Let U be the set of incomparable (ordered) pairs in $[n]^d$. Define $\delta := \min_{(x,y) \in U} \{\mathfrak{m}(\mathbf{0}, x) + \mathfrak{m}(x, y) - \mathfrak{m}(\mathbf{0}, y)\}$. By Observation 3.7.2, $\delta > 0$. Define

$$g(x) := \frac{\delta}{2R} \cdot f(x) - \mathfrak{m}(\mathbf{0}, x)$$

Lemma 3.7.3. $\mathsf{dist}_{\mathcal{D}}(g, \mathcal{P}) = \mathsf{dist}_{\mathcal{D}}(f, \mathtt{MON})$.

Proof. We show that (u, v) violates $\mathcal{P}(\mathfrak{m})$ of g iff it violates monotonicity of f. First, the 'only if' case. Assume $g(u) - g(v) > \mathfrak{m}(u, v)$. Plugging in the expression for $g(\cdot)$ and rearranging,

$$\frac{\delta}{2R}(f(u) - f(v)) > \mathfrak{m}(\mathbf{0}, u) + \mathfrak{m}(u, v) - \mathfrak{m}(\mathbf{0}, v)$$

By triangle inequality on the RHS, $f(u) > f(v)$. Note that $f(u) - f(v) \leq R$ so $\frac{\delta}{2R}(f(u) - f(v)) \leq \delta/2$. So $\delta/2 > \mathfrak{m}(\mathbf{0}, u) + \mathfrak{m}(u, v) - \mathfrak{m}(\mathbf{0}, v)$. By choice of δ, the RHS must be zero. By Observation 3.7.2, $u \prec v$, and (u, v) is a violation to monotonicity of f. Now the 'only if' case, so

$u \prec v$ and $f(u) > f(v)$. Note that $\mathfrak{m}(\mathbf{0}, v) = \mathfrak{m}(\mathbf{0}, u) + \mathfrak{m}(u, v)$. We deduce that (u, v) is also a violation to $\mathcal{P}(\mathfrak{m})$ for g.

$$g(u) - g(v) = \frac{\delta}{2R}(f(u) - f(v)) + \mathfrak{m}(\mathbf{0}, v) - \mathfrak{m}(\mathbf{0}, u) = \frac{\delta}{2R}(f(u) - f(v)) + \mathfrak{m}(u, v) > \mathfrak{m}(u, v)$$

□

Our main reduction theorem is the following.

Theorem 3.7.4. *Fix domain $[n]^d$ and a product distribution $\mathcal{D}$. Suppose there exists a Q-query tester for testing a bounded-derivative property $\mathcal{P}$ with distance parameter ϵ. Then there exists a $Q + 10/\epsilon$-query tester for monotonicity for functions $f : [n]^d \mapsto \mathbb{N}$ over $\mathcal{D}$ with distance parameter 2ϵ.*

Proof. The monotonicity tester first queries $10/\varepsilon$ points of $[n]^d$, each i.i.d. from $\mathcal{D}$. Let the maximum f-value among these be this M. Consider the truncated function $f' : [n]^d \mapsto [M]$, where $f'(x) = M$ if $f(x) \geq M$ and $f'(x) = f(x)$ otherwise. If f is monotone, f' is monotone. Note that $\text{dist}_{\mathcal{D}}(f, f') < \varepsilon$. So if f is 2ε-far from monotone, f' is ε-far from monotone. We can apply the $\mathcal{P}(\mathbf{B})$ tester on the function g obtained from Lemma 3.7.3. □

3.7.2 Monotonicity Lower Bound Framework

The lower bound for monotonicity testing goes by the proof strategy set up in [CS13b]. This is based on arguments in [Fis04, CS13b] that reduce general testers to comparison-based testers. We encapsulate the main approach in the following theorem, proven implicitly in [CS13b]. (We use `MON` to denote the monotonicity property.)

Theorem 3.7.5. *Fix domain $[n]^d$, distribution $\mathcal{D}$, proximity parameter ε, and positive integer L possibly depending on $\mathcal{D}$ and ε. A pair (x, y) distinguishes function g from h if $h(x) < h(y)$ and $g(x) > g(y)$. Suppose there is a collection of 'hard' functions $h, g_1, \ldots, g_L : [n]^d \mapsto \mathbb{N}$ such that*

- *The function h is monotone.*
- *Every $\text{dist}_{\mathcal{D}}(g_i, \texttt{MON}) \geq \varepsilon$.*
- *Pairs in any set $Q \subset [n]^d$, can distinguish at most $|Q|$ of the g_i's from h.*

Then any (even adaptive, two-sided) monotonicity tester w.r.t. $\mathcal{D}$ for functions $f : [n]^d \mapsto \mathbb{N}$ with distance parameter ε must make $\Omega(L)$ queries.

In Section 3.7.3 and Section 3.7.4, we first describe hard functions for the line and the hypercube domain, respectively. The general hypergrid is addressed in Section 3.7.5.

3.7.3 The Line

Theorem 3.7.6. *Fix a parameter ε. If $\mathcal{D}$ is $(2\varepsilon, \rho)$-stable, then any ϵ-monotonicity tester w.r.t. $\mathcal{D}$ for functions $f : [n]^d \mapsto \mathbb{N}$ requires $\Omega(\rho\Delta^*(\mathcal{D}))$ queries.*

Not surprisingly, the lower bound construction is also based on BSTs. We specifically use the *median BST* [Meh75]. When $n = 1$, then the tree is the singleton. For a general n, let $t \in [n]$ be the smallest index such that $\mu(\{1, \cdots, t\}) \geq 1/2$ (henceforth, in this section, we use μ to denote $\mu_{\mathcal{D}}$). The root of T is t. Recur the construction on the intervals $[1, t-1]$ and $[t+1, n]$. By construction, the probability mass of any subtree together with its parent is greater than the probability mass of the sibling subtree. This *median property* will be utilized later. We follow the framework of Theorem 3.7.5 to construct a collection of hard functions. The monotone function h can be anything; $h(i) = 3i$ works. We will construct a function g_j $(j \geq 1)$ for each non-root level of the median BST. Consider the nodes at depth $j-1$ (observe the use of $j-1$, and not j). Each of these corresponds to an interval, and we denote this sequence of intervals by $\mathsf{I}_j^1, \mathsf{I}_j^2, \ldots$. (Because internal nodes of the tree are also elements in $[n]$, there are gaps between these intervals.) Let $L_{\geq j} := \{x : \mathsf{depth}_T(x) \geq j\}$ be the nodes at depth j and higher. We have the following simple claim.

Claim 3.7.7. *I_j^k can be further partitioned into $\mathsf{I}_j^{k,\mathsf{left}}$ and $\mathsf{I}_j^{k,\mathsf{right}}$ such that $\sum_k \min\left(\mu(\mathsf{I}_j^{k,\mathsf{left}}), \mu(\mathsf{I}_j^{k,\mathsf{right}})\right) \geq \frac{\mu(L_{\geq j})}{2}$.*

Proof. Consider the node u_k corresponding I_j^k, and let the nodes in the left and right subtrees be S_ℓ and S_r. If $\mu(S_\ell) \leq \mu(S_r)$, then $\mathsf{I}_j^{k,\mathsf{left}} = S_\ell \cup u_k$ and $\mathsf{I}_j^{k,\mathsf{right}} = S_r$. Otherwise, $\mathsf{I}_j^{k,\mathsf{left}} = S_\ell$ and $\mathsf{I}_j^{k,\mathsf{right}} = u_k \cup S_r$. By the median property of the BST, $\min(\mu(\mathsf{I}_j^{k,\mathsf{left}}), \mu(\mathsf{I}_j^{k,\mathsf{right}})) = \mathsf{max}(\mu(S_\ell), \mu(S_r))$ $\geq (\mu(S_\ell) + \mu(S_r))/2$. □

We describe the non-monotone g_j's and follow up with some claims. Let $\mathsf{lca}(x, y)$ denote the least common ancestor of x and y in T.

$$g_j(x) = \begin{cases} 2x \quad \text{if } x \notin \bigcup_k \mathsf{I}_j^k \\ 2x + 2(b-m) + 1 \quad \text{if } x \in \mathsf{I}_j^{k,\mathsf{left}} = [a, m], \quad \text{where } \mathsf{I}_j^k = [a, b]. \\ 2x - 2(m-a) - 1 \quad \text{if } x \in \mathsf{I}_j^{k,\mathsf{right}} = [m+1, b], \text{ where } \mathsf{I}_j^k = [a, b]. \end{cases} \tag{3.13}$$

Claim 3.7.8. (i) $\mathsf{dist}_{\mathcal{D}}(g_j, \mathtt{MON}) \geq \frac{\mu(L_{\geq j})}{2}$. (ii) *If (x, y) distinguishes g_j from h, then $\mathsf{lca}(x, y)$ lies in level $(j-1)$.*

Proof. All elements in $\mathsf{I}_j^{k,\mathsf{left}}$ are in violation with all elements in $\mathsf{I}_j^{k,\mathsf{right}}$ for all k. To see this, let $x \in \mathsf{I}_j^{k,\mathsf{left}}$ and $y \in \mathsf{I}_j^{k,\mathsf{right}}$, and so $x \prec y$. Denote $\mathsf{I}_j^k = [a, b]$,

$$g_j(x) - g_j(y) = 2x + 2(b - m) + 1 - 2y + 2(m - a) + 1 = 2(x - a) + 2(b - y) + 2 > 0$$

The vertex cover of the violation graph of g_i has mass at least $\sum_k \min(\mu(\mathsf{I}_j^{k,\mathsf{left}}), \mu(\mathsf{I}_j^{k,\mathsf{right}})) \geq \mu(L_{\geq j})/2$ (Claim 3.7.7). This proves part (i). To prove part (ii), let $x \prec y$ distinguish g_j from h, so $g_j(x) > g_j(y)$. We claim there exists a k^* such that $x \in \mathsf{I}_j^{k^*,\mathsf{left}}$ and $y \in \mathsf{I}_j^{k^*,\mathsf{right}}$. For any $\mathsf{I}_j^k = [a, b]$, the g_j values lie in $[2a + 1, 2b + 1]$. Hence, if $x \in \mathsf{I}_j^k$ and $y \notin \mathsf{I}_j^k$ (or vice versa), (x, y) is not a violation. So x and y lie in the same $\mathsf{I}_j^{k^*}$, But the function restricted to $\mathsf{I}_j^{k^*,\mathsf{left}}$ or $\mathsf{I}_j^{k^*,\mathsf{right}}$ is increasing, completing the proof. □

The following claim is a simple combinatorial statement about trees.

Claim 3.7.9. *Given a subset Q of $[n]$, let $\mathsf{lca}(Q) = \{\mathsf{lca}(x, y) : x, y \in Q\}$. Then $|\mathsf{lca}(Q)| \leq |Q| - 1$.*

Proof. The proof is by induction on $|Q|$. The base case of $|Q| = 2$ is trivial. Suppose $|Q| > 2$. Consider the subset $P \subseteq Q$ of all elements of Q, none of whose ancestors are in Q. Also observe that if $P = Q$, then $\mathsf{lca}(Q)$ are precisely the internal nodes of a binary tree whose leaves are Q, and therefore $|\mathsf{lca}(Q)| \leq |Q| - 1$. If P is a singleton, then $\mathsf{lca}(Q) = \mathsf{lca}(Q \setminus P) + 1 \leq |Q \setminus P| - 1 + 1 = |Q| - 1$ (inequality from induction hypothesis). So assume $P \subset Q$ and $|P| \neq 1$. For $p \in P$, let S_p be the set of elements of Q appearing in the tree rooted at p. For every $x \in S_p$ and $y \in S_{p'}$ ($p \neq p'$), $\mathsf{lca}(x, y) = \mathsf{lca}(p, p')$. Furthermore, the sets S_p non-trivially partition Q. Therefore, $\mathsf{lca}(Q) = \mathsf{lca}(P) \cup \bigcup_{p \in P} \mathsf{lca}(S_p)$. Applying the induction hypothesis, $|\mathsf{lca}(Q)| \leq |P| - 1 + \sum_{p \in P} |S_p| - |P| = |Q| - 1$. □

Let ℓ_ε be the largest ℓ such that $\mu(L_{\geq \ell}) \geq 2\varepsilon$. By Claim 3.7.8.(i), the collection of functions $\{g_1, \ldots, g_{\ell_\varepsilon}\}$ are each ε-far from monotone. By Claim 3.7.8.(ii) and Claim 3.7.9, a subset $Q \subseteq [n]$ can't distinguish more than $|Q|$ of these functions from h. Theorem 3.7.5 gives an $\Omega(\ell_\varepsilon)$ lower bound and Theorem 3.7.6 follows from Claim 3.7.10.

Claim 3.7.10. $\ell_\epsilon \geq \rho \Delta^*(\mathcal{D})$.

Proof. Consider the distribution $\mathcal{D}'$ that transfers all the mass from $L_{\geq \ell_\varepsilon + 1}$ to the remaining vertices proportionally. That is, if $\nu := \mu(L_{\geq \ell_\varepsilon + 1})$, then $\mu_{\mathcal{D}'}(i) = 0$ for $i \in L_{\geq \ell_\varepsilon + 1}$, and $\mu_{\mathcal{D}'}(i) = \mu_{\mathcal{D}}(i)/(1 - \nu)$ for the rest. Observe that $||\mathcal{D} - \mathcal{D}'||_{\mathsf{TV}} = \mu_{\mathcal{D}}(L_{\geq \ell_\varepsilon + 1}) < 2\varepsilon$. Also observe that since T is a binary tree of height ℓ_ε, $\ell_\varepsilon \geq \Delta^*(\mathcal{D}')$: the LHS is the max depth, the RHS is the (weighted) average depth. Now, we use stability of $\mathcal{D}$. Since $\mathcal{D}$ is $(2\varepsilon, \rho)$-stable, $\Delta^*(\mathcal{D}') \geq \rho \Delta^*(\mathcal{D})$. □

3.7.4 The Boolean Hypercube

For the boolean hypercube, the lower bound doesn't require the stability assumption. Any product distribution over $\{0,1\}^d$ is determined by the d fractions $(\mu_1, \ldots, \mu_d)$, where μ_r is the probability of 0 on the r-th coordinate. Let $\theta_r := \min(\mu_r, 1-\mu_r)$.

Theorem 3.7.11. *Any monotonicity tester w.r.t. $\mathcal{D}$ for functions $f : \{0,1\}^d \mapsto \mathbb{N}$ with distance parameter $\epsilon \leq 1/10$ must make $\Omega\left(\sum_{r=1}^{d} \min(\mu_r, 1-\mu_r)\right)$ queries.*

We begin with the basic setup. A tester for non-trivial ε makes at least 1 query, so we can assume that $\sum_{r=1}^{d} \theta_r > 1$. For ease of exposition, assume $\theta_r = \mu_r$, for all $1 \leq r \leq d$. (If not, we need to divide into two cases depending on θ_r and argue analogously for each case.) Assume wlog $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_d$. Partition $[d]$ into contiguous segments $I_1, \ldots, I_b, I_{b+1}$ such that for each $1 \leq a \leq b$, $\sum_{r \in I_a} \theta_r \in [1/2, 1)$. Observe that $b = \Theta\left(\sum_r \theta_r\right)$. For $1 \leq a \leq b$, define the indicator functions $\chi_a : \{0,1\}^d \mapsto \{0,1\}$ as follows:

$$\chi_a(x) = \begin{cases} 1 & \text{if } \forall i \in I_a, x_i = 1 \\ 0 & \text{otherwise} \quad (\exists i \in I_a, x_i = 0) \end{cases}$$

By Theorem 3.7.5, we need to define the set of functions with appropriate properties. The monotone function $h(\cdot)$ is defined as $h(x) = \sum_{a=1}^{b} \chi_a(x) 2^a$. The functions $g_1, \ldots, g_b$ are defined as

$$g_a(x) = \begin{cases} h(x) - 2^r - 1 & \text{if } \chi_a(x) = 1 \\ h(x) & \text{if } \chi_a(x) = 0 \end{cases}$$

We prove all the desired properties.

Claim 3.7.12. *For all a, $\mathsf{dist}_{\mathcal{D}}(g_a, \mathtt{MON}) \geq 1/10$.*

Proof. Let I denote I_a, and $J = [n] \setminus I$. Think of $x = (x_I, x_J)$. Fix $\mathbf{v}$ in $\{0,1\}^{|J|}$, and define sets $X_1(\mathbf{v}) := \{x | \chi_a(x) = 1, x_J = \mathbf{v}\}$ (a singleton) and $X_0(\mathbf{v}) = \{x | \chi_a(x) = 0, x_J = \mathbf{v}\}$. Note that $\bigcup_{\mathbf{v}}(X_1(\mathbf{v}) \cup X_0(\mathbf{v}))$ forms a partition of the cube. For $c \neq a$, $\chi_c(x)$ is the same for all $x \in (X_0(\mathbf{v}) \cup X_1(\mathbf{v}))$. Hence, for *any* $x \in X_0(\mathbf{v})$ and $y \in X_1(\mathbf{v})$, $x \prec y$ and $g_a(x) > g_a(y)$. Any vertex cover in the violation graph must contain either $X_1(\mathbf{v})$ or $X_0(\mathbf{v})$, for each $\mathbf{v}$. Let $\mathcal{D}_I$ be the conditional distribution on the I-coordinates. In the following, we use the inequalities $\sum_{i \in I} \theta_i \in [1/2, 1)$ and $1 - t \in [e^{-2t}, e^t]$ for $t \leq 1/2$.

$$\mu_{\mathcal{D}_I}(X_1(\mathbf{v})) \quad = \quad \prod_{i \in I}(1 - \theta_i) \geq \exp(-2\sum_{i \in I} \theta_i) \geq e^{-2} > 1/10$$

$$\mu_{\mathcal{D}_I}(X_0(\mathbf{v})) \;=\; 1-\prod_{i\in I}(1-\theta_i) \geq 1-\exp(-\sum_{i\in I}\theta_i) \geq 1-e^{-1/2} > 1/10.$$

For each $\mathbf{v}$, the conditional mass of the vertex cover is at least $1/10$, and therefore, the $\mu_{\mathcal{D}}$ mass of the vertex cover is at least $1/10$. ☐

A pair x, y in $[n]^d$ *captures* index a if a is the largest index such that $\chi_a(x) \neq \chi_a(y)$. Furthermore, a set Q captures a if it contains a pair capturing a.

Claim 3.7.13. *If Q distinguishes g_a from h, then Q must capture a.*

Proof. Consider $x, y \in Q$ where $h(x) < h(y)$ but $g_a(x) > g_a(y)$. It must be that $\chi_a(x) = 0$ and $\chi_a(y) = 1$. Suppose this pair does not capture a. There must exist index $c > a$ (let it be the largest) such that $\chi_c(x) \neq \chi_c(y)$. Because $h(x) < h(y)$, $\chi_c(x) = 0$ and $\chi_c(y) = 1$. By definition, $g_a(y) - g_a(x) = (h(y) - 2^a - 1) - h(x)$. We have $h(y) - h(x) = \sum_{t=1}^{c}(\chi_t(y) - \chi_t(x))2^t$. Since $\chi_a(x) = \chi_c(x) = 0$ and $\chi_a(y) = \chi_c(y) = 1$, $h(y) - h(x) \geq 2^c + 2^a - \sum_{t<c:t\neq a} 2^t$. Combining,

$$g_a(y) - g_a(x) \geq 2^c + 2^a - \sum_{t<c:t\neq a} 2^t - 2^a - 1 = 2^a > 0.$$

☐

Claim 3.7.14. *[Lifted from [CS13b].] A set Q captures at most $|Q| - 1$ coordinates.*

Proof. We prove this by induction on $|Q|$. When $|Q| = 2$, this is trivially true. Otherwise, pick the largest coordinate j captured by Q and let $Q_0 = \{x : x_j = 0\}$ and $Q_1 = \{x : x_j = 1\}$. By induction, Q_0 captures at most $|Q_0| - 1$ coordinates, and Q_1 captures at most $|Q_1| - 1$ coordinates. Pairs $(x, y) \in Q_0 \times Q_1$ only capture coordinate j. The total number of captured coordinates is at most $|Q_0| - 1 + |Q_1| - 1 + 1 = |Q| - 1$. ☐

We can now invoke Theorem 3.7.5 to get an $\Omega(b) = \Omega(\sum_r \theta_r)$ lower bound thereby proving Theorem 3.7.11. The hypercube lower bound can be generalized to give a weak lower bound for hypergrids, which will be useful for proving the stronger bound. Fix a dimension r. For any $1 \leq j \leq n$, define $\theta_r^j := \min(\sum_{k\leq j} \mu_{\mathcal{D}_r}(k), 1 - \sum_{k\leq j} \mu_{\mathcal{D}_r}(k))$. Define $\theta_r := \max_{1\leq j\leq n}\theta_r^j$. Note that θ_r generalizes the above definition for the hypercube. The following theorem follows by a reduction to the hypercube lower bound.

Theorem 3.7.15. *Any monotonicity tester on the hypergrid with distance parameter $\varepsilon \leq 1/10$, makes $\Omega\left(\sum_{r=1}^{d} \theta_r\right)$ queries.*

Proof. For $1 \leq r \leq d$, let $1 \leq j_r \leq n$ be the j such that $\theta_r = \theta_r^j$. Project the hypergrid onto a Boolean hypercube using the following mapping $\psi : [n]^d \to \{0,1\}^d$: for $x \in [n]^d$, $\psi(x)_r = 0$ if $x_r \leq j_r$, and 1 otherwise. The corresponding product distribution $\mathcal{D}'$ on the hypercube puts $\mu_{\mathcal{D}'_r}(0) = \sum_{k \leq j_r} \mu_{\mathcal{D}_r}(k)$, for all r. Note that $\min(\mu_r, 1-\mu_r) = \theta_r$. Given any function f on $\{0,1\}^d$, extend it to g over the hypergrid in the natural way: for $x \in [n]^d$, $g(x) = f(\psi(x))$. Note that $\mathsf{dist}_{\mathcal{D}'}(f, \texttt{MON}) = \mathsf{dist}_{\mathcal{D}}(g, \texttt{MON})$. (This is akin to Theorem 3.4.16.) Any tester for g over $[n]^d$ induces a tester for f on $\{0,1\}^d$ with as good a query complexity: whenever the hypergrid tester queries $x \in [n]^d$, the hypercube tester queries $\psi(x)$. Therefore, the lower bound Theorem 3.7.11 for the hypercube implies Theorem 3.7.15. □

3.7.5 The Hypergrid

Our main lower bound result is the following, which implies Theorem 3.1.4 via Theorem 3.7.4.

Theorem 3.7.16. *For any parameter $\epsilon < 1/10$, and for any $(120\varepsilon, \rho)$-stable, product distribution $\mathcal{D}$, any (even adaptive, two-sided) montonicity tester w.r.t. $\mathcal{D}$ for functions $f : [n]^d \mapsto \mathbb{N}$ with proximity parameter ε requires $\Omega(\rho\Delta^*(\mathcal{D}))$ queries.*

3.7.5.1 Intuition

Since we already have a proof for $d = 1$ in Section 3.7.3, an obvious approach to prove Theorem 3.1.4 is via some form of induction on the dimension. Any of the g_j-functions on $[n]$ in Section 3.7.3 can be extended the obvious way to a function on $[n]^d$. Given (say) $g_j : [n] \mapsto \mathbb{N}$, we can define $f : [n]^d \mapsto \mathbb{N}$ as $f(x) = g_j(x_1)$. Thus, we embed the hard functions for $\mathcal{D}_1$ along dimension 1. One can envisage a way do the same for dimension 2, and so on and so forth, thereby leading to $\sum_i \Delta^*(\mathcal{D}_i)$ hard functions in all. There is a caveat here. The construction of Section 3.7.3 for (say) $\mathcal{D}_1$ requires the stability of $\mathcal{D}_1$. Otherwise, we don't necessarily get $\Omega(\Delta^*(\mathcal{D}_1))$ functions with distance at least ϵ. For instance, if the root of the median BST has more than $(1-\epsilon)$ fraction of the weight, we get at most one hard function of distance at least ϵ. So, the above approach requires stability of all the *marginals* of $\mathcal{D}$. Unfortunately, there exist stable product distributions with all marginals unstable. Consider $\mathcal{D} = \prod_r \mathcal{D}_r$, where each $\mathcal{D}_r = (\frac{1}{(n-1)d}, \ldots, \frac{1}{(n-1)d}, 1 - \frac{1}{d})$. Note that $\Delta^*(\mathcal{D}) \approx \log n$. Each $\mathcal{D}_r$ is individually unstable (for $\varepsilon > 1/d$), since there is a $\mathcal{D}'_i$ with all the mass on the nth coordinate, such that $\|\mathcal{D}_i - \mathcal{D}'_i\|_{TV} = 1/d$ and $\Delta^*(\mathcal{D}'_i) = 0$. On the other hand, it is not hard to see that $\mathcal{D}$ is $(1/100, 1/100)$-stable. A new idea is required to construct the lower bound. To see this, suppose there is a product distribution $\mathcal{D}'$ such that $\Delta^*(\mathcal{D}') < \Delta^*(\mathcal{D})/100 = (\log n)/100$. Markov's inequality implies that for $\Omega(d)$ dimensions, $\|\mathcal{D}'_r - \mathcal{D}_r\|_{\mathsf{TV}} = \Omega(1/d)$. A calculation shows that $\|\mathcal{D} - \mathcal{D}'\|_{\mathsf{TV}}$ must be at least $1/100$. In sum, for any constants ε, ρ, there exist (ε, ρ)-stable distributions $\mathcal{D}$ such that each marginal $\mathcal{D}_r$ is only $(\varepsilon/d, \rho)$-stable. This is a major roadblock for a lower bound construction, and therefore a new idea is required. We design an *aggregation* technique that does the following. Start with 1D functions $g^1_{j_1}$ and $g^2_{j_2}$ that are hard functions from Section 3.7.3 for $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively. Suppose the corresponding distances to monotonicity are $\varepsilon^{(1)}$ and $\varepsilon^{(2)}$. We construct a function $f : [n]^d \mapsto \mathbb{N}$ that is $\varepsilon^{(1)} + \varepsilon^{(2)}$-far, so we can effectively add their distances. If we can aggregate $\Omega(d)$ 1D functions, each with distance ε/d, then we get a desired hard function. As can be expected, this construction is quite delicate, because we embed violations in many dimensions simultaneouly. Furthermore, we need to argue that this aggregation can produce enough "independent" hard functions, so we get a large enough lower bound (from Theorem 3.7.5). And that is where the hard work lies.

3.7.5.2 Setup and Construction

Fix ϵ and let $\epsilon' = 120\epsilon$. Fix the (ϵ', ρ)-stable distribution $\mathcal{D}$. Since $\mathcal{D}$ is (ϵ', ρ)-stable, for any $\mathcal{D}'$ with $\|\mathcal{D}' - \mathcal{D}\|_{\mathsf{TV}} \leq \epsilon'$, we have $\Delta^*(\mathcal{D}') \geq \rho\Delta^*(\mathcal{D})$. We denote the median BST for $\mathcal{D}_r$ as T_r, Δ_r as the expected depth w.r.t. $\mathcal{D}_r$, and $\Delta(\mathcal{D}) = \sum_{r=1}^{d} \Delta_r$. The following shows that the median BST is near optimal.

Lemma 3.7.17. *For any product distribution $\mathcal{D} = \prod_r \mathcal{D}_r$, $\Delta(\mathcal{D}) \leq 5\Delta^*(\mathcal{D})$.*

Proof. Fix a coordinate r. The depth of a vertex u in T_r is at most $\log_2(1/\mu_{\mathcal{D}_r}(u))$, so we get $\Delta_r \leq H(\mathcal{D}_r)$, the Shannon entropy of $\mathcal{D}_r$. It is also known (cf. Thm 2 in [Meh75]) that $H(\mathcal{D}_r) \leq \log_2 3(\Delta^*(\mathcal{D}_r) + 1)$. To see this, notice that any BST can be converted into a prefix-free ternary code of expected length $(\Delta^*(\mathcal{D}_r) + 1)$, say, over the alphabet 'left','right', and 'stop'. Therefore, if $\Delta^*(\mathcal{D}_r) \geq 1/2$, we have $\Delta_r \leq 5\Delta^*(\mathcal{D}_r)$. If $\Delta^*(\mathcal{D}_r) < 1/2$, then since $\Delta^*(\mathcal{D}_r) \geq 1 - \Pr[\text{root}]$, we get $\mu^* := \mu_{\mathcal{D}_r}(u^*) > 1/2$ where u^* is the root of the optimal BST T^*. But this implies u^* is also the root of T_r by construction of the median BST. Now we can prove via induction. If p and q are the total masses of the nodes in the left and right sub-tree of T^* (and therefore also T_r), and Δ_1^* (resp. Δ_1) and Δ_2^* (resp. Δ_2)be the expected depths of these subtrees in T^* (resp. T_r), then we get, $\Delta^*(\mathcal{D}_r) = p\Delta_1^* + q\Delta_2^* + (1 - \mu^*) \leq 5p\Delta_1 + 5q\Delta_2 + (1 - \mu^*) \leq 5\Delta_r$. □

Theorem 3.7.5 requires the definition of a monotone function and a collection of ε-far from monotone functions with additional properties. The monotone function is $\mathsf{val}(x) := \sum_{r=1}^{d} 2(2n + 1)^r x_r$. The non-monotone functions (which we refer to as "hard" functions) are constructed via aggregation. From Section 3.7.3, for each dimension r and each level $j \geq 1$ in tree T_r, we have a

1D “hard" function $g_j^r : [n] \mapsto \mathbb{N}$. It is useful to abstract out some of the properties of g_j^r that were proved in Section 3.7.3. Let L_j^r be the nodes in T_r at level j. Each level corresponds to a collection of intervals of $[n]$. We use $L_{\geq j}^r := \bigcup_{j' \geq j} L_{j'}^r$ and $L_{<j}^r = \bigcup_{j'<j} L_{j'}^r$. We use the shorthand $\mu_{\geq j}^r$ to denote $\mu_{\mathcal{D}_r}(L_{\geq j}^r)$. The following lemma is a restatement of Claim 3.7.7 and Claim 3.7.8.

Lemma 3.7.18. *Consider $g_j^r : [n] \mapsto \mathbb{N}$, for $j \geq 1$ All violations to monotonicity are contained in intervals corresponding to L_{j-1}^r, and the distance to monotonicity is at least $\mu_{\geq j}^r/2$. Furthermore, any violation (x, y) has $\mathsf{lca}(x, y)$ in L_{j-1}^r.*

The aggregation process takes as input a *map* $\psi : [d] \mapsto \{\perp\} \cup \{2, 3, 4, \ldots\}$. Note that if $\psi(r) \neq \perp$, then $\psi(r) > 1$. Informally, $\psi(r)$, when not equating to $\perp$, tells us the level of T_r whose hard function is to be included in the aggregation. We define $\Psi^{-1} := \{r | \psi(r) \neq \perp\}$, the subset of relevant dimensions. Given the map ψ, we aggregate the collection of 1D functions $\{g_{\psi(r)}^r | r \in \Psi^{-1}\}$ into a single hard function for $[n]^d$ as follows.

$$\gg(x) := \sum_{r \in \Psi^{-1}} (2n+1)^r g_{\psi(r)}^r(x_r) + \sum_{r \notin \Psi^{-1}} 2(2n+1)^r x_r \tag{3.14}$$

Observe that the latter sum is identical to the corresponding portion in $\mathsf{val}(x)$. The first summand takes the hard function corresponding to the $\psi(r)$th level of T_r for $r \in \Psi^{-1}$ and aggregates them via multiplying them with a suitable power of $(2n + 1)$.

Definition 3.7.19. *A map ψ is* **useful** *if the following are true.*

- $\sum_{r \in \Psi^{-1}} \mu_{\geq \psi(r)}^r \in (\varepsilon', 1)$
- *For all $r \in \Psi^{-1}$, $\mu_{\geq \psi(r)}^r \geq \frac{\mu_{\geq \psi(r)-1}^r}{2}$.*

In plain English, the first point states that total distance of the hard functions picked should be at least ϵ'. The second point is a technicality which is required to argue about the distance of the aggregated function. It states that in each relevant T_r, the total mass on the nodes lying in the $\psi(r)$th layer and below shouldn't be much smaller than the total mass on the nodes lying on the $(\psi(r)-1)$th layer and below.

Lemma 3.7.20. *If ψ is useful, $\mathsf{dist}_{\mathcal{D}}(\gg, \mathtt{MON}) \geq \varepsilon$.*

Proof. It is convenient to consider restrictions of $\gg$ where all coordinates in $[d] \setminus \Psi^{-1}$ are fixed. This gives rise to $|\Psi^{-1}|$-dimensional functions. We argue that each such restriction is ε-far from monotone, which proves the lemma. Abusing notation, we use $\gg$ to refer to an arbitrary such restriction. Fix some $r \in \Psi^{-1}$. Define the subset $S_r := \{x \in [n]^d : x_s \in L_{<\psi(s)-1}^s, \forall s \neq r\}$ to be the set of points x with the sth coordinate appearing in the first $(\psi(s)-2)$ layers of the tree T_s, for all $s \neq r$. We stress that this is well-defined because $\psi(s) \geq 2$ by definition of ψ. Note that each S_r is a collection of r-lines and the restriction of $\gg$ on each line exactly a multiple of $g^r_{\psi(r)}$. By Lemma 3.7.18, all violations to monotonicity in such lines lie in the intervals corresponding to $L^r_{\geq\psi(r)-1}$, and the mass of the vertex cover of the violation graph (restricted to the line) is at least $\mu^r_{\geq\psi(r)}/2$. Thus the total contribution to distance of $\gg$ from S_r is at least $\frac{\mu^r_{\geq\psi(r)}}{2} \cdot \mu_{\mathcal{D}_{-r}}(\prod_{s\neq r} L^s_{<\psi(s)-1})$. What is crucial to note is that the regions of violations in S_r is *disjoint* from the regions of violation in $S_{r'}$ for $r' \neq r$. Therefore, the contributions to the distance of $\gg$ add up, and this gives

$$\begin{aligned}
\mathsf{dist}_{\mathcal{D}}(\gg, \texttt{MON}) &\geq \frac{1}{2}\sum_{r\in\Psi^{-1}} \mu^r_{\geq\psi(r)} \cdot \mu_{\mathcal{D}_{-r}}\Big(\prod_{s\neq r} L^s_{<\psi(s)-1}\Big) \\
&= \frac{1}{2}\sum_{r\in\Psi^{-1}} \mu^r_{\geq\psi(r)} \prod_{s\neq r}(1-\mu^s_{\geq\psi(s)-1}) \\
&\geq \frac{1}{2}\sum_{r\in\Psi^{-1}} \mu^r_{\geq\psi(r)} \prod_{s\neq r}(1-2\mu^s_{\geq\psi(s)}) \quad \text{(point 2 in def. of useful map)}
\end{aligned}$$

We can apply the bound, $\sum_{r\in\Psi^{-1}} \mu^r_{\geq\psi(r)} \in (\varepsilon', 1)$, since ψ is useful. We lower bound the product by $\exp(-4\sum_{s\neq r}\mu^s_{\geq\psi(s)})$, which by the above bound, is at least e^{-4}. So, $\mathsf{dist}_{\mathcal{D}}(\gg, \texttt{MON}) \geq \sum_{r\in\Psi^{-1}} \mu^r_{\geq\psi(r)}/120 \geq \varepsilon'/120 = \epsilon$. ☐

Definition 3.7.21. *Two maps ψ_1, ψ_2 are* disjoint *if: $\{(r,\psi_1(r)) | r \in \Psi_1^{-1}\}$ and $\{(r,\psi_2(r)) | r \in \Psi_2^{-1}\}$ are disjoint.*

That is, for every tree T_r, ψ_1 and ψ_2 point to different layers of the tree (or they point to $\perp$).

Lemma 3.7.22. *Consider a set of maps $\psi_1, \psi_2, \ldots$ that are all pairwise disjoint. A set of Q queries can distinguish at most $|Q|-1$ of these functions from* val.

Proof. Say a pair (x,y) of queries captures the (unique) tuple (r,j) if the largest coordinate in which x and y differ is r, and furthermore $\mathsf{lca}(x_r, y_r)$ in T_r lies in level $(j-1)$. A set Q captures (r,j) if some pair in Q captures (r,j). We first show that if (x,y) distinguishes $\gg$ from val for some map ψ, then (x,y) captures a pair $(r,\psi(r))$ for some $r \in \Psi^{-1}$. Assume wlog $\mathsf{val}(x) < \mathsf{val}(y)$, and so $\gg(x) >\gg (y)$. Let a be the largest coordinate at which x and y differ; since $\mathsf{val}(x) < \mathsf{val}(y)$, we get $x_a < y_a$. Suppose $g^a_{\psi(a)}(x_a)$ and $g^a_{\psi(a)}(y_a)$ is not a violation. By the construction, this implies that $g^a_{\psi(a)}(y_a) - g^a_{\psi(a)}(x_a) \geq 1$. Furthermore, $g^r_{\psi(r)}$ is always in the range $[1, 2n]$ for any r.

$$\gg(y) - \gg(x) \;=\; (2n+1)^a(g^a_{\psi(a)}(y_a) - g^a_{\psi(a)}(x_a)) + \sum_{r<a, r\in\Psi^{-1}} (2n+1)^r(g^r_{\psi(r)}(y_r) - g^r_{\psi(r)}(x_r))$$

$$\begin{aligned} &\geq (2n+1)^a - (2n)\sum_{r<a}(2n+1)^r \\ &= (2n+1)^a - (2n)\cdot\frac{(2n+1)^a - 1}{2n} \quad > 0, \end{aligned}$$

So $(g^a_{\psi(a)}(x_a), g^a_{\psi(a)}(y_a))$ is a violation. Immediately, we deduce that $\psi(a) \neq \bot$, so $a \in \Psi^{-1}$. By Claim 3.7.8, $\mathsf{lca}(x_a, y_a)$ lies in level $\psi(a) - 1$ of T_a, and hence, (x, y) captures $(a, \psi(a))$. As we prove in Claim 3.7.23, Q queries can capture at most $|Q| - 1$ such tuples. The proof is completed by noting the maps $\psi_1, \psi_2, \ldots$ are pairwise disjoint. $\square$

Claim 3.7.23. *A nonempty set Q can only capture at most $|Q| - 1$ tuples (r, j).*

Proof. Proof is by induction on $|Q|$. If $|Q| = 2$, then the claim trivially holds. Assume $|Q| > 2$. Let s be the largest dimension such that there are at least two points in Q differing in that dimension. For $c = 1$ to n, let $Q_c := \{x \in Q : x_s = c\}$. By definition, $Q_c \subset Q$. Reorder the dimensions such that Q_c is non-empty for $c = 1 \ldots q \leq n$. By induction, each Q_c captures at most $|Q_c| - 1$ pairs for $1 \leq c \leq q$. Consider (x, y) with $x \in Q_c$ and $y \in Q_{c'}$ for $c \neq c'$. The largest coordinate where they differ is exactly s. All tuples captured by such pairs is of the form (s, ℓ), where ℓ is the lca in T_s of some $c, c' \in \{1 \ldots, q\}$. By Claim 3.7.9, the total number of such points is at most $q - 1$. Thus, the total number of tuples captured is at most $\sum_{a=1}^{q} |Q_a| - q + (q - 1) = |Q| - 1$. $\square$

Now we are ready to construct the hard functions. Let us go back to the framework of Theorem 3.7.5. From Lemma 3.7.20 and Lemma 3.7.22, it suffices to construct a sequence $\psi_1, \psi_2, \ldots$ of pairwise disjoint, useful maps. The number of such maps will exactly be our lower bound. The exact construction is a little tricky, since the conditions of usefulness are somewhat cumbersone. We use the following definition.

- **Allowed levels:** A level j is allowed w.r.t. dimension r if $j > 1$ and $\mu^r_{\geq j} \geq \mu^r_{\geq j-1}/2$. This is in lines with point 2 of the usefulness definition.
- **Level sets A_r:** A_r is the set of allowed levels of tree T_r.

It is convenient to define an abstract procedure that constructs these maps. We have a stack S_r for each $r \in [d]$, whose elements are allowed levels. The stack S_r is initialized with A_r in increasing order, that is the head (top entry) of the stack is the least (that is, closest to root) level in A_r. In each *round*, we will construct a map ψ. Denote the head of S_r by h_r. Note that $h_r > 1$ by definition of allowed levels. Maintain a running count initialized to 0. We go through the stacks in an arbitrary order popping off a *single* element from each stack. In a round, we never touch the same stack more than once. When we pop S_r, we set $\psi(r) := h_r$ and add $\mu^r_{\geq h_r}$ to the running count. We stop as soon as the running count enters the interval $[\varepsilon', 1]$. For all r for which $\psi(r)$ hasn't been defined, we set $\psi(r) = \bot$. This completes the description of a single map. Observe, by definition of allowed levels and the stopping condition, ψ is useful. When $\sum_{r=1}^{d} \mu^r_{\geq h_r} < \varepsilon'$, we cannot complete the construction. So the procedure terminates, discarding the final map. Let the set of maps constructed be $\mathbf{\Psi}$. By construction, the maps are useful. Furthermore, they are pairwise disjoint, because once a layer is popped out, it never appears again. We now basically show that $|\mathbf{\Psi}|$ is large, using the (ε', ρ)-stability of $\mathcal{D}$. This proves that the number of hard functions is large. We have to first deal with an annoying corner case of $\mathcal{D}$.

Theorem 3.7.24. *If* $\sum_r \mu^r_{\geq 1} > \rho\Delta(\mathcal{D})/12$, *then any monotonicity tester requires* $\Omega(\rho\Delta^*(\mathcal{D}))$ *queries.*

Proof. We simply apply the hypercube lower bound. Recall the definition of θ_r described before Theorem 3.7.15. Note that $\mu^r_{\geq 1}$ is simply the total $\mathcal{D}_r$-mass of everything in T_r other than the root. By the median property of the T_r, θ_r is ensured to be at least half of this mass, and hence $\theta_r \geq \mu^r_{\geq 1}/2$. Combining with Theorem 3.7.15, we get a lower bound of $\Omega(\sum_r \mu^r_{\geq 1})$, which by assumption, is $\Omega(\rho\Delta(\mathcal{D}))$. An application of Lemma 3.7.17 completes the proof. □

Now we come to the main bound of $|\mathbf{\Psi}|$. We need some setup for the proof. The following simple observation is crucial. This follows since $\mathbf{E}[Z] = \sum_{k\in\mathbb{N}} \Pr[Z \geq k]$, for any non-negative, integer valued random variable.

Claim 3.7.25. *For all* r, $\sum_{j\geq 1} \mu^r_{\geq j} = \mathbf{E}_{x\sim\mathcal{D}_r}[\mathsf{depth}_{T_r}(x)] = \Delta_r$.

The following lemma completes the entire lower bound.

Lemma 3.7.26. *Suppose* $\sum_r \mu^r_{\geq 1} \leq \rho\Delta(\mathcal{D})/12$. *Then* $|\mathbf{\Psi}| = \Omega(\rho\Delta(\mathcal{D}))$.

Proof. Let h_r denote the head of S_r when the procedure terminates. So, $\sum_{r=1}^{d} \mu^r_{\geq h_r} < \varepsilon'$. For any $\psi \in \mathbf{\Psi}$, $\sum_{r\in\Psi^{-1}} \mu^r_{\geq \psi(r)} \leq 1$. Hence, $|\mathbf{\Psi}|$ is at least the total sum over popped elements $\mu^r_{\geq j}$. Writing this out and expanding out a summation,

$$|\mathbf{\Psi}| \geq \sum_{r\in[d]} \sum_{j<h_r, j\in A_r} \mu^r_{\geq j} = \sum_{r\in[d]} \left[\sum_{j=1}^{h_r-1} \mu^r_{\geq j} - \mu^r_{\geq 1} - \sum_{1<j<h_r : j\notin A_r} \mu^r_{\geq j} \right]$$

Recall that $h_r > 1$ and so the summations are well-defined. For any level $1 < j \notin A_r$, we have $\mu^r_{\geq j} < \mu^r_{\geq j-1}/2$. Therefore, $\sum_{1<j<h_r : j\notin A_r} \mu^r_{\geq j} < \sum_{j=1}^{h_r-2} \mu^r_{\geq j}/2$. Plugging this bound in and applying the lemma assumption,

$$|\mathbf{\Psi}| \geq \sum_{r\in[d]} \sum_{j=1}^{h_r-1} \mu^r_{\geq j}/2 - \sum_{r\in[d]} \mu^r_{\geq 1} \geq \sum_{r\in[d]} \sum_{j=1}^{h_r-1} \mu^r_{\geq j}/2 - \rho\Delta(\mathcal{D})/12 \tag{3.15}$$

We need to lower bound the double summation above. Observe that the second summation is $\sum_{j\geq 1}\mu^r_{\geq j} - (\mu^r_{\geq h_r} + \mu^r_{\geq h_r+1} + \cdots)$. The first term, by Claim 3.7.25 is precisely $\Delta(\mathcal{D})$, and by definition of h_r, each of the terms in the parenthesis is at most ε'. However, the number of terms in the parenthesis can be quite large, and this doesn't seem to get any lower bound on the summation. Here's where stability of $\mathcal{D}$ saves the day. Construct a distribution $\mathcal{D}'_r$ be the distribution on $[n]$ as follows. Move the entire probability mass away from $L^r_{\geq h_r}$ and distribute it on the ancestral nodes in level $(h_r - 1)$ of T_r. More precisely, $\mu_{\mathcal{D}'_r}(u) = 0$ if $u \in L^r_{\geq h_r}$, $\mu_{\mathcal{D}'_r}(u) = \mu_{\mathcal{D}_r}(u)$ if $u \in L^r_{<h_r-1}$, and $\mu_{\mathcal{D}'_r}(u) = \sum_v \mu_{\mathcal{D}_r}(v)$ for $u \in L^r_{h_r-1}$ where the summation is over children v of u in T_r. Letting $\mathcal{D}' := \prod_r \mathcal{D}'_i$, we see that $||\mathcal{D} - \mathcal{D}'||_{\mathsf{TV}} \leq \sum_{r=1}^d \mu^r_{\geq h_r} < \varepsilon'$. Since $\mathcal{D}$ is (ε', ρ)-stable, we get $\Delta^*(\mathcal{D}') \geq \rho\Delta^*(\mathcal{D})$. Now we can apply Claim 3.7.25 on $\mathcal{D}'_r$ and T_r to get $\mathbf{E}_{x\sim\mathcal{D}'_r}[\mathsf{depth}_{T_r}(x)] = \sum_{j\geq 1}\mu_{\mathcal{D}'_r}(L^r_{\geq j}) = \sum_{j=1}^{h_r-1}\mu^r_{\geq j}$. This expected depth is by definition at least $\Delta^*(\mathcal{D}'_r)$. Therefore, we get a lower bound of $\sum_{r=1}^d \Delta^*(\mathcal{D}'_r)/2 = \Delta^*(\mathcal{D}')/2$ on the double summation in Eq. (3.15). Using the stability of $\mathcal{D}'$ this is at least $\rho\Delta^*(\mathcal{D})/2$. Substituting we get

$$\begin{aligned} |\mathbf{\Psi}| &\geq \rho\Delta^*(\mathcal{D})/2 - \rho\Delta(\mathcal{D})/12 \\ &\geq \rho\Delta^*(\mathcal{D})/2 - 5\rho\Delta^*(\mathcal{D})/12 = \Omega(\rho\Delta^*(\mathcal{D})) \qquad \text{(by Lemma 3.7.17)} \end{aligned}$$

□

We put it all together to prove the main lower bound, Theorem 3.7.16. If $\sum_r \mu^r_{\geq 1} > \rho\Delta(\mathcal{D})/12$, Theorem 3.7.24 proves Theorem 3.7.16. Otherwise, by Lemma 3.7.26 we have constructed $\Omega(\rho\Delta^*(\mathcal{D}))$ pairwise disjoint, useful maps. Each map yields a hard function of distance at least ε (by Lemma 3.7.20), and these functions satisfy the conditions of Theorem 3.7.5, which implies Theorem 3.7.16.

Chapter 4

Erasure-Resilient Property Testing

4.1 Introduction

In this chapter, we revisit the question of how sublinear-time algorithms access their data. With very few exceptions, all algorithms studied in the literature on sublinear-time algorithms have *oracle* access to data[1]. However, in many applications, this assumption is unrealistic. The oracle may be unable to reveal parts of the data due to privacy concerns, or when some of the values get erased by mistake or by an adversary. Motivated by these scenarios, we propose to study sublinear algorithms that work with partially erased data.

Formally, we view a dataset as a function over some discrete domain $\mathcal{D}$, such as $[n] = \{1, \ldots, n\}$ or $[n]^d$. For example, the classical problem of testing whether a list of n numbers is sorted in nondecreasing order can be viewed as a problem of testing whether a function $f : [n] \to \mathbb{R}$ is monotone (nondecreasing). Given a parameter $\alpha \in (0, 1)$, we say that a function is α-*erased* if at most an α fraction of its domain points are marked as "erased" or protected (that is, an algorithm is denied access to these values). An algorithm that takes an α-erased function as its input does not know which values are erased until it queries the corresponding domain points. For each queried point x, the algorithm either learns $f(x)$ or, if x is an erased point, gets back a special symbol $\perp$. We study algorithms that work in the presence of *adversarial erasures*. In other words, the query complexity of an algorithm is the number of queries it makes in the *worst case* over all α-erased input functions. An algorithm is required to be correct only on the non-erased part of the domain. E.g., it should test whether the input function is monotone on that part of the domain.

In this work, we initiate a systematic study of property testers that are resilient to the presence

---

[1] Some known sublinear-time algorithms only require random labeled samples from the domain, and this is another type of access that has received some attention [GR15]. There is also a line of work, initiated by [BFR$^+$13], that studies sublinear algorithms that access distributions, as opposed to fixed datasets. In this work, we focus on fixed datasets.

of adversarial erasures. An α-erasure-resilient ε-tester is given parameters $\alpha, \varepsilon \in (0,1)$, along with oracle access to an α-erased function f. The tester has to accept with high probability if f, restricted to non-erased part of the domain, satisfies the desired property $\mathcal{P}$ and reject with high probability if it is ε-far from $\mathcal{P}$.

Erasure-resilient testers for more challenging properties. One challenge in designing erasure-resilient testers by using existing algorithms in the standard model as a starting point is that many existing algorithms are more likely to query certain points in the domain. Therefore, if these points are erased, the algorithms will break. Specifically, the optimal algorithms for testing whether a list of numbers is sorted (and there are at least three different algorithms for this problem [EKK+00, BGJ+12, CS13a]) have this feature. Moreover, it is known that an algorithm that makes uniformly random queries is far from optimal. (It will have to make $\Theta(\sqrt{n})$ queries instead of $\Theta(\log n)$ for n-element lists [EKK+00, Fis04].)

There is a number of well-studied properties for which all known optimal algorithms heavily rely on querying specific points. Most prominent examples include monotonicity, the Lipschitz properties and, more generally, bounded-derivative properties of real-valued functions on $[n]$ and $[n]^d$, as well as convexity of real-valued functions on $[n]$. It is especially challenging to deal with real-valued functions in our model, because there are many possibilities for erased values. We give efficient erasure-resilient testers for all aforementioned properties of real-valued functions.

4.1.1 The Erasure-Resilient Testing Model

We formalize our erasure-resilient model for the case of property testing [RS96, GGR98]. Erasure-resilient versions of other computational models, such as tolerant testing [PRR06a], can be defined analogously.

Definition 4.1.1 (α-erased function)**.** *Let $\mathcal{D}$ be a domain, $\mathcal{R}$ be a range, and $\alpha \in (0,1)$. A function*[2] *$f : \mathcal{D} \mapsto \mathcal{R} \cup \{\perp\}$ is α-*erased *if f evaluates to $\perp$ on at most an α fraction of domain points. The points on which f evaluates to $\perp$ are called* erased. *The set of remaining (non-erased) points is denoted by $\mathcal{N}$.*

A function f is ε-far from a property (set) $\mathcal{P}$ if it needs to be changed on at least an ε fraction of domain points to obtain a function in $\mathcal{P}$. A function $f' : \mathcal{D} \to \mathcal{R}$ that differs from a function f only on points erased in f is called a *restoration* of f.

---

[2] Any object can be viewed as a function. E.g., an n-element array of real numbers can be viewed as a function $f : [n] \to \mathbb{R}$, an image—as a map from the plane to the set of colors, and a graph—as a map from the set of vertex pairs to $\{0,1\}$.

Definition 4.1.2 (Erasure-resilient tester)**.** *An* α-erasure-resilient ε-tester *of property* $\mathcal{P}$ *gets input parameters* $\alpha, \varepsilon \in (0,1)$ *and oracle access to an* α*-erased function* $f : \mathcal{D} \to \mathcal{R} \cup \{\perp\}$. *It outputs, with probability*[3] *at least 2/3,*

- **accept** *if there is a restoration* $f' : \mathcal{D} \to \mathcal{R}$ *of* f *that satisfies* $\mathcal{P}$*;*
- **reject** *if every restoration* $f' : \mathcal{D} \to \mathcal{R}$ *of* f *needs to be changed on at least an* ε *fraction of* $\mathcal{N}$*, the non-erased portion of* f*'s domain, in order to satisfy* $\mathcal{P}$ *(that is,* f' *is* $\varepsilon \cdot \frac{|\mathcal{N}|}{|\mathcal{D}|}$*-far from* $\mathcal{P}$*).*

The tester has 1-sided error *if the first item holds with probability 1.*

Let $f_{|\mathcal{N}}$ denote the function f restricted to the set $\mathcal{N}$ of non-erased points. Observe that we can define a property $\mathcal{P}'$ such that the erasure-resilient tester is simply required to distinguish the case that $f_{|\mathcal{N}}$ satisfies $\mathcal{P}'$ from the case that it is ε-far from satisfying it. (For example, if $\mathcal{P}$ is monotonicity of functions on a partially-ordered domain $\mathcal{D}$ then $\mathcal{P}'$ is monotonicity of functions on $\mathcal{N}$.) However, it is different from the standard property testing problem because the tester does not know in advance which points are erased.

4.1.2 Our Results

We give efficient erasure-resilient testers for all properties discussed in Section 2.1. All our testers have optimal complexity for the case with no erasures and have an additional benefit of not relying too heavily on the value of the input function at any specific point.

4.1.3 Monotonicity on the Line

We start by giving (in Section 4.2) an erasure-resilient monotonicity tester on $[n]$.

Theorem 4.1.3 (Monotonicity tester on the line)**.** *There exists a one-sided error* α*-erasure-resilient* ε*-tester for monotonicity of real-valued functions on the line* $[n]$ *that works for all* $\alpha, \varepsilon \in (0,1)$*, with query complexity* $O\left(\frac{\alpha}{1-\alpha} \cdot \frac{1}{\varepsilon^2} \log \frac{1}{\varepsilon} \cdot \log n\right)$. *Moreover, if* $\alpha = O(\varepsilon)$ *then the query complexity is* $O\left(\frac{1}{1-\alpha} \cdot \frac{1}{\varepsilon} \log \frac{1}{\varepsilon} \cdot \log n\right)$, *and if* $\alpha < \varepsilon$ *then it is* $O\left(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon} \cdot \log n\right)$.

---

[3] In general, the error probability can be any $\delta \in (0,1)$. For simplicity, we formulate our model and the results with $\delta = 1/3$. To get results for general δ, by standard arguments, it is enough to multiply the complexity of an algorithm by $\log 1/\delta$.

Without erasure resilience, the complexity of testing monotonicity of functions $f : [n] \mapsto \mathbb{R}$ is $\Theta(\log n/\varepsilon)$ [EKK$^+$00, Fis04]. Thus, the query complexity of our erasure-resilient tester has optimal dependence on the domain size and, for $\alpha < \varepsilon$, has nearly optimal dependence (up to the logarithmic factor) on ε.

The starting point of our algorithm is the tester for sortedness from [EKK$^+$00]. This tester picks a random element of the input array and performs a binary search for that element. It rejects if the binary search does not lead to the right position. The first challenge is that the tester always queries the middle element of the array and is very likely to query other elements that are close to the root in the binary search tree. So, it will break if these elements are erased. To make it resilient to erasures, we randomize the binary tree with respect to which it performs the binary search. The second challenge is that the tester does not know which points are erased. To counteract that, our tester samples sample points from appropriate intervals until it encounters a non-erased point. To analyze the tester, we use Devroye's bound [Dev86] on the depth of a random binary search tree, as well as our own combinatorial lemma (Lemma 4.2.2) that bounds, for every binary search tree, the fraction of search paths that contain some intervals "dense" with erasures. This lemma might be of independent interest because it applies when density of any type of bad points, not necessarily erasures, needs to be limited.

4.1.4 BDPs on the Hypergrid

In Section 4.3, we generalize our monotonicity tester in two ways: (1) to work over general hypergrid domains, and (2) to apply to all BDPs. We achieve it by giving (1) a reduction from testing BDPs on the line to testing monotonicity on the line that applies to erasure-resilient testers and (2) an erasure-resilient version of the dimension reduction from [CDJS15]. We obtain the following result.

Theorem 4.1.4 (BDP tester on the hypergrid)**.** *For every BDP $\mathcal{P}$, there exists a one-sided error α-erasure-resilient ε-tester for $\mathcal{P}$ of real-valued functions on the hypergrid $[n]^d$ that works for all $\alpha, \varepsilon \in (0,1)$, where $\alpha \leq \varepsilon/161d$, with query complexity $O\left(\frac{d^2}{\varepsilon^2(1-\alpha)^2} \cdot \log \frac{d}{\varepsilon(1-\alpha)} \cdot H\right)$, where $H = \mathsf{max}\{\log n, \log \frac{d}{\varepsilon(1-\alpha)}\}$.*

4.1.5 Convexity on the Line

Finally, in Section 4.4, we develop additional techniques to design a tester for convexity (not a BDP) on the line. When α is small, the query complexity of our tester has optimal dependence on n and nearly optimal (up to the log factor) dependence on ε.

Theorem 4.1.5 (Convexity tester on the line)**.** *There exists a one-sided error α-erasure-resilient ε-tester for convexity of real-valued functions on the line $[n]$ that works for all $\alpha, \varepsilon \in (0, 1)$, with query complexity $O\left(\frac{\alpha}{1-\alpha} \cdot \frac{1}{\varepsilon^3} \log \frac{1}{\varepsilon} \cdot \log^2 n\right)$. Moreover, the query complexity simplifies to $O\left(\frac{1}{1-\alpha} \cdot \frac{1}{\varepsilon} \log \frac{1}{\varepsilon} \cdot \log n\right)$ when $\alpha = O\left(\frac{\varepsilon^2}{\log n}\right)$ and to $O\left(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon} \cdot \log n\right)$ when $\alpha < \frac{\varepsilon^2}{\log n}$.*

Our algorithm builds on the convexity tester by Parnas et al. [PRR06a] and our erasure-resilient monotonicity tester for the line. Our convexity tester for the case when no erased points are present is conceptually simpler than the original tester in [PRR06a]. To make it erasure-resilient, we pick a random (non-erased) point and do a binary search for it in a random binary search tree. However, instead of checking whether the selected point can be found, as in our monotonicity tester, the convexity tester checks a more complicated "goodness condition" in each visited interval of the binary search tree. It boils down to checking that the slope of the functions between pairs of carefully selected points satisfies the convexity condition.

The analysis of our convexity tester is more involved than for other properties. The further complication is that, in addition to sampling queries in each interval to catch non-erased points, we also need *walking queries* to find the next (or previous) non-erased point adjacent to a given point. We prove another combinatorial lemma (Lemma 4.4.1) that limits a number of points with a long run of adjacent erased points.

4.2 Erasure-Resilient Monotonicity Tester For the Line

In this section, we prove Theorem 4.1.3. Recall that, for a function $f : \mathcal{N} \mapsto \mathbb{R} \cup \{\perp\}$, the set of non-erased points (the ones that map to $\mathbb{R}$) is denoted by $\mathcal{N}$, and that, for $\mathcal{N} \subseteq [n]$, a function $f : \mathcal{N} \mapsto \mathbb{R}$ is monotone if $x < y$ implies $f(x) \leq f(y)$ for all $x, y \in \mathcal{N}$. Also recall that the tester does not know $\mathcal{N}$.

We first give a high level overview of the algorithm. The algorithm has oracle access to f and takes α and ε as inputs. In each iteration, it samples points uniformly at random from $[n]$ until it gets a non-erased point s, and then does a randomized binary search for s. It rejects if a violation to monotonicity is found in its search. It accepts if either no violation is found, or if the the number of queries exceed Q, where Q is a function of ε,α and n. In the description of our tester (Algorithm 1), we use $I[i, j]$ to denote the set of natural numbers from i until and including j. We alternatively refer to it as the interval from i to j.

Algorithm 1 Erasure-Resilient Monotonicity Tester for the Line

```
Input: parameters $\alpha, \varepsilon \in (0,1)$; oracle access to $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$
 1: Set $Q = c \cdot \log n$, where $c = \left\lceil \frac{14}{\varepsilon}\left(1 + \ln\left(\frac{28}{\varepsilon}\right)\right)\left(1 + \frac{28\alpha}{\varepsilon(1-\alpha)}\right)\right\rceil$.
 2: Accept at any point if the number of queries exceeds $Q$.
 3: repeat $2/\varepsilon$ times:
 4:     Sample points uniformly at random from $I[1,n]$ and query them until we get a point $s \in \mathcal{N}$.
 5:     Set $\ell \leftarrow 1, r \leftarrow n$.
 6:     while $\ell \leq r$ do
 7:         Sample and query points uniformly at random from $I[\ell, r]$ until we get a point $m \in \mathcal{N}$.
 8:         if $s < m$ then
 9:             Reject if $f(s) \geq f(m)$. Otherwise, set $r \leftarrow m - 1$.
10:         if $s > m$ then
11:             Reject if $f(s) \leq f(m)$. Otherwise, set $\ell \leftarrow m + 1$.
12:         if $s = m$ then
13:             Go to Step 3.                                        ▷ Search completed.
14: Accept.
```

4.2.1 Analysis Outline and Technical Ingredients

Every iteration of Algorithm 1 can be viewed as a traversal of a uniformly random search path in a uniformly random binary search tree defined on the set $\mathcal{N}$ of non-erased points. A search path is a path from the root to some node Γ of T. Given a binary search tree T over $\mathcal{N}$, we associate every node of T with a unique sub-interval I of $I[1,n]$ as follows. The root of T is associated with $I[1,n]$. Suppose the interval associated with a node Γ in T that contains $s \in \mathcal{N}$ is $I[i,j]$. Then the interval associated with the left child of Γ is $I[i, s-1]$ and the interval associated with the right child of Γ is $I[s+1, j]$.

If $f_{|\mathcal{N}}$ is ε-far from monotone, we prove that, with high probability, the tester finds a violation. It is easy to prove this, using a generalization of an argument from [EKK$^+$00], for the case when Algorithm 1 manages to complete all iterations of Step 3 before it runs out of queries.

The challenge is that the algorithm might get stuck pursuing long paths in a random search tree and waste many queries on erased points. Using a result by [Dev86], we deduce that, with high probability, the depth of a uniformly random binary search tree on $\mathcal{N}$ is $O(\log n)$. To resolve the issue of many possible queries to erased points, we prove a combinatorial lemma on the structure of binary search trees, which shows that all the intervals in most of the search paths in any binary search tree have *sufficiently many* non-erased points. We first define some terminology that we use to prove our combinatorial lemma.

Definition 4.2.1 (Dense and Sparse Intervals)**.** *The* density *of a interval $I \subseteq [1, n]$ is defined as the fraction of erased domain points in I. An interval is δ-dense if its density is at least δ, where $\delta \in (0, 1)$. Otherwise, the interval is δ-sparse.*

The following lemma bounds the fraction of search paths that contain some δ-dense intervals, for every binary search tree. Recall that, a search path in a search tree T is a path from the root to some node of T . We believe that the lemma is of independent interest.

Lemma 4.2.2 (Combinatorial lemma)**.** *Consider an arbitrary binary search tree T on $\mathcal{N}$. For all $\delta \in [0, 1)$, the fraction of search paths in T that contain some δ-dense intervals is at most*

$$\frac{\alpha}{1-\alpha} \cdot \frac{1-\delta}{\delta}.$$

Proof. A search path is called δ-heavy if it contains a δ-dense interval. Non-erased points in the leaves of δ-heavy paths are called δ-heavy points. Note that the number of δ-heavy paths and the number of δ-heavy points are equal. We will bound the fraction of δ-heavy paths in T.

Let the largest δ-dense interval on a δ-heavy path be called its primary δ-dense interval. Let $\mathcal{I}$ denote the set of primary δ-dense intervals corresponding to the δ-heavy paths of T. More than one δ-heavy path can have the same primary δ-dense interval. By definition, no interval in $\mathcal{I}$ can be an ancestor or descendant of another interval from $\mathcal{I}$ and hence the intervals in $\mathcal{I}$ are disjoint. Also, by definition, every δ-heavy non-erased point belongs to some interval in $\mathcal{I}$.

Let η denote the fraction of δ-heavy paths in T. Consider $I \in \mathcal{I}$. Since I is δ-dense, $\frac{|I \setminus \mathcal{N}|}{|I|} \geq \delta$. Putting this together with the fact that $|I| = |I \setminus \mathcal{N}| + |I \cap \mathcal{N}|$, we get $\frac{|I \setminus \mathcal{N}|}{|I \cap \mathcal{N}|} \geq \frac{\delta}{1-\delta}$. It follows that $\sum_{I \in \mathcal{I}} |I \setminus \mathcal{N}| \geq \frac{\delta}{1-\delta} \cdot \sum_{I \in \mathcal{I}} |I \cap \mathcal{N}|$. Since the intervals in $\mathcal{I}$ are disjoint, $\sum_{I \in \mathcal{I}} |I \setminus \mathcal{N}|$ is at most the number of erased domain points in the domain and $\sum_{I \in \mathcal{I}} |I_{\mathcal{N}}|$ is equal to the number of δ-heavy non-erased domain points, which is $\eta \cdot |\mathcal{N}|$. Putting all of this together,

$$n\alpha \geq \sum_{I \in \mathcal{I}} |I \setminus \mathcal{N}| \geq \frac{\delta}{1-\delta} \cdot \sum_{I \in \mathcal{I}} |I \cap \mathcal{N}| \geq \frac{\delta}{1-\delta} \cdot \eta \cdot |\mathcal{N}| \geq \frac{\delta}{1-\delta} \cdot \eta \cdot n(1-\alpha).$$

The last inequality uses the fact that the number of non-erased points in the whole instance is at least $n(1-\alpha)$. This implies that, $n(1-\alpha)\delta\eta \leq n\alpha(1-\delta)$. Solving for η, the statement of the lemma follows. $\square$

4.2.2 Analysis of the Tester

Now we analyze the tester. The query complexity of the tester is clear from its description. The main statement of Theorem 4.1.3 follows from Lemma 4.2.3, proved next. The "moreover" part of the theorem follows by substituting appropriate values of parameters in the query bound in Algorithm 1.

Lemma 4.2.3. *Algorithm 1 accepts if $f_{|\mathcal{N}}$ is monotone, and rejects with probability at least $2/3$ if $f_{|\mathcal{N}}$ is ε-far from monotone.*

Proof. The tester accepts whenever $f_{|\mathcal{N}}$ is monotone. To prove the other part of the lemma, assume that $f_{|\mathcal{N}}$ is ε-far from monotone. Let A be the event that the tester accepts f. Let q denote the total number of queries made. We prove that $\Pr[A] \leq 1/3$. The event A occurs if either $q > Q$ or the tester does not find a violation in any of the $\frac{2}{\varepsilon}$ iterations of Step 3. Thus,

$$\Pr\left[A\right] \leq \Pr\left[A|q \leq Q\right] + \Pr\left[q > Q\right].$$

First, we bound the probability that the tester does not find violations in one iteration of Step 3 conditioned on the event that $q \leq Q$. Each iteration of the tester can be viewed as a traversal of a uniformly random search path P in a uniformly random binary search tree defined over $\mathcal{N}$. Consider an arbitrary binary search tree T defined over points in $\mathcal{N}$. A point $s \in \mathcal{N}$ is called *searchable* with respect to T if Algorithm 1 does not detect a violation to monotonicity while traversing the search path to s in T. Consider two indices $i, j \in \mathcal{N}$, where $i < j$, both searchable with respect to T. Let $a \in \mathcal{N}$ be the pivot corresponding to the lowest common ancestor of the leaves containing i and j. Since i and j are both *searchable*, it must be the case that $f(i) < f(a)$ and $f(a) < f(j)$ and hence, $f(i) < f(j)$. Thus, for every tree T, the function restricted to the domain points that are *searchable* with respect to T is monotone. Therefore, if $f_{|\mathcal{N}}$ is ε-far from monotone, for every binary search tree T, at least an ε-fraction of the points in $\mathcal{N}$ are not *searchable*. Thus, the tester detects a violation with probability ε in each iteration. Consequently,

$$\Pr\left[A|q \leq Q\right] \leq (1-\varepsilon)^{\frac{2}{\varepsilon}}.$$

In the rest of the proof, we will bound $\Pr[q > Q]$. We will do this by proving that, with high probability, in each iteration the number of queries made by the tester is at most $Q/(\frac{2}{\varepsilon})$. Let us first show that the length of a root to leaf path in a random binary search tree is $O(\log n)$ with high probability.

Claim 4.2.4 ([Dev86], Lemma 3.1)**.** *Fix $n \geq 2$. Let H_n be the depth of a random binary search tree over n points. Then, for any positive integer $H \geq \log n$,*

$$\Pr\left[H_n \geq H\right] \leq \frac{1}{n}\left(\frac{2e\log n}{H}\right)^{H}.$$

Corollary 4.2.5. *The number of non-erased domain points in a random binary search path traversed by the algorithm is at most $7\log n$ with probability at least $1 - 1/n^2$.*

Next, we show using Lemma 4.2.2 that, with high probability, the fraction of non-erased points in each interval encountered by the tester is large.

Claim 4.2.6. *If P is a search path traversed by the algorithm, with probability $1-\eta$, the density of every interval on P is at most $\left(1+\frac{(1-\alpha)\cdot\eta}{\alpha}\right)^{-1}$.*

Proof. Let $\delta = \left(1+\frac{(1-\alpha)\cdot\eta}{\alpha}\right)^{-1}$. The fraction of δ-heavy search paths with respect to any binary search tree T is at most $\frac{\alpha}{1-\alpha}\cdot\frac{1-\delta}{\delta}$ from Lemma 4.2.2, which is equal to η when $\delta = \left(1+\frac{(1-\alpha)\cdot\eta}{\alpha}\right)^{-1}$. Since the algorithm traverses a uniformly random search path in each iteration, the lemma follows. □

Next, we prove that, with high probability, the total number of queries made by the algorithm to traverse a random search path is small, if the number of intervals on the path is bounded and the densities of those intervals are small. We use a tail bound from [Jan14] on the sum of geometric random variables to do this.

Claim 4.2.7 ([Jan14], Corollary 2.4)**.** *Let $X = \sum_{i=1}^{k} X_i$, where $k \geq 1$ and X_i, for $i \in [k]$, are independent geometric random variables. That is, X_i is the number of tosses until the first head when tossing a coin with heads probability p_i, where $0 < p_i \leq 1$ for all $i \in [k]$. Then, for any $\lambda \geq 1$,*

$$\Pr\left[X \geq \lambda \cdot \mathbf{E}\left[X\right]\right] \leq e^{1-\lambda}.$$

Corollary 4.2.8. *Consider a random search path P traversed by the algorithm. Let H denote the length of P, and suppose that the density of each interval encountered by the algorithm along P is at most δ_P. Then the total number of queries made by the algorithm to traverse P is at most $\frac{(1+\ln 1/\eta)\cdot H}{1-\delta_P}$ with probability at least $1-\eta$.*

Proof. Let $v_1, v_2, \ldots, v_H$ denote the non-erased domain points sampled while traversing P in the order in which they were sampled. Let X_1 denote the number of queries made until v_1 was sampled from $[n]$. For $1 < i \leq H$, let X_i denote the number of queries made to sample v_i after v_{i-1} was sampled. The total number of queries made by the algorithm to traverse P is then $X = \sum_{i=1}^{H} X_i$. Clearly, each X_i is a geometric random variable with parameter p_i for $i \in [H]$, where $1 - p_i$ is the density of the interval from which v_i was sampled. Since the density of every interval on P is at most δ_P, we have $\min_i\{p_i\} \geq 1 - \delta_P$. Therefore,

$$\mathbf{E}[X] = \sum_{i=1}^{H} \frac{1}{p_i} \leq \frac{H}{1-\delta_P}.$$

Hence, by Claim 4.2.7, for all $\lambda \geq 1$,

$$\Pr\left[X \geq \lambda \cdot \frac{H}{1-\delta_P}\right] \leq \Pr[X \geq \lambda \cdot \mathbf{E}[X]] \leq e^{1-\lambda}.$$

Setting $\eta = e^{1-\lambda}$, we get the corollary. □

We now prove an upper bound on $\Pr[q > Q]$, where q is the total number of queries made by Algorithm 1. Let $\delta = \left(1 + \eta\frac{(1-\alpha)}{\alpha}\right)^{-1}$, $H = 7\log n$ and $\eta = \varepsilon/28$. Let q_i denote the number of queries made in iteration i, respectively, for $1 \leq i \leq \left\lceil \frac{2}{\varepsilon} \right\rceil$.

By a union bound,

$$\Pr\left[q > Q\right] \leq \sum_{i=1}^{\left\lceil \frac{2}{\varepsilon} \right\rceil} \Pr\left[q_i > Q/\left(\frac{2}{\varepsilon}\right)\right].$$

Let G_i denote the (good) event that the length of P_i is at most H and that the density of each interval in P_i is at most δ. Let $|P_i|$ stand for the length of the path P_i. For each iteration i,

$$\begin{aligned}\Pr\left[q_i > \frac{Q\varepsilon}{2}\right] &\leq \Pr\left[q_i > \frac{Q\varepsilon}{2} | G_i\right] + \Pr\left[\overline{G_i}\right] \leq \Pr\left[q_i > \frac{H \cdot (1 + \ln(1/\eta))}{(1-\delta)} | G_i\right] + \Pr\left[\overline{G_i}\right] \\ &\leq \eta + \eta + \frac{1}{n^2}\end{aligned}$$

where the third inequality uses Corollary 4.2.8, Claim 4.2.6 and Corollary 4.2.5. Since $\eta = \varepsilon/28$, the error probability in this case becomes $\varepsilon/14 + 1/n^2 \leq \varepsilon/12$. Consequently, $\Pr\left[q > Q\right] \leq \frac{2}{\varepsilon} \cdot \frac{\varepsilon}{12} = 1/6$. Thus, $\Pr\left[A\right] \leq (1-\varepsilon)^{\frac{2}{\varepsilon}} + 1/6 \leq 1/3$. □

4.3 Erasure-Resilient BDP Testers For the Hypergrid

In this section, we present our erasure-resilient testers for all bounded derivative properties over hypergrid domains (see Definition 2.0.1) and prove Theorem 4.1.4. First, we show in Lemma 4.3.4

that testing of any BDP on $[n]$ reduces to testing monotonicity on $[n]$. A consequence of this reduction is Theorem 4.3.5. Next, we prove Lemma 4.3.7 and Lemma 4.3.8 that reduces the problem of erasure-resilient testing of a BDP over hypergrid domains to testing of the same property over the line.

4.3.1 Erasure-Resilient BDP Tester For the Line

In Lemma 4.3.4, we show that (erasure-resilient) testing of bounded derivative properties (BDPs) on the line reduces to monotonicity testing on the line and prove Theorem 4.3.5. As noted in Section 2.1, BDPs comprise of a large class of properties that have been studied in the property testing literature.

Given a function $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$, and a bounded derivative property $\mathcal{P}$, we first define the notion of a violated pair in f with respect to $\mathcal{P}$.

Definition 4.3.1 (Violated pair)**.** *Given a function $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$ and bounding family $\mathbf{B}$ consisting of functions $l, u : [n-1] \mapsto \mathbb{R}$, two points $x, y \in \mathcal{N}$ such that $x < y$* violate *the property $\mathcal{P}(\mathbf{B})$ with respect to f if $f(x) - f(y) > \mathfrak{m}_{\mathbf{B}}(x, y) = -\sum_{t=x}^{y-1} l(t)$ or $f(y) - f(x) > \mathfrak{m}_{\mathbf{B}}(y, x) = \sum_{t=x}^{y-1} u(t)$. The pairs (x, y) and (y, x) are called* violated.

Consider a bounded derivative property $\mathcal{P}$ of functions defined over $[n]$ and associated bounding functions $l, u : [n-1] \mapsto \mathbb{R}$. The following claim states that, we may assume w.l.o.g. that $l(i) = -u(i)$ for all $i \in [n-1]$. We use it in the proof of Claim 4.3.3.

Claim 4.3.2. *Consider a function $f : [n] \to \mathbb{R} \cup \{\perp\}$ and a bounding function family $\mathbf{B}$ over $[n]$ with $l, u : [n-1] \mapsto \mathbb{R}$. Let $g : [n] \mapsto \mathbb{R} \cup \{\perp\}$ be a function that takes the value $f(i) + \sum_{j=i}^{n-1} \frac{l(j)+u(j)}{2}$ for each $i \in \mathcal{N}$ and is erased on the remaining points. Let $\mathbf{B}'$ be a bounding function family over $[n]$ with $l', u' : [n-1] \mapsto \mathbb{R}$ such that $u'(i) = -l'(i) = \frac{u(i)-l(i)}{2}$ for all $i \in [n-1]$. Then $x, y \in \mathcal{N}$ violate $\mathcal{P}(\mathbf{B})$ with respect to f iff x, y violate $\mathcal{P}(\mathbf{B}')$ with respect to g.*

Proof. Note that $(x, y) \in \mathcal{N}$, where $x < y$, is not violated with respect to f if and only if $\max\{f(x) - f(y) - \mathfrak{m}_{\mathbf{B}}(x, y), f(y) - f(x) - \mathfrak{m}_{\mathbf{B}}(y, x)\} \leq 0$. We have

$$\begin{aligned} g(x) - g(y) - \mathfrak{m}_{\mathbf{B}'}(x, y) &= f(x) - f(y) + \sum_{i=x}^{y-1} \frac{u(i) + l(i)}{2} - \sum_{i=x}^{y-1} \frac{u(i) - l(i)}{2} \\ &= f(x) - f(y) - \sum_{i=x}^{y-1} l(i) = f(x) - f(y) - \mathfrak{m}_{\mathbf{B}}(x, y). \end{aligned}$$

Also,

$$\begin{aligned} g(y) - g(x) - \mathfrak{m}_{\mathbf{B}'}(y,x) &= f(y) - f(x) - \sum_{i=x}^{y-1} \frac{u(i)+l(i)}{2} - \sum_{i=x}^{y-1} \frac{u(i)-l(i)}{2} \\ &= f(y) - f(x) - \sum_{i=x}^{y-1} u(i) = f(y) - f(x) - \mathfrak{m}_{\mathbf{B}}(y,x). \end{aligned}$$

Thus, $\max\{g(x)-g(y)-\mathfrak{m}_{\mathbf{B}'}(x,y), g(y)-g(x)-\mathfrak{m}_{\mathbf{B}'}(y,x)\} = \max\{f(x)-f(y)-\mathfrak{m}_{\mathbf{B}}(x,y), f(y)-f(x)-\mathfrak{m}_{\mathbf{B}}(y,x)\}$. The claim follows. ☐

The following claim shows a reduction from testing BDPs over $[n]$ to testing monotonicity over $[n]$.

Claim 4.3.3. *Consider an α-erased function $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$ and bounding functions $l, u : [n-1] \mapsto \mathbb{R}$ such that $-l(i) = u(i) = \gamma(i)$ for all $i \in [n-1]$. Let $\mathcal{P}$ be the BDP defined by l and u. Let $g, h : [n] \mapsto \mathbb{R} \cup \{\perp\}$ be two functions that take the values $g(i) = f(i) - \sum_{r=i}^{n-1} \gamma(r)$ and $h(i) = -f(i) - \sum_{r=i}^{n-1} \gamma(r)$ for all $i \in \mathcal{N}$ and are erased on the remaining points. Then, the following conditions hold:*

1. *$x, y \in \mathcal{N}$ violate $\mathcal{P}$ with respect to f iff x, y violate monotonicity with respect to either g or h.*
2. *If f is in $\mathcal{P}$, then both g and h are both monotone.*
3. *If f is ε-far from $\mathcal{P}$, then either g or h is at least $\varepsilon/4$-far from monotonicity.*

Proof. Consider a pair $(i,j) \in \mathcal{N} \times \mathcal{N}$ where $i < j$. We have,

$$g(i) - g(j) = f(i) - f(j) - \sum_{r=i}^{j-1} \gamma(r) \qquad \text{and} \qquad h(i) - h(j) = f(j) - f(i) - \sum_{r=i}^{j-1} \gamma(r).$$

If (i,j) is not violated with respect to f, we have $f(j) - f(i) - \sum_{r=i}^{j-1} \gamma(r) \leq 0$ and $f(i) - f(j) - \sum_{r=i}^{j-1} \gamma(r) \leq 0$. Thus, (i,j) satisfies the monotonicity property with respect to g and h. If (i,j) is violated with respect to f, then either $f(j) - f(i) - \sum_{r=i}^{j-1} \gamma(r) > 0$ or $f(i) - f(j) - \sum_{r=i}^{j-1} \gamma(r) > 0$. That is, either g or h violates monotonicity.

Define the violation graph G_f as follows. The vertex set corresponds to $\mathcal{N}$. There is an (undirected) edge between $i \in \mathcal{N}$ and $j \in \mathcal{N}$ iff the pair (i,j) violates the property $\mathcal{P}$. By Lemma 2.5 in [CDJS15], the size of every maximal matching is at least $\varepsilon \cdot |\mathcal{N}|/2$. Consider a maximal matching M in G_f. From the discussion above, every edge in M violates monotonicity with respect to either g or h. Therefore, at least $\varepsilon \cdot |\mathcal{N}|/4$ edges are violated with respect to at least one of g and

h. Assume w.l.o.g. that at least $\varepsilon \cdot |\mathcal{N}|/4$ edges from M are violated with respect to h. One has to change the function value of at least one endpoint of each edge to repair it. Since M is a matching in the violation graph G_h as well, at least $\varepsilon \cdot |\mathcal{N}|/4$ function values of h have to change to make h monotone. This means that h is at least $\varepsilon/4$-far from monotone. ☐

Therefore, in order to test the bounded derivative property $\mathcal{P}$ on f with proximity parameter ε, one can test monotonicity on g and h with proximity parameter $\varepsilon/4$ and error probability 1/6 and accept iff both tests accept.

Lemma 4.3.4. *Let $Q_{\texttt{mon}}(\alpha, \varepsilon, n)$ denote the query complexity of α-erasure-resilient ε-testing of monotonicity of real-valued functions on the line. Then, for every BDP, α-erasure-resilient ε-testing of real-valued functions on the line has query complexity $O(Q_{\texttt{mon}}(\alpha, \varepsilon/4, n))$. The same statement holds for 1-sided error testing.*

The following theorem is a direct consequence of Lemma 4.3.4 and Theorem 4.1.3.

Theorem 4.3.5 (BDP tester on the line)**.** *For every BDP $\mathcal{P}$, there exists a one-sided error α-erasure-resilient ε-tester for $\mathcal{P}$ of real-valued functions on the line that works for all $\alpha, \varepsilon \in (0, 1)$, with query complexity $O\left(\frac{\alpha}{1-\alpha} \cdot \frac{1}{\varepsilon^2} \log \frac{1}{\varepsilon} \cdot \log n\right)$. Moreover, if $\alpha = O(\varepsilon)$ then the query complexity is $O\left(\frac{1}{1-\alpha} \cdot \frac{1}{\varepsilon} \log \frac{1}{\varepsilon} \cdot \log n\right)$, and if $\alpha < \varepsilon$ then it is $O\left(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon} \cdot \log n\right)$.*

4.3.2 Erasure-Resilient Dimension Reduction

Consider an α-erased function $f : [n]^d \mapsto \mathbb{R} \cup \{\perp\}$. Let $\mathcal{P}$ be a bounded derivative property of functions defined over $[n]^d$. Let $\mathcal{L}$ denote the set of all *axis-parallel lines* in $[n]^d$. In this section, we prove two important properties of a uniformly random axis parallel line in Lemma 4.3.7 and Lemma 4.3.8, which we jointly call erasure-resilient dimension reduction. We first introduce some notation.

Let $\mathcal{P}_{\mathcal{D}}$ denote the set of functions over a domain $\mathcal{D} \subseteq [n]^d$ satisfying $\mathcal{P}$. Let $\mathcal{P}^i_{\mathcal{D}}$ denote the set of functions over $\mathcal{D}$ with no violations to $\mathcal{P}$ along dimension i for all $i \in [d]$. We use $\mathcal{P}^i$ as shorthand for $\mathcal{P}^i_{[n]^d}$. Given a function $f : \mathcal{D} \mapsto \mathbb{R}$, the Hamming distance of f from $\mathcal{P}_{\mathcal{D}}$, denoted by $\text{dist}(f, \mathcal{P}_{\mathcal{D}})$, is the least number of points on which f needs to be changed to satisfy $\mathcal{P}_{\mathcal{D}}$. The relative Hamming distance between f and $\mathcal{P}_{\mathcal{D}}$ is $\text{dist}(f, \mathcal{P}_{\mathcal{D}})/|\mathcal{D}|$. Let $\ell \sim \mathcal{L}$ be a uniformly random axis-parallel line. Let $\mathcal{N}_\ell$ denote the set of non-erased points on ℓ and f_ℓ denote the function $f_{|\mathcal{N}}$ restricted to ℓ. Lemma 4.3.7 shows that the expected relative Hamming distance of f_ℓ from $\mathcal{P}_{\mathcal{N}_\ell}$ is roughly proportional to the relative Hamming distance of $f_{|\mathcal{N}}$ from $\mathcal{P}_{\mathcal{N}}$. First, we prove Claim 4.3.6 that we use in our proof of Lemma 4.3.7.

Claim 4.3.6.

$$\frac{1}{4}\text{dist}(f_{|\mathcal{N}}, \mathcal{P}_{\mathcal{N}}) \leq \sum_{i=1}^{d} \text{dist}(f_{|\mathcal{N}}, \mathcal{P}^i_{\mathcal{N}}) + \alpha \cdot d \cdot n^d.$$

Proof. Let $g : [n]^d \mapsto \mathbb{R}$ be a function in $\mathcal{P}$ such that $g_{|\mathcal{N}}$ is the closest function to $f_{|\mathcal{N}}$ in $\mathcal{P}_{\mathcal{N}}$. Such a function can be constructed as shown in [CDJS15]. We define $f_* : [n]^d \mapsto \mathbb{R}$, a restoration of f, such that $f_*(x) = f(x)$ for all $x \in \mathcal{N}$ and $f_*(x) = g(x)$ for all $x \notin \mathcal{N}$. Note that g is the function closest to f_* in $\mathcal{P}$.

Also, let $f^i_* : [n]^d \mapsto \mathbb{R}$ in $\mathcal{P}^i$ be such that $f^i_{*|\mathcal{N}}$ is the closest function to $f_{|\mathcal{N}}$ in $\mathcal{P}^i_{\mathcal{N}}$ for all $i \in [d]$. Therefore, we have,

$$\begin{aligned}
\frac{1}{4}\text{dist}(f_{|\mathcal{N}}, \mathcal{P}_{\mathcal{N}}) &= \frac{1}{4}\text{dist}(f_*, \mathcal{P}) \\
&\leq \sum_{i=1}^{d} \text{dist}(f_*, \mathcal{P}^i) && \text{by dimension reduction from [CDJS15]} \\
&\leq \sum_{i=1}^{d} \text{dist}(f_*, f^i_*) && \text{because } f^i_* \in \mathcal{P}^i \\
&\leq \sum_{i=1}^{d} \text{dist}(f_{|\mathcal{N}}, \mathcal{P}^i_{\mathcal{N}}) + d \cdot \alpha \cdot n^d.
\end{aligned}$$

The last inequality holds because, for all $i \in [d]$, $\text{dist}(f_*, f^i_*) \leq \text{dist}(f_{|\mathcal{N}}, \mathcal{P}^i_{\mathcal{N}}) + \alpha \cdot n^d$. ☐

We now use Claim 4.3.6 to prove the first part of our dimension reduction.

Lemma 4.3.7 (Dimension reduction: distance)**.** *Let ε_f be the relative Hamming distance of $f_{|\mathcal{N}}$ from $\mathcal{P}_{\mathcal{N}}$. Given $\ell \in \mathcal{L}$, let ε_ℓ denote the relative Hamming distance of f_ℓ from $\mathcal{P}_{\mathcal{N}_\ell}$. Then*

$$\mathbf{E}_{\ell \sim \mathcal{L}}[\varepsilon_\ell] \geq \frac{(1-\alpha) \cdot \varepsilon_f}{4d} - \alpha.$$

Proof. There are d axis-parallel directions and, therefore, dn^{d-1} axis-parallel lines in $[n]^d$. Thus, the probability of picking a specific axis parallel line ℓ uniformly at random is $1/dn^{d-1}$. Let $\mathcal{L}_i$ denote the set of axis parallel lines along dimension i.

$$\begin{aligned}
\mathbf{E}_{\ell \sim \mathcal{L}}[\varepsilon_\ell] &= \sum_{\ell \in \mathcal{L}} \varepsilon_\ell \cdot \Pr(\ell) \\
&= \sum_{i=1}^{d} \sum_{\ell \in \mathcal{L}_i} \varepsilon_\ell \cdot \Pr(\ell)
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{dn^{d-1}} \cdot \sum_{i=1}^{d} \sum_{\ell \in \mathcal{L}_i} \frac{\text{dist}(f_\ell, \mathcal{P}_{\mathcal{N}_\ell})}{|\mathcal{N}_\ell|} \\
&\geq \frac{1}{dn^{d}} \cdot \sum_{i=1}^{d} \sum_{\ell \in \mathcal{L}_i} \text{dist}(f_\ell, \mathcal{P}_{\mathcal{N}_\ell}) && \text{since } |\mathcal{N}_\ell| \leq n \\
&= \frac{1}{dn^{d}} \cdot \sum_{i=1}^{d} \text{dist}(f_{|\mathcal{N}}, \mathcal{P}^i_{\mathcal{N}}) \\
&\geq \frac{1}{dn^{d}} \cdot \left(\frac{\text{dist}(f_{|\mathcal{N}}, \mathcal{P}_{\mathcal{N}})}{4} - \alpha d \cdot n^d \right) && \text{by Claim 4.3.6} \\
&\geq \frac{1-\alpha}{4d} \cdot \varepsilon_f - \alpha.
\end{aligned}
$$

□

We conclude this section with the second part of our dimension reduction.

Lemma 4.3.8 (Dimension reduction: fraction of erasures)**.** *Consider an α-erased function $f : [n]^d \mapsto \mathbb{R} \cup \{\perp\}$. Given an axis-parallel line $\ell \in \mathcal{L}$, let α_ℓ denote the fraction of erased points in ℓ. Then, for every $\eta \in (0,1)$,*

$$\Pr_{\ell \sim \mathcal{L}}[\alpha_\ell > \alpha/\eta] \leq \eta.$$

Proof. Note that a uniformly randomly sampled point in $[n]^d$ is erased with probability α. We can sample a point uniformly at random by first sampling a line $\ell \in \mathcal{L}$ uniformly at random and then sampling a point uniformly randomly on ℓ, which is erased with probability α_ℓ. Therefore we have

$$\alpha = \sum_{\ell \in \mathcal{L}} \Pr[\ell] \cdot \alpha_\ell = \mathbf{E}_{\ell \sim \mathcal{L}}[\alpha_\ell].$$

The claim then follows from Markov inequality. □

4.3.3 Erasure-Resilient BDP Tester For the Hypergrid

In this section, we show that for every BDP $\mathcal{P}$, there exists an erasure-resilient tester for functions $f : [n]^d \mapsto \mathbb{R} \cup \{\perp\}$. We first present and analyze an erasure-resilient monotonicity tester for functions defined over $[n]^d$ and then present our erasure-resilient testers for all bounded derivative properties. Theorem 4.1.4 follows from the analysis of our monotonicity tester and Claim 4.3.3.

As before, we use $\mathcal{L}$ to denote the set of axis parallel lines in $[n]^d$. In each iteration, all our testers sample a line ℓ uniformly at random from $\mathcal{L}$ and does a random binary search for a uniformly sampled non-erased point in ℓ. It rejects if it detects a violation to the desired property in any of its iterations.

We present our erasure-resilient tester for monotonicity in Algorithm 2.

Algorithm 2 Erasure-Resilient Monotonicity Tester for $[n]^d$

```
Input: parameters $\varepsilon \in (0,1), \alpha \in (0, \varepsilon/81d]$; oracle access to $f : [n]^d \to \mathbb{R}$
1: Set $Q = \left\lceil H \cdot \left(1 + \frac{40d}{\varepsilon(1-\alpha)}\right) \cdot \left(1 + \ln\left(\frac{40d}{\varepsilon(1-\alpha)}\right)\right)\right\rceil$, where $H = 7\log\left(\max\left\{n, \sqrt{\frac{40d}{\varepsilon(1-\alpha)}}\right\}\right)$.
2: repeat $16d/\varepsilon(1-\alpha)$ times:
3:     Sample a line $\ell \in \mathcal{L}$ uniformly at random.
4:     Sample points uniformly at random from $\ell$ and query them until we get a non-erased point $s$.
5:     Perform a randomized binary search for $s$ on $\ell$ as in Algorithm 1.
6:     Reject if any violation to monotonicity is found.
7:     Go to Step 2 if the number of queries made in this iteration exceed $Q$.
8: Accept
```

We now analyze the above tester. The bound on the query complexity of Algorithm 2 is evident from the description of the tester. In the following lemma, we prove that the algorithm, with high probability, rejects an α-erased function that is ε-far from monotonicity.

Lemma 4.3.9. *If $f_{|\mathcal{N}}$ is ε-far from monotone, then the tester accepts with probability at most $1/3$.*

Proof. For $\ell \in \mathcal{L}$, let f_ℓ denote f restricted to the line ℓ. Let ε_ℓ denote the relative Hamming distance of f_ℓ from monotonicity and let α_ℓ denote the fraction of erasures on ℓ. As shown in the proof of Lemma 4.2.3, the probability that a random binary search on ℓ does not find a violation to monotonicity is at most $1 - \varepsilon_\ell$.

Consider a particular iteration of the tester. Let P_ℓ denote the random search path traversed by Algorithm 2 if it samples the line $\ell \in \mathcal{L}$ in that iteration and let q_ℓ denote the number of queries made. Let A be the event that the tester does not find a violation to monotonicity in that iteration. Then

$$\begin{aligned}
\Pr[A] &\le \sum_{\ell\in\mathcal{L}} \left(\Pr[A|q_\ell \le Q] + \Pr[q_\ell > Q]\right)\Pr[\ell] \\
&\le \sum_{\ell\in\mathcal{L}} (1-\varepsilon_\ell)\cdot\Pr[\ell] + \sum_{\ell\in\mathcal{L}} \Pr[q_\ell > Q]\cdot\Pr[\ell] \\
&\le 1 - \frac{\varepsilon(1-\alpha)}{4d} + \alpha + \sum_{\ell\in\mathcal{L}} \Pr[q_\ell > Q]\cdot\Pr[\ell]. && \text{by Lemma 4.3.7}
\end{aligned}$$

Next we bound $\Pr[q_\ell > Q]$ for $\ell \in \mathcal{L}$. Assume for now that $n \geq \sqrt{40d/(\varepsilon(1-\alpha))}$. Therefore $H = 7\log n$ in the description of the tester. Let $\eta = \frac{\varepsilon(1-\alpha)}{40d}$ and let $\delta_\ell = (1 + \eta\frac{(1-\alpha_\ell)}{\alpha_\ell})^{-1}$. Our choice of α implies that $\alpha_\ell \leq \alpha/\eta \leq 1/2$[4] and hence we have $\delta_\ell \leq (1+\eta)^{-1}$. It is not hard to see that the limit on the the number of queries in each iteration, Q, satisfies,

$$Q \geq \frac{7\log n(1+\ln(1/\eta))}{(1-(1+\eta)^{-1})} \geq \frac{7\log n(1+\ln(1/\eta))}{(1-\delta_\ell)}.$$

Let G_ℓ denote the (good) event that $\alpha_\ell \leq \alpha/\eta$, that the density of each interval in P_ℓ is at most δ_ℓ and that the length of P_ℓ is at most $7\log n$. Therefore,

$$\begin{aligned}\Pr[q_\ell > Q] &\leq \Pr[q_\ell > Q|G_\ell] + \Pr\left[\overline{G_\ell}\right]\\ &\leq \Pr\left[q_\ell > \frac{7\log n(1+\ln(1/\eta))}{(1-\delta_\ell)}|G_\ell\right] + \Pr\left[\overline{G_\ell}\right]\\ &\leq \eta+\eta+\eta+\frac{1}{n^2}.\end{aligned}$$

The last inequality above uses Corollary 4.2.8, Lemma 4.3.8, Claim 4.2.6 and Corollary 4.2.5 respectively. Therefore we have,

$$\begin{aligned}\Pr[q > Q] &= \sum_{\ell\in\mathcal{L}} \Pr[q_\ell > Q]\Pr[\ell]\\ &\leq \sum_{\ell\in\mathcal{L}}\left(3\eta + \frac{1}{n^2}\right)\Pr[\ell] = 3\eta + \frac{1}{n^2}.\end{aligned}$$

Note that $\frac{1}{n^2} \leq \eta$ for $n \geq \sqrt{40d/(\varepsilon(1-\alpha))}$. Therefore $\Pr[A] \leq 1-\frac{\varepsilon(1-\alpha)}{4d}+\alpha+4\eta \leq 1-\frac{\varepsilon(1-\alpha)}{8d}$. Hence, the probability that Algorithm 2 fails to detect a violation is at most $1/3$ since we have $\frac{16d}{\varepsilon(1-\alpha)}$ iterations. The analysis in the case when $n < \sqrt{40d/(\varepsilon(1-\alpha))}$ is similar to the above and results in the same bound on failure probability. □

We now present our (erasure-resilient) tester for a BDP $\mathcal{P}$ of functions $f : [n] \mapsto \mathbb{R}\cup\{\perp\}$ in Algorithm 3. In addition to α, ε and oracle access to f, it takes in the bounding function family $\mathbf{B}$ defining the BDP that we are testing for.

The analysis of this tester is along the lines of that for Algorithm 2. We give an outline of that here. Let $\mathcal{P}$ be the property that we test for. From Lemma 4.3.7 it follows that if $f_{|\mathcal{N}} : \mathcal{N} \mapsto \mathbb{R}$ is far from satisfying $\mathcal{P}$, then $f_\ell : \mathcal{N}_\ell \mapsto \mathcal{R}$ is also far from satisfying $\mathcal{P}$, where ℓ is a uniformly randomly

---

[4] It is not hard to see that $\alpha \leq \frac{\varepsilon}{81d} \leq \frac{\varepsilon}{80d+\varepsilon} \leq \frac{\varepsilon}{80d} = \frac{\eta}{2}$.

Algorithm 3 Erasure-Resilient BDP Tester for $[n]^d$

```
Input: parameters $\varepsilon \in (0,1), \alpha \in (0, \varepsilon/161d]$; oracle access to $f : [n]^d \to \mathbb{R}$; $\mathbf{B} = \{l_1, u_1, \ldots, l_d, u_d\}$.
1:  Set $Q = \left\lceil H \cdot \left(1 + \frac{80d}{\varepsilon(1-\alpha)}\right) \cdot \left(1 + \ln\left(\frac{80d}{\varepsilon(1-\alpha)}\right)\right)\right\rceil$, where $H = 7 \log\left(\max\left\{n, \sqrt{\frac{80d}{\varepsilon(1-\alpha)}}\right\}\right)$.
2:  repeat $200d/\varepsilon(1-\alpha)$ times:
3:      Sample a line $\ell \in \mathcal{L}$ uniformly at random.
4:      Let $i \in [d]$ be such that $\ell \in \mathcal{L}_i$.
5:      Let $g$ and $h$ be functions as defined in Claim 4.3.3 using $f_\ell, l_i$ and $u_i$.
6:      Sample points uniformly at random from $\ell$ and query them until we get a non-erased point $s$.
7:      Perform a randomized binary search for $s$ on $\ell$ as in Algorithm 1.
8:      Reject if any violation to monotonicity is found in either $g$ or $h$.
9:      Go to Step 2 if the number of queries made in this iteration exceed $Q$.
10: Accept.
```

chosen line in $\mathcal{L}$, f_ℓ is the function $f_{|\mathcal{N}}$ restricted to ℓ and $\mathcal{N}_\ell$ is the set of non-erased points on ℓ. From Claim 4.3.3, it is clear that testing for $\mathcal{P}$ of functions $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$ can be reduced to testing monotonicity. Using these facts and setting $\eta = \frac{\varepsilon(1-\alpha)}{80d}$, the statement of Theorem 4.1.4 follows.

4.4 Erasure-Resilient Convexity Tester For the Line

In this section, we prove Theorem 4.1.5. Given an α-erased function $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$, let ν_i denote the i-th non-erased domain point in $[n]$. The derivative of f at a point $\nu_i \in \mathcal{N}$, denoted by $\partial f(\nu_i)$, is $\frac{f(\nu_{i+1}) - f(\nu_i)}{\nu_{i+1} - \nu_i}$, whenever $\nu_{i+1} \leq n$. The function f is convex iff $\partial f(\nu_i) \leq \partial f(\nu_{i+1})$ for all $i \in [|\mathcal{N}| - 2]$. Our tester builds upon the ideas in the convexity tester from [PRR06a] and is conceptually simpler than their tester.

A high level idea of the tester is as follows. Our tester (Algorithm 4) has several iterations. In each iteration, it traverses a uniformly random *search* path in a binary search tree chosen uniformly at random. For each interval on the search path that it takes, it performs some checks. The algorithm accepts if none of these checks fail.

4.4.1 Technical Ingredients and Analysis Overview

We now give an overview of the analysis of the tester. We refer the reader to the full version for the complete analysis. Every iteration of the tester can be thought of as a traversal of a uniformly

Algorithm 4 Erasure-Resilient Convexity Tester

```
Input: parameters $\varepsilon, \alpha \in (0,1)$; oracle access to $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$.
Set $Q = c \log^2 n \frac{\alpha \log(8/\varepsilon)}{\varepsilon^3(1-\alpha)}$, where $c$ is a sufficiently large constant.
Accept if the number of queries exceeds $Q$.
repeat $2/\varepsilon$ times
    Sample points in $I[1,n]$ uniformly at random and query them until we get a point $\nu_{\mathbf{p}} \in \mathcal{N}$.
    TEST-INTERVAL($I[1,n], *, -\infty, +\infty, \nu_{\mathbf{p}}$) and reject if it rejects.    ▷ $*$ is a placeholder argument.
Accept.
```

Procedure 5 TEST-INTERVAL($I[i,j], \mathcal{A} = \langle a_1, a_2, \ldots, a_k \rangle, m_\ell, m_r, \nu_{\mathbf{p}}$)

```
Input: interval $I[i,j]$; a sorted list of non-erased points $\mathcal{A}$ or a placeholder $*$; left slope $m_\ell \in \mathbb{R}$;
    right slope $m_r \in \mathbb{R}$; target point $\nu_{\mathbf{p}} \in \mathcal{N}$.
Sample points uniformly at random from $I[i,j]$ and query them until we get a point $\nu_\lambda \in \mathcal{N}$.
Sequentially query points $\nu_\lambda + 1, \nu_\lambda + 2 \ldots$ until we get the next non-erased point $\nu_{\lambda+1}$.
Sequentially query points $\nu_\lambda - 1, \nu_\lambda - 2 \ldots$ until we get the next non-erased point $\nu_{\lambda-1}$.
Let $\mathcal{A}'$ denote the sorted list of points in the set $\{a_1, a_2, \ldots, a_k\} \cup \{\nu_\lambda, \nu_{\lambda-1}, \nu_{\lambda+1}\}$.
Let $m_i = \frac{f(b_{i+1}) - f(b_i)}{b_{i+1} - b_i}$
Reject if $m_\ell \le m_1 \le m_2 \le \cdots \le m_{k-1} \le m_r$ is not true.  ▷ Reject if $m_i$s are non-increasing.
if $\nu_{\mathbf{p}} < \nu_\lambda$ then
    Let $\mathcal{A}'_\ell$ denote the sorted list of points in $\mathcal{A}'$ that are smaller than $\nu_\lambda$.
    Reject if TEST-INTERVAL($I[i, \nu_\lambda - 1], \mathcal{A}'_\ell, m_\ell, \partial f(\nu_{\lambda-1}), \nu_{\mathbf{p}}$) rejects.
if $\nu_{\mathbf{p}} > \nu_\lambda$ then
    Let $\mathcal{A}'_r$ denote the sorted list of points in $\mathcal{A}'$ that are larger than $\nu_\lambda$.
    Reject if TEST-INTERVAL($I[\nu_\lambda + 1, j], \mathcal{A}'_r, \partial f(\nu_\lambda), m_r, \nu_{\mathbf{p}}$) rejects.
Accept.
```

random *search* path of a uniformly random binary search tree on $\mathcal{N}$, just as Algorithm 1. For each interval on such a path, we check a set of conditions computed based on the values at some non-erased points in the interval, called *anchor points*, and two real numbers, called the left and right slopes. First, we prove that, with high probability, the algorithm does not run out of its budget of queries Q. In the second part of the analysis we prove that, conditioned on the aforementioned event happening, in every iteration, with probability ε, the tester will detect a violation while testing for a function that is ε-far from being convex.

To analyze the query complexity, we classify the queries that the tester makes into two kinds and analyze them differently. The queries where the tester repeatedly samples and queries from an interval until it finds a non-erased domain point are called *sampling queries*. The queries where the tester keeps querying consecutive points, starting from a non-erased point, until it gets the next non-erased point are called *walking queries*. To deal with sampling queries, we use the combinatorial lemma (Lemma 4.2.2) and Claim 4.2.6. To analyze the walking queries, we prove a different combinatorial lemma (Lemma 4.4.1) that bounds the fraction of points with many erased points in their *vicinity*. Since all the walking queries start from points that we sample uniformly at random, with high probability, the number of walking queries made in any iteration is *small*. The remaining ideas in this part of the proof are similar to those in the analysis of Algorithm 1.

Lemma 4.4.1 (Combinatorial Lemma 2)**.** *Let I be a δ-sparse interval (see Definition 4.2.1). For every non-erased point $\nu_x \in I$, let $e(\nu_x)$ denote the number of erased points between ν_{x-1} and ν_{x+1}. Let β_ℓ denote the fraction of points $\nu_p \in \mathcal{N}$ such that $e(\nu_p) \geq \ell$. Then β_ℓ is at most $\frac{4\delta}{(1-\delta)\ell}$.*

Proof. We have $\sum_{\nu_x \in I \cap \mathcal{N}} e(\nu_x) \leq 2|I \setminus \mathcal{N}| < 2\delta|I|$. The first inequality follows from the observation that each erased point falls in the interval $I[\nu_{x-1}, \nu_{x+1}]$ for at most two non-erased points $\nu_x \in I$. The second inequality holds because I is δ-sparse. Then $\beta_\ell \cdot |I \cap \mathcal{N}| \cdot \ell \leq 2 \cdot \sum_{\nu_x \in I \cap \mathcal{N}} e(\nu_x) < 4\delta \cdot |I|$. Since $|I \cap \mathcal{N}| > (1-\delta)|I|$, the lemma follows. □

We now define notation for the latter part of the proof. Consider a search path traversed by the algorithm. Let $I[i,j]$ be an interval on the path. Consider the execution of TEST-INTERVAL (Procedure 5) called with $I[i,j]$ as the first argument. We call the non-erased point ν_λ sampled in Step 1 its *pivot*, the sorted list of non-erased points $\mathcal{A}'$ in Step 4 its *anchor sequence* and the values m_ℓ and m_r as its *left* and *right slopes*, respectively. That is, given a binary search tree $\mathcal{T}$, we associate each interval appearing in the tree with a pivot, an anchor sequence and two slopes.

Consider a binary search tree $\mathcal{T}$ and a function $f : [n] \mapsto \mathbb{R} \cup \{\perp\}$. Let $I[i,j]$ be an interval appearing in $\mathcal{T}$ with an anchor sequence $\mathcal{A} = \langle b_1, b_2, \ldots, b_k \rangle$ and slopes m_ℓ and m_r. Let $m_i = \frac{f(b_{i+1}) - f(b_i)}{b_{i+1} - b_i}$ $\forall i \in [k-1]$.

Definition 4.4.2 (Good Interval, Bad Interval)**.** *An interval $I[i,j]$ is* good *if $m_\ell \leq m_1 \leq m_2 \leq \cdots \leq m_{k-1} \leq m_r$. Otherwise, it is* bad.

Definition 4.4.3 (Violator Interval)**.** *An interval $I[i,j]$ is a* violator *if it is* bad *and all its ancestor intervals in $\mathcal{T}$ are* good.

Definition 4.4.4 (Witness)**.** *A non-erased domain point is a* witness *with respect to $\mathcal{T}$ if it belongs to a violator interval in $\mathcal{T}$.*

In Claim 4.4.7, we prove that if $f_{|\mathcal{N}}$ is ε-far from being convex, then, for every binary search tree $\mathcal{T}$, the fraction of non-erased domain points that are witnesses is at least ε. To prove this, we start by assuming that there is a tree in which the fraction of witnesses is less than ε. We show that we can correct the function values on the witnesses and get a convex function, which gives a contradiction.

4.4.2 Analysis of the Tester

In this section, we analyze Algorithm 4. The bound on the query complexity of the algorithm is evident from its description. It remains to prove its correctness.

Lemma 4.4.5. *Algorithm 4 accepts if $f_{|\mathcal{N}}$ is convex, and rejects with probability at least $2/3$ if $f_{|\mathcal{N}}$ is ε-far from convex.*

The tester accepts whenever $f_{|\mathcal{N}}$ is convex. To prove the other part of the lemma, assume that $f_{|\mathcal{N}}$ is ε-far from being convex. Let A be the event that the tester accepts f. Let q denote the total number of queries made. We prove that $\Pr[A] \leq 1/3$. The event A occurs if either $q > Q$ or the tester does not find a violation in any of the $4/\varepsilon$ iterations of Step 3. Thus,

$$\Pr\left[A\right] \leq \Pr\left[A|q \leq Q\right] + \Pr\left[q > Q\right].$$

First, we bound $\Pr[q > Q]$. It is enough to prove that, with high probability, in each iteration the number of queries made by the tester is at most $Q/(\frac{2}{\varepsilon})$, since, by a union bound,

$$\Pr\left[q > Q\right] \leq \sum_{i=1}^{\lceil \frac{2}{\varepsilon} \rceil} \Pr\left[q_i > Q/\left(\frac{2}{\varepsilon}\right)\right],$$

where q_i is the number of queries made in the i-th iteration.

Let $\delta = \left(1 + \eta\frac{(1-\alpha)}{\alpha}\right)^{-1}$, $H = 7\log n$ and $\eta = \varepsilon/38$. Consider the i^{th} iteration of the tester. Let P_i denote the search path taken by the algorithm, and let $|P_i|$ denote the number of intervals on P_i. Let q_i^w and q_i^s denote the number of walking queries and sampling queries in iteration i.

Let G_i denote the (good) event that the length of P_i is at most $7\log n$, the density of each interval in P_i is at most δ. By Corollary 4.2.8,

$$\Pr\left[q_i^s > c_1\left(1 + \log\frac{1}{\varepsilon}\right) \cdot \left(1 + \frac{\alpha}{\varepsilon(1-\alpha)}\right) \cdot \log n | G_i\right] \leq \eta, \tag{4.1}$$

where c_1 is a large enough constant. To analyze the walking queries, we use Corollary 4.4.6 of Lemma 4.4.1.

Corollary 4.4.6. *Let I be a δ-sparse interval. Then the number of walking queries made in Step 2 and Step 3 of Procedure 5 is more than $\frac{14\alpha \log n}{\eta^2(1-\alpha)}$ with probability at most $\frac{\eta}{7 \log n}$.*

Proof. By Lemma 4.4.1, the fraction of non-erased domain points ν_λ in I with $e(\nu_\lambda) > \frac{14\alpha \log n}{\eta^2(1-\alpha)}$ is at most $\frac{\eta}{7 \log n}$. Since the pivot in I is chosen uniformly at random from $I \cap \mathcal{N}$, the claim about probability follows. ☐

For a large enough constant c_1, by Corollary 4.4.6 and a union bound over all the intervals in P_i,

$$\Pr\left[q_w^i > c_1 \cdot \frac{\alpha \log^2 n}{\varepsilon^2(1-\alpha)}|G_i\right] \leq \eta. \tag{4.2}$$

Therefore,

$$\begin{aligned}
\Pr\left[q_i > \frac{Q\varepsilon}{2}\right] \leq \Pr&\left[q_w^i > c_1 \cdot \frac{\alpha \log^2 n}{\varepsilon^2(1-\alpha)}|G_i\right] + \\
&\Pr\left[q_i^s > c_1\left(1 + \log\frac{1}{\varepsilon}\right) \cdot \left(1 + \frac{\alpha}{\varepsilon(1-\alpha)}\right) \cdot \log n|G_i\right] + \Pr[\overline{G_i}] \\
&\leq \eta + \eta + \left(\eta + \frac{1}{n^2}\right),
\end{aligned}$$

where the second inequality uses Eq. (4.1), Eq. (4.2), Claim 4.2.6 and Corollary 4.2.5 with a union bound, . Since $\eta = \frac{\varepsilon}{38}$, the error probability in this case becomes $3\varepsilon/38 + 1/n^2 \leq \varepsilon/12$. Hence, $\Pr[q > Q] \leq \frac{4}{\varepsilon} \cdot \frac{\varepsilon}{12} = 1/6$.

Next, we bound $\Pr[A|q \leq Q]$. To do this, we first prove that the fraction of witnesses (see Definition 4.4.4) in any binary search tree $\mathcal{T}$ is at least ε. We do it in the following claim.

Claim 4.4.7. *If $f_{|\mathcal{N}}$ is ε-far from convex, then the fraction of witnesses in every binary search tree $\mathcal{T}$ is more than ε.*

Proof. Assume for the sake of contradiction that there is a binary search tree $\mathcal{T}$ such that the fraction of witnesses with respect to $\mathcal{T}$ is at most ε. In the following, we will construct a convex function $g : [n] \mapsto \mathbb{R} \cup \{\perp\}$ by changing the values of f only on witnesses with respect to $\mathcal{T}$. Since the fraction of witnesses is at most ε, functions f and g will differ on at most an ε fraction of non-erased domain points, which results in a contradiction. Since, by our assumption, the fraction of witnesses is at most ε, the interval $I[1, n]$ cannot be a violator.

Consider a violator interval $I[i, j]$ in $\mathcal{T}$. Let the anchor sequence and slopes associated with the *parent interval* of $I[i, j]$ be $\mathcal{A} = \langle b_1, b_2, \dots, b_k\rangle$ and m_ℓ and m_r, respectively. Assume that $I[i, j]$ is the right child of its parent. The case when $I[i, j]$ is the left child of its parent is similar.

Let $b_x, b_{x+1}, \ldots, b_k$ be the set of points common to $I[i,j]$ and $\mathcal{A}$. By definition, b_x is the smallest non-erased domain point in $I[i,j]$. Also, the left slope of $I[i,j]$ is $\frac{f(b_x)-f(b_{x-1})}{b_x-b_{x-1}}$ and its right slope is equal to m_r.

Let $m_y = \frac{f(b_{y+1})-f(b_y)}{b_{y+1}-b_y}$ for all integers y such that $y \in [x-1, k)$. We define g as follows.

- For each $t \in \{b_x, b_{x+1}, \ldots, b_k\}$, set $g(t) = f(t)$.
- For each integer $y \in [x,k)$ and $t \in \mathcal{N} \cap (b_y, b_{y+1})$, set

$$g(t) = f(b_y) + m_y \cdot (t - b_y)$$

- For each $t \in \mathcal{N}$ such that $t > b_k$, set

$$g(t) = f(b_k) + m_{k-1} \cdot (t - b_k).$$

Since $I[i,j]$ is a violator, the parent interval of $I[i,j]$ is good, by definition. This implies that $m_{x-1} \le m_x \le \ldots \le m_k \le m_r$. Therefore, the derivatives of non-erased points in $I[i,j]$ are non-decreasing with respect to g, by virtue of our assignment.

To prove that $g_{|\mathcal{N}}$ is convex, we first show that every interval in $\mathcal{T}$ is good with respect to g.

1. Consider an interval I in $\mathcal{T}$ that is good with respect to f. If I has no ancestors or descendants that are violators, it remains good with respect to g as well, since $g(t) = f(t)$ for all $t \in I[i,j]$.
2. Consider an interval I that has a descendant I' that is a violator. The definition of g on points in I' ensures that $g(t) = f(t)$ for every point t common to the anchor sequence of I and the interval I'. Thus, I remains good with respect to g.
3. Consider a node I that is either a violator or has a violator ancestor I'. By definition, the parent of I' is good with respect to f. Therefore, by the definition of g on I', we have $\partial g(t-1) \le \partial g(t)$ for all $t \in \mathcal{N}$ such that $t \in I'$. Therefore, I' is good with respect to g, and hence I is also good with respect to g.

We proved that every interval in the tree $\mathcal{T}$ is good with respect to g. We now prove that $g_{|\mathcal{N}}$ is convex. Consider a point $\nu_t \in \mathcal{N}$ such that $2 \le t \le |\mathcal{N}| - 1$. This point occurs in $\mathcal{T}$ either as a pivot in a non-leaf interval or as the sole non-erased domain point in a leaf interval. In the former case, the condition $\partial f(\nu_{t-1}) \le \partial f(\nu_t)$ is part of the goodness condition of the corresponding interval and is satisfied. In the latter case, $\partial f(\nu_{t-1})$ and $\partial f(\nu_t)$ are the left and right slopes of the leaf and are compared as part of the goodness condition of the leaf. Thus, $\partial f(\nu_{t-1}) \leq \partial f(\nu_t)$ for all $\nu_t \in \mathcal{N}$ such that $2 \leq t \leq |\mathcal{N}| - 1$. Thus, $g_{|\mathcal{N}}$ is convex. $\square$

We conclude our analysis by bounding the probability that the tester does not find a violation. Since the target point ν_p is chosen uniformly at random from the set of non-erased domain points, the probability that it is a witness is at least ε and thus, the tester detects a violation to convexity with probability at least ε in every iteration. Therefore,

$$\Pr[A] \leq \Pr[A|q \leq Q] + \Pr[q > Q] \leq (1-\varepsilon)^{\frac{2}{\varepsilon}} + 1/6 \leq 1/3.$$

$\square$

Bibliography

[AC06] N. Ailon and B. Chazelle. Information theory in property testing and monotonicity testing in higher dimension. *Inform. and Comput.*, 204(11):1704–1717, 2006. 2, 3, 8, 12, 13, 15

[ACCL07] N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Estimating the distance to a monotone function. *Random Structures Algorithms*, 31(3):371–383, 2007. 8, 15, 18

[AJMR12] P. Awasthi, M. Jha, M. Molinaro, and S. Raskhodnikova. Testing Lipschitz functions on hypergrid domains. In *Proceedings, International Workshop on Randomization and Computation (RANDOM)*, 2012. 9, 14

[ALM$^+$98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998. 2

[BBM12] E. Blais, J. Brody, and K. Matulef. Property testing lower bounds via communication complexity. *Comp. Complexity*, 21(2):311–358, 2012. 8, 12

[BCGSM12] J. Briët, S. Chakraborty, D. García-Soriano, and A. Matsliah. Monotonicity testing and shortest-path routing on the cube. *Combinatorica*, 32(1):35–53, 2012. 8

[BFR$^+$13] Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing closeness of discrete distributions. *J. ACM*, 60(1):4, 2013. 44

[BGJ$^+$12] Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM J. Comput.*, 41(6):1380–1425, 2012. 8, 15, 18, 45

[BKF$^+$11] Sagi Ben-Moshe, Yaron Kanza, Eldar Fischer, Arie Matsliah, Mani Fischer, and Carl Staelin. Detecting and exploiting near-sortedness for efficient relational query evaluation. In Tova Milo, editor, *Database Theory - ICDT 2011, 14th International Conference, Uppsala, Sweden, March 21-24, 2011, Proceedings*, pages 256–267. ACM, 2011. 2

[BRW05] T. Batu, R. Rubinfeld, and P. White. Fast approximate PCPs for multidimensional bin-packing problems. *Inform. and Comput.*, 196(1):42–56, 2005. 8, 15, 18

[BRY14a] P. Berman, S. Raskhodnikova, and G. Yaroslavtsev. Testing with respect to l_p distances. In *Proceedings, ACM Symp. on Theory of Computing (STOC)*, 2014. 2

[BRY14b] E. Blais, S. Raskhodnikova, and G. Yaroslavtsev. Lower bounds for testing properties of functions on hypergrid domains. In *Proceedings, IEEE Conference on Computational Complexity (CCC)*, March 2014. 8, 9, 10, 13, 15

[CDJS15] Deeparnab Chakrabarty, Kashyap Dixit, Madhav Jha, and C. Seshadhri. Property testing on product distributions: Optimal testers for bounded derivative properties. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1809–1828. SIAM, 2015. 2, 8, 9, 11, 47, 55, 57

[CS13a] Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *STOC*, pages 419–428, 2013. 8, 9, 12, 14, 16, 22, 45

[CS13b] Deeparnab Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. In *APPROX-RANDOM*, pages 425–435, 2013. 4, 8, 12, 15, 32, 36

[Dev86] Luc Devroye. A note on the height of binary search trees. *J. ACM*, 33(3):489–498, 1986. 5, 47, 49, 52

[DGL$^+$99] Y. Dodis, O. Goldreich, E. Lehman, S. Raskhodnikova, D. Ron, and A. Samorodnitsky. Improved testing algorithms for monotonicity. In *Proceedings, International Workshop on Randomization and Computation (RANDOM)*, 1999. 3, 8, 12, 13

[DJRT13] Kashyap Dixit, Madhav Jha, Sofya Raskhodnikova, and Abhradeep Thakurta. Testing the lipschitz property over product distributions with applications to data privacy. In *TCC*, pages 418–436, 2013. 2, 3, 9, 11

[DMNS06] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings, Theory of Cryptography Conference (TCC)*, 2006. 11

[EKK$^+$00] F. Ergun, S. Kannan, R. Kumar, R. Rubinfeld, and M. Viswanathan. Spot-checkers. *J. Comput. System Sci.*, 60(3):717–751, 2000. 8, 15, 18, 45, 47, 49

[Fis04] E. Fischer. On the strength of comparisons in property testing. *Inform. and Comput.*, 189(1):107–116, 2004. 4, 8, 15, 32, 45, 47

[FLN$^+$02] Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 474–483, New York, NY, USA, 2002. ACM. 8, 12, 18

[GGL$^+$00] O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samorodnitsky. Testing monotonicity. *Combinatorica*, 20:301–337, 2000. 3, 8, 12, 13

[GGR98] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. 1, 2, 4, 7, 45

[GR15] Oded Goldreich and Dana Ron. On sample-based testers. In Tim Roughgarden, editor, *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*, pages 337–345. ACM, 2015. 44

[HK05] S. Halevy and E. Kushilevitz. A lower bound for distribution-free monotonicity testing. In *Proceedings, International Workshop on Randomization and Computation (RANDOM)*, pages 330–341, 2005. 2

[HK08a] S. Halevy and E. Kushilevitz. Testing monotonicity over graph products. *Random Structures Algorithms*, 33(1):44–67, 2008. 2, 3, 8, 13, 15

[HK08b] Shirley Halevy and Eyal Kushilevitz. Distribution-free connectivity testing for sparse graphs. *Algorithmica*, 51(1):24–48, 2008. 2

[Jan14] Svante Janson. Large deviations for sums of geometric and exponential variables. 2014. `http://www2.math.uu.se/~svante/papers/sjN14.pdf`. 52

[JR13] Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of Lipschitz functions with applications to data privacy. *SIAM J. Comput.*, 42(2):700–731, 2013. 2, 3, 9, 14, 15, 18

[Knu73] D. E. Knuth. *The Art of Computer Programming Vol III: Sorting and Searching*, volume 3. Addison-Wesley, 1973. 4, 8, 12

[KRTA13] Simon Korman, Daniel Reichman, Gilad Tsur, and Shai Avidan. Fast-match: Fast affine template matching. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 2331–2338. IEEE, 2013. 2

[LR01] E. Lehman and D. Ron. On disjoint chains of subsets. *J. Combin. Theory Ser. A*, 94(2):399–404, 2001. 8, 12

[Meh75] K. Mehlhorn. Nearly optimal binary search trees. *Acta Informatica*, 5:287–295, 1975. 12, 32, 38

[PRR03] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. On testing convexity and submodularity. *SIAM J. Comput.*, 32(5):1158–1184, 2003. 5

[PRR06a] M. Parnas, D. Ron, and R. Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. System Sci.*, 6(72):1012–1042, 2006. 8, 10, 45, 48, 61

[PRR06b] M. Parnas, D. Ron, and R. Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. System Sci.*, 6(72):1012–1042, 2006. 15

[RS96] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM J. Comput.*, 25:647–668, 1996. 1, 2, 4, 7, 45

[RT14] Dana Ron and Gilad Tsur. Testing properties of sparse images. *ACM Transactions on Algorithms*, 10(4):17:1–17:52, 2014. 2

[Val84] Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. 2

[Yao82] F. F. Yao. Speed-up in dynamic programming. *J. Alg. Discrete Math.*, 3:532–540, 1982. 12

Vita

Kashyap Dixit

Kashyap Dixit is a Ph.D candidate in the Department of Computer Science and Engineering at the Pennsylvania State University. Kashyap completed his Bachelor and Master of Technology degrees from Indian Institute of Technology in 2008. Thereafter, he worked as a software engineer in IBM Research, India until July 2010. Kashyap started his PhD program in August 2010. His research interests are sublinear algorithms, property testing, random sampling and counting algorithms.