Testing Lipschitz Functions on Hypergrid Domains*

Pranjal Awasthi[1], Madhav Jha[2], Marco Molinaro[1], and Sofya Raskhodnikova[2]**

[1] Carnegie Mellon University, USA,
{pawasthi,molinaro}@cmu.edu.
[2] Pennsylvania State University, USA,
{mxj201,sofya}@cse.psu.edu.

Abstract. A function $f(x_1, ..., x_d)$, where each input is an integer from 1 to n and output is a real number, is Lipschitz if changing one of the inputs by 1 changes the output by at most 1. In other words, Lipschitz functions are not very sensitive to small changes in the input. Our main result is an efficient tester for the Lipschitz property of functions $f : [n]^d \to \delta\mathbb{Z}$, where $\delta \in (0, 1]$ and $\delta\mathbb{Z}$ is the set of integer multiples of δ.
The main tool in the analysis of our tester is a smoothing procedure that makes a function Lipschitz by modifying it at a few points. Its analysis is already nontrivial for the 1-dimensional version, which we call Bubble Smooth, in analogy to Bubble Sort. In one step, Bubble Smooth modifies two values that violate the Lipschitz property, i.e., differ by more than 1, by transferring δ units from the larger to the smaller. We define a *transfer graph* to keep track of the transfers, and use it to show that the ℓ_1 distance between f and BubbleSmooth(f) is at most twice the ℓ_1 distance from f to the nearest Lipschitz function. Bubble Smooth has other important properties, which allow us to obtain a *dimension reduction*, i.e., a reduction from testing functions on multidimensional domains to testing functions on the 1-dimensional domain, that incurs only a small multiplicative overhead in the running time and thus avoids the exponential dependence on the dimension.

1 Introduction

Property testing aims to understand how much information is needed to decide (approximately) whether an object has a property. A *property tester* [8, 5] is given oracle access to an object and a proximity parameter $\epsilon \in (0, 1)$. If an object has the desired property, the tester *accepts* it with probability at least $2/3$; if the object is ϵ-far from having the desired property then the tester *rejects* it with probability at least $2/3$. Specifically, for properties of functions, ϵ-far means that a given function differs on at least an ϵ fraction of the domain points from any function with the property. Properties of different types of objects have been studied, including graphs, metrics spaces, images and functions.

We present efficient testers for the Lipschitz property of functions[3] $f : [n]^d \to \delta\mathbb{Z}$, where $\delta \in (0, 1]$ and $\delta\mathbb{Z}$ is the set of integer multiples of δ. A function f is *c-Lipschitz*

* All omitted proofs appear in the full version [1].

** P.A. is supported by NSF grant CCF-1116892. M.J. and S.R. are supported by NSF CAREER grant CCF-0845701 and NSF grant CCF-0729171. M.M. is supported by NSF grant CMMI-1024554.

[3] The set $\{1, \ldots, n\}$ is denoted by $[n]$.

(with respect to the ℓ_1 metric on the domain) if $|f(x) - f(y)| \leq c \cdot |x - y|_1$. Points in the domain $[n]^d$ can be thought of as vertices of a d-dimensional *hypergrid*, where every pair of points at ℓ_1 distance 1 is connected by an edge. Each edge (x, y) imposes a constraint $|f(x) - f(y)| \leq c$ and a function f is c-Lipschitz iff every edge constraint is satisfied. We say a function is *Lipschitz* if it is 1-Lipschitz. (Note that rescaling by a factor of $1/c$ converts a c-Lipschitz function into a Lipschitz function.)

Testing of the Lipschitz property was first studied by Jha and Raskhodnikova [7] who motivated it by applications to data privacy and program verification. They presented testers for the Lipschitz property of functions on the domains $\{0,1\}^d$ (the *hypercube*) and $[n]$ (the *line*) that run in time $O(d^2/(\delta\epsilon))$ and $O(\log n/\epsilon)$, respectively. Even though the applications in [7] are most convincing for functions on general hypergrid domains (in one of their applications, for instance, a point in $[n]^d$ represents a histogram of a private database), no nontrivial tester for functions on such general domains was known prior to this work.

1.1 Our Results

We present two efficient testers of the Lipschitz property of functions of the form $f : [n]^d \to \delta\mathbb{Z}$ with running time polynomial in d, n and $(\delta\epsilon)^{-1}$. Our testers are faster for functions whose image has small diameter.

Definition 1.1 (Image diameter). *Given a function $f : [n]^d \to \mathbb{R}$, its* image diameter *is* $\mathrm{ImgD}(f) = \max_{x\in[n]^d} f(x) - \min_{y\in[n]^d} f(y)$.

Observe that a Lipschitz function on $[n]^d$ must have image diameter at most nd. However, image diameter can be arbitrarily large for a non-Lipschitz function.

Our testers are *nonadaptive*, that is, their queries do not depend on answers to previous queries. The first tester has *1-sided error*, that is, it always accepts Lipschitz functions. The second tester is faster (when $\sqrt{d} \gg \log(1/\epsilon)$ and $\mathrm{ImgD}(f)$ is large), but has 2-sided error, i.e., it can err on both positive and negative instances.

Theorem 1.1 (Lipschitz testers). *For*[4] *all $\delta, \epsilon \in (0, 1]$, the Lipschitz property of functions $f : [n]^d \to \delta\mathbb{Z}$ can be tested nonadaptively with the following time complexity:*

(1) in $O\left(\frac{d}{\delta\epsilon} \cdot \min\{\mathrm{ImgD}(f), nd\} \cdot \log\min\{\mathrm{ImgD}(f), n\}\right)$ time with 1-sided error.

(2) in $O\left(\frac{d}{\delta\epsilon} \cdot \min\left\{\mathrm{ImgD}(f), n\sqrt{d\log(1/\epsilon)}\right\} \cdot \log\min\{\mathrm{ImgD}(f), n\}\right)$ time with 2-sided error.

If $\mathrm{ImgD}(f)$, δ and ϵ are constant, then both testers run in $O(d)$ time. This is tight already for the range $\{0, 1, 2\}$, even for the special case of the hypercube domain [7].

1.2 Our Techniques

For clarity of presentation, we state and prove all our theorems for $\delta = 1$, i.e., for integer-valued functions. In the full version, by discretizing (as was done in [7]), we extend our results to the range $\delta\mathbb{Z}$.

[4] If $\delta > 1$ then f is Lipschitz iff it is 0-Lipschitz (that is, constant). Testing if a function is constant takes $O(1/\epsilon)$ time.

The main challenge in designing a tester for functions on the hypergrid domains is avoiding an exponential dependence on the dimension d. We achieve this via a *dimension reduction*, i.e., a reduction from testing functions on the hypergrid $[n]^d$ to testing functions on the line $[n]$, that incurs only an $O(d \cdot \min\{\mathrm{ImgD}, nd\})$ multiplicative overhead in the running time. In order to do this, we relate the distance to the Lipschitz property of a function f on the hypergrid to the average distance to the Lipschitz property of restrictions of f to 1-dimensional (axis-parallel) lines. For $i \in [d]$, let $e^i \in [n]^d$ be 1 on the ith coordinate and 0 on the remaining coordinates. Then for every dimension $i \in [d]$ and $\alpha \in [n]^d$ with $\alpha_i = 0$, the *line g of f along dimension i with position* α is the restriction of f defined by $g(x_i) = f(\alpha + x_i \cdot e^i)$, where x_i ranges over $[n]$. We denote the set of lines of f along dimension i by L_f^i and the set of all lines, i.e., $\bigcup_{i\in[d]} L_f^i$, by L_f. We denote the relative distance of a function h to the Lipschitz property, i.e., the fraction of input points where the function needs to be changed in order to become Lipschitz, by $\epsilon^{Lip}(h)$. The technical core of our dimension reduction is the following theorem that demonstrates that if a function on the hypergrid is far from the Lipschitz property then a random line from L_f is, in expectation, also far from it.

Theorem 1.2 (Dimension reduction). *For all functions $f : [n]^d \to \mathbb{Z}$, the following holds:* $\mathbb{E}_{g\leftarrow L_f}\left[\epsilon^{Lip}(g)\right] \geq \frac{\epsilon^{Lip}(f)}{2\cdot d\cdot \mathrm{ImgD}(f)}$.

To obtain this result, we introduce a smoothing procedure that "repairs" a function (i.e., makes it Lipschitz) one dimension at a time, while modifying it at a few points. Such procedures have been previously designed for restoring monotonicity of Boolean functions [4, 3] and for restoring the Lipschitz property of functions on the hypercube domain [7]. The key challenge is to find a smoothing procedure that satisfies the following three requirements: *(1) It makes all lines along dimension i (i.e., in L_f^i) Lipschitz. (2) It changes only a small number of function values. (3) It does not make lines in other dimensions less Lipschitz, according to some measure.*

Smoothing Procedure for 1-dimensional Functions. Our first technical contribution is a local smoothing procedure for functions $f : [n] \to \mathbb{Z}$, which we call **BubbleSmooth**, in analogy to Bubble Sort. In one *basic step*, **BubbleSmooth** modifies two consecutive values (i.e., $f(i)$ and $f(i+1)$ for some $i \in [n-1]$) that violate the Lipschitz property, namely, differ by more than 1. It decreases the larger and increases the smaller by 1, i.e., it transfers a unit from the larger to the smaller. See Algorithm 1 for the description of the order in which basic steps are applied. **BubbleSmooth** is a natural generalization of the *averaging operator* in [7], used to repair an edge of the hypercube, that can also be viewed as several applications of the basic step to the edge.

One challenge in analyzing **BubbleSmooth** is that when it is applied to all lines in one dimension, it may increase the average distance to the Lipschitz property for the lines in the remaining dimensions. Our second key technical insight is to use the ℓ_1 distance to the Lipschitz property to measure the performance of our procedure on the line and its effect on other dimensions. The ℓ_1 distance between functions f and f' on the same domain, denoted by $|f - f'|_1$, is the sum of $|f(x) - f'(x)|$ over all values x in the domain. The ℓ_1 distance of a function f to the nearest Lipschitz function over the same domain is denoted by $\ell_1^{Lip}(f)$. Observe that the Hamming distance and the ℓ_1 distance from a function to a property can differ by at most $\mathrm{ImgD}(f)$. Later, we leverage the fact that Lipschitz functions have a relatively small image diameter to relate the ℓ_1 distance to the Hamming distance.

We prove that **BubbleSmooth** returns a Lipschitz function and that it makes at most twice as many changes in terms of ℓ_1 distance as necessary to make a function Lipschitz.

Theorem 1.3. *Consider a function $f : [n] \to \mathbb{Z}$ and let f' be the function returned by **BubbleSmooth**(f). Then (1) function f' is Lipschitz and (2) $|f - f'|_1 \leq 2 \cdot \ell_1^{Lip}(f)$.*

The proof of the second part of this theorem requires several technical insights. One of the challenges is that **BubbleSmooth** changes many function values, but then undoes most changes during subsequent steps. We define a transfer graph to keep track of the transfers that move a unit of function value during each basic step. Its vertex set is $[n]$ and an edge (x, y) represents that a unit was transferred from $f(x)$ to $f(y)$. Since two transfers (x, y) and (y, z) are equivalent to a transfer (x, z), we can merge the corresponding edges in the transfer graph, proceeding with such merges until no vertex in it has both incoming and outgoing edges. As a result, we get a transfer graph, where the number of edges, $|E|$, is twice the ℓ_1 distance from the original to the final function.

To prove that $|E| \leq \ell_1^{Lip}(f)$, we show that the transfer graph has a matching with the violation score at least $|E|$. The *violation score* of an edge (or a pair) (x, y) is the quantity by which $|f(x) - f(y)|$ exceeds the distance between x and y. (Recall that $|f(x) - f(y)| \leq |x - y|$ for all Lipschitz functions f on domain $[n]$.) The violation score of a matching is the sum of the violation scores over all edges in the matching. We observe (in Lemma 2.3) that $\ell_1^{Lip}(f)$ is at least a violation score of any matching. The crucial step in obtaining a matching with a large violation score is pinpointing a provable, but strong enough property of the transfer graph that guarantees such a matching. Specifically, we show that the violation score of each edge in the graph is at least the number of edges adjacent to its endpoints at its (suitably defined) *moment of creation* (Lemma 2.1). E.g., this statement is not true for adjacent edges in the final transfer graph. The construction of a matching with a large violation score in the transfer graph is one of the key technical contributions of this paper. It is the focus of Section 2.

Dimension Reduction with respect to $\boldsymbol{\ell_1}$. Our smoothing procedure for functions on the hypergrids applies **BubbleSmooth** to repair all lines in dimensions $1, 2, ..., d$, one dimension at a time. We show that for all $i, j \in [d]$ applying **BubbleSmooth** in dimension i does not increase the expected $\ell_1^{Lip}(f)$ for a random line g in dimension j. The key feature of our smoothing procedure that makes the analysis tractable is that it can be broken down into steps, each consisting of one application of the basic step of **BubbleSmooth** to the same positions $(k, k+1)$ on all lines in a specific dimension. This allows us to show that one such step does not make other dimensions worse in terms of the ℓ_1 distance to the Lipschitz property. The cleanest statement of the resulting dimension reduction is with respect to the ℓ_1 distance.

Theorem 1.4. *For all functions $f : [n]^d \to \mathbb{Z}$, we have: $\sum_{g \in L_f} \ell_1^{Lip}(g) \geq \frac{\ell_1^{Lip}(f)}{2}$.*

Our Testers and Effective Image Diameter. The main component of our tester repeats the following procedure: *Pick a line uniformly at random and run one step of the line tester.* (We use the line tester from [7].) Our dimension reduction (Theorem 1.2) is crucial in analyzing this component. However, the bound in Theorem 1.2 depends on the image diameter of the function f. In the case of a non-Lipschitz function, it can be arbitrarily large, but for a Lipschitz function on $[n]^d$ it is at most the diameter of the space, namely nd (notice this factor in part (1) of Theorem 1.1). In fact, for our application we can also use the *observable diameter* of the space [6]: since the hypergrid exhibits Gaussian-type concentration of measure, a Lipschitz function maps the vast majority of points to an interval of size $O(n\sqrt{d})$ (notice this factor in part (2) of Theorem 1.1). Our testers use a preliminary step to rule out functions with large image diameter (resulting in 1-sided error) or with large observable diameter (resulting in 2-sided error).

1.3 Comparison to Previous and Concurrent Work

Jha and Raskhodnikova [7] gave a 1-sided error nonadaptive testers for the Lipschitz property of functions of the form $f:\{0,1\}^d \to \delta\mathbb{Z}$ and $f:[n]\to\mathbb{R}$ that run in time $O\left(\frac{d}{\delta\epsilon}\cdot\min\{\mathrm{ImgD}(f),d\}\right)$ and $O\left(\frac{\log n}{\epsilon}\right)$, respectively. They also showed that $\Omega(d)$ queries are necessary for testing the Lipschitz property on the domain $\{0,1\}^d$, even when the range is $\{0,1,2\}$. No nontrivial tester of the Lipschitz property of functions on the domain $[n]^d$ was known prior to this work.

Our first tester from Theorem 1.1 naturally generalizes the testers of [7] to functions on the domain $[n]^d$. As in [7], our tester has at most quadratic dependence on the dimension d. Our second tester from Theorem 1.1 gives an improvement in the running time over the hypercube tester in [7] at the expense of allowing 2-sided error. In this specific case, Theorem 1.1 gives a tester with running time $\tilde{O}(d^{1.5}/(\delta\epsilon))$.

Concurrently with our work, Chakrabarty and Seshadhri [2] gave an ingenious analysis of the simple edge test for the Lipschitz property (and monotonicity) of functions $f:\{0,1\}^d\to\mathbb{R}$ that shows that it is enough to run it for $O(d/\epsilon)$ time. Their analysis does not apply to functions on the domain $[n]^d$.

Organization. In Section 2, we present and analyze **BubbleSmooth**, our procedure for smoothing 1-dimensional functions, and prove Theorem 1.3. In Section 3, we use **BubbleSmooth** to construct a smoothing procedure for multidimensional functions that leads to the dimension reduction of Theorems 1.2 and 1.4. Our Lipschitz testers for functions on hypergrids claimed in Theorem 1.1 are presented in Section 4.

2 BubbleSmooth and its Analysis

In this section, we describe **BubbleSmooth** and prove Theorem 1.3 which asserts that **BubbleSmooth**(f) outputs a Lipschitz function that does not differ too much from f in the ℓ_1 distance. In Section 2.1, we present **BubbleSmooth** (Algorithm 1) and show that it outputs a Lipschitz function. Sections 2.2 and 2.3 are devoted to proving part (2) of Theorem 1.3. At the high level, the proof follows the ideas explained in Section 1.2 (right after Theorem 1.3). In Section 2.2, we define our transfer graph (Definition 2.3) and prove its key property (Lemma 2.1). In Section 2.3, we show that the existence of a matching with a large violation score implies that f is far from Lipschitz in the ℓ_1 distance (Lemma 2.3) and complete the proof of part (2) of Theorem 1.3 by constructing such a matching in the transfer graph.

2.1 Description of BubbleSmooth and Proof of Part (1) of Theorem 1.3

We begin this section by recalling two basic definitions from [7].

Definition 2.1 (Violation score). *Let f be a function and x, y be points in its domain. The pair (x, y) is* violated *by f if $|f(x)-f(y)| > |x-y|_1$. The* violation score *of (x, y), denoted by $\mathrm{vs}_f(x, y)$, is $|f(x) - f(y)| - |x - y|_1$ if it is violated and 0 otherwise.*

Definition 2.2 (Basic operator). *Given $f : [n]^d \to \mathbb{Z}$ and $x, y \in [n]^d$, where $|x-y|_1 = 1$ and vertex names x and y are chosen so that $f(x) \le f(y)$, the basic operator $\mathbb{B}_{x,y}$ works as follows: If the pair (x, y) is not violated by f then $\mathbb{B}_{x,y}[f]$ is identical to f. Otherwise, $\mathbb{B}_{x,y}[f](x) = f(x) + 1$ and $\mathbb{B}_{x,y}[f](y) = f(y) - 1$.*

In this section, we view a function $f : [n] \to \mathbb{Z}$ as an integer-valued sequence $f(1), f(2), \ldots, f(n)$. We denote the subsequence $f(i), f(i+1), \ldots, f(j)$ by $f[i..j]$. Naturally, a sequence $f[i..j]$ is *Lipschitz* if $|f(k) - f(k+1)| \le 1$ for all $i \le k \le j-1$. Algorithm 1 presents a formal description of **BubbleSmooth**.

Algorithm 1: BubbleSmooth (Input: an integer sequence $f[1 \ldots n]$)

for $i = n-1$ *to* 1 **do**
  `// Start phase` i.
  while $|f(i) - f(i+1)| > 1$ **do** `//` $(i, i+1)$ `is violated by` f
    LinePass(i).
return f

Algorithm 2: LinePass (Input: integer i)

for $j = i$ *to* $n-1$ **do**
  $f \leftarrow \mathbb{B}_{j,j+1}[f]$. `// Apply basic operator (see Definition 2.2.)`

We start analyzing the behavior of **BubbleSmooth** by proving part (1) of Theorem 1.3, which states that **BubbleSmooth** returns a Lipschitz function.

Proof (of part (1) of Theorem 1.3). Consider an integer sequence $f[1..n]$ and let $f'[1..n]$ be the sequence returned by **BubbleSmooth**(f). We prove that f' is Lipschitz by induction on the phase of **BubbleSmooth**. Initially, $f(n)$ is vacuously Lipschitz. We fix $i \in [n]$, assume $f[i+1..n]$ is Lipschitz at the beginning of phase i and show this phase terminates and that $f[i..n]$ is Lipschitz at the end of the phase.

Consider an execution of **LinePass**(i). Assume $f[i+1..n]$ is Lipschitz in the beginning of this execution. Let j be the index, such that at the beginning of the execution, $f[i..j]$ is the longest strictly monotone sequence starting from $f(i)$. Then **LinePass**(i) modifies two elements: $f(i)$ and $f(j)$. If $f(i) > f(j)$ then $f(i)$ is decreased by 1 and

$f(j)$ is increased by 1, i.e., 1 unit is *transferred* from i to j. Similarly, if $f(i) < f(j)$ then 1 unit is transferred from j to i. It is easy to see that after this transfer is performed, $f[i+1..n]$ is still Lipschitz. Moreover, each iteration of **LinePass**(i) reduces the violation score of the pair $(i, i+1)$ by at least 1. Thus, phase i terminates with $f[i..n]$ being Lipschitz. □

2.2 Transfer Graph

In the proof of part (1) of Theorem 1.3, we established that for all $i \in [n]$, each iteration of **LinePass**(i) transfers one unit to or from i. We record the transfers in the *transfer graph* $T = ([n], E)$, defined next. A transfer from x to y is recorded as a directed edge (x, y). The edges of the transfer graph are ordered (indexed), according to when they were added to the graph. The edge (i, j) (resp., (j, i)) corresponding to the most resent transfer is combined with a previously added edge (j, k) (resp., (k, j)) if such an edge exists. This is done because transfers from x to y and from y to z are equivalent to a transfer from x to z. If a new edge (x, y) is merged with an existing edge (y, z), the combined edge retains the index of the edge (y, z).

Definition 2.3 (Transfer graph). *The* transfer graph $T = ([n], E)$*, where the edge set* $E = (e_1, \ldots, e_t)$ *is ordered and edges are not necessarily distinct. The graph is defined by the following procedure. Initially,* $E = \emptyset$ *and* $t = 0$*. Each new run of* ***LinePass*** *during the execution of* ***BubbleSmooth****, transfers a unit from* i *to* j *(or resp., from* j *to* i*) for some* i *and* j*. If* j *has no outgoing (resp., incoming) edge in* T*, then increment* t *by 1 and add the edge* $e_t = (i, j)$ *(resp.,* $e_t = (j, i)$*) to* E*. Otherwise, let* e_s *be an outgoing edge* (j, k) *(resp., an incoming edge* (k, j)*) with the largest index* s*. Replace* (j, k) *with* (i, k)*, i.e.,* $e_s \leftarrow (i, k)$*. (Replace* (k, j) *with* (k, i)*, i.e.,* $e_s \leftarrow (k, i)$*.) The final transfer graph is denoted by* T^**.*

As mentioned previously, the order of creation of edges is important to formalize the desired property of the transfer graph, so we need to consider the subgraphs that consist of the first s edges $e_1, \ldots, e_s$ of E.

Definition 2.4 (Degrees). *Consider a transfer graph* T *at some time during the execution of* ***BubbleSmooth****. For all* $s \in \{0, \ldots, t\}$ *its subgraph graph* T_s *is defined as* $([n], (e_1, \ldots, e_s))$*, where* $(e_1, \ldots, e_t)$ *is the ordered edge set of* T*. (When* $s = 0$*, the edge set of* T_s *is empty.) The degree of a vertex* $x \in [n]$ *of* T_s *is denoted by* $deg_s(x)$*; when* T_s *is a subgraph of the final transfer graph, it is denoted by* $deg^*_s(x)$*.*

Observe that at no point in time can a vertex in T simultaneously have an incoming and an outgoing edge because such edges would get merged into one edge.

Lemma 2.1 (Key property of transfer graph) *Let* f *be an input function given to* ***BubbleSmooth****. Then for each edge* $e_s = (x, y)$ *of the final transfer graph* T^**, the following holds:* $\mathrm{vs}_f(x, y) \geq deg^*_s(x) + deg^*_s(y) - 1$*.*

To prove this lemma, we consider each phase of **BubbleSmooth** separately and formulate a slightly stronger invariant that holds at every point during that phase.

Definition 2.5. *For all $i \in [n-1]$, let Δ_i be the degree of i in the transfer graph at the end of phase i.*

The following stronger invariant of the transfer graph directly implies Lemma 2.1.

Claim 2.2 (Invariant for phase i) *Let f be an input function given to* ***BubbleSmooth****. At every point during the execution of* ***BubbleSmooth***(f)*, for each edge $e_s = (x, y)$ of the transfer graph T,*

$$f(x) - f(y) \geq deg_s(x) + deg_s(y) - 1 + |x - y|.$$

Moreover, for each phase $i \in [n-1]$, after each execution of ***LinePass***(i)*, for each edge e_s incident on vertex i, the following (stronger) condition holds:*

if the edge $e_s = (i, j)$, i.e., it is outgoing from i, then $f(i) - f(j) \geq \Delta_i + deg_s(j) - 1 + |i - j|$;

if the edge $e_s = (j, i)$, i.e., it is incoming into i, then $f(j) - f(i) \geq \Delta_i + deg_s(j) - 1 + |i - j|$.

Observe that all transfers involving i during phase i are in the same direction: if in the beginning of the phase we have $f(i) > f(i+1)$, then all transfers are from i; if we have $f(i) < f(i+1)$ instead, then all transfers are to i. In particular, whenever an edge incident to i is added, it is not modified subsequently during phase i. So for all s, $deg_s(i)$ never exceeds Δ_i during phase i and the condition in Claim 2.2 is indeed stronger than that in Lemma 2.1. The proof of Claim 2.2 is omitted.

2.3 Matchings of Violated Pairs

Part (2) of Theorem 1.3 states that the ℓ_1 distance between f and **BubbleSmooth**(f) is at most $2 \cdot \ell_1^{Lip}(f)$. By definition of the transfer graph $T = ([n], E)$, the distance $|f - \textbf{BubbleSmooth}(f)|_1 = 2|E|$. Lemma 2.3 shows that $\ell_1^{Lip}(f)$ is bounded below by the violation score of any matching. We complete the proof of Theorem 1.3 by showing that T has a matching with violation score $|E|$.

Lemma 2.3 *Let M be a matching of pairs (x, y), where x and y are in the (discrete) domain of a function f. Then $\ell_1^{Lip}(f) \geq \mathrm{vs}_f(M)$, where $\mathrm{vs}_f(M) = \sum_{(x,y)\in M} \mathrm{vs}_f(x, y)$ is the violation score of M.*

Proof. Let f^* be a closest Lipschitz function to f (on the same domain as f) with respect to the ℓ_1 distance, i.e., $|f - f^*|_1 = \ell_1(f, Lip)$. Consider a pair $(x, y) \in M$. Since $|f(x) - f(y)| = d(x, y) + \mathrm{vs}_f(x, y)$ and $|f^*(x) - f^*(y)| \leq d(x, y)$, it follows by the triangle inequality that $|f(x) - f^*(x)| + |f(y) - f^*(y)| \geq \mathrm{vs}_f(x, y)$. Since M is a matching, we can add over all of its pairs to obtain

$$\begin{aligned} \ell_1(f, Lip) = |f - f^*|_1 &\geq \sum_{(x,y)\in M} (|f(x) - f^*(x)| + |f(y) - f^*(y)|) \\ &\geq \sum_{(x,y)\in M} \mathrm{vs}_f(x, y) = \mathrm{vs}_f(M), \end{aligned}$$

which concludes the proof. □

Now using Lemma 2.1 we exhibit a matching in the final transfer graph which has large violation score, concluding the proof of Theorem 1.3.

Proof (of part (2) of Theorem 1.3). Let $T^* = ([n], E)$ be the final transfer graph corresponding to the execution of **BubbleSmooth** on f and let $E = \{e_1, \ldots, e_t\}$. By definition of the transfer graph, $|f - f'|_1 = \sum_{i \in [n]} deg_t(i) = 2|E|$. By Lemma 2.3, it is enough to show that there is a matching M of pairs violated by f with the violation score $\mathrm{vs}_f(M) \geq |E|$.

We claim that T contains such a matching. It can be constructed greedily by repeating the following step, starting with $s = t$: add e_s to M and then remove e_s and all other edges adjacent to its endpoints from T; set s to be the number of edges remaining in E. In each step, at most $deg_s(x) + deg_s(y) - 1$ are removed from T. ("At most" because T can have multiple edges.) By Lemma 2.1, $\mathrm{vs}_f(x, y) \geq deg_s(x) + deg_s(y) - 1$. So, at each step of the greedy procedure, the violation score of the pair (x, y) added to M is at least the number of edges removed from T. Therefore, $\mathrm{vs}_f(M) \geq |E|$. □

3 Dimension Reduction: Proof of Theorems 1.2 and 1.4

In this section, we explain the main ideas used to prove Theorems 1.2 and 1.4 that connect the distance of a function to being Lipschitz to the distance of its lines to being Lipschitz. Effectively, these results reduce the task of testing a multidimensional function to the task of testing its lines. Our main contribution in this section is a smoothing procedure that makes a function Lipschitz by modifying it at a few points by repairing one dimension at a time. In Definition 3.1, we present the *dimension operator* that repairs all lines in a specified dimension by applying **BubbleSmooth** to each of them. The important properties of the dimension operator are summarized in Lemma 3.1 which is the key ingredient in the proofs of Theorems 1.2 and 1.4. The derivation of Theorems 1.2 and 1.4 from Lemma 3.1 appears in the full version.

Recall from the discussion in Section 1.2 that we denote the set of lines of f along dimension i by L_f^i and the set of all lines of f by $L_f = L_f^i$.

Definition 3.1 (Dimension operator A_i). *Given $f : [n]^d \to \mathbb{Z}$ and dimension $i \in [d]$, the* dimension operator *A_i applies **BubbleSmooth** to every function $g \in L_f^i$ and returns the resulting function.*

Next lemma summarizes the properties of the dimension operator.

Lemma 3.1 (Properties of the dimension operator A_i) *For all $i \in [d]$, the dimension operator A_i satisfies the following properties for every function $f : [n]^d \to \mathbb{Z}$.*

1. *(Repairs dimension i.) Every $g \in L^i_{A_i[f]}$ is Lipschitz.*
2. *(Does not modify the function too much.) $|f - A_i[f]|_1 \leq 2 \cdot \sum_{g \in L_f^i} \ell_1^{Lip}(g)$.*
3. *(Does not spoil other dimensions.) For all $j \neq i$ in $[d]$, it does not increase the expected ℓ_1 distance of a random line in dimension j to the Lipschitz property, i.e., $\mathbb{E}_{g \leftarrow L^j_{A_i[f]}}[\ell_1^{Lip}(g)] \leq \mathbb{E}_{g \leftarrow L^j_f}[\ell_1^{Lip}(g)]$.*

Proof. **Item 1.** Item 1 follows from part (1) of Theorem 1.3.

Item 2. Since the dimension operator A_i operates by applying **BubbleSmooth** to all (disjoint) lines in L_f^i, we get $|f - A_i[f]|_1 = \sum_{g \in L_f^i} |g - \textbf{BubbleSmooth}[g]|_1$. The latter is at most $\sum_{g \in L_f^i} 2 \cdot \ell_1^{Lip}(g)$ by Part (2) of Theorem 1.3, thus proving the item.

Item 3. Fix i and j. First, we give a standard argument [4, 3, 7] that it is enough to prove this statement for $n \times n$ grids. Namely, every $\alpha \in [n]^d$ with $\alpha_i = \alpha_j = 0$ defines a restriction of a function f to an $n \times n$ grid by $h(x_i, x_j) = f(\alpha + x_i \cdot e^i + x_j \cdot e^j)$, where x_i and x_j range over $[n]$. (Recall that $e^i \in [n]^d$ is 1 on the ith coordinate and 0 on the remaining coordinates.) If the item holds for all 2-dimensional grids, we can average over all such grids defined by different α to obtain the statement for the d-dimensional function f. Now fix an arbitrary restriction $h : [n]^2 \to \mathbb{Z}$ as discussed and think of h as an $n \times n$ matrix with rows (resp., columns) corresponding to lines in dimension i (resp., in dimension j).

The key feature of our dimension operator A_i is that it can be broken down into steps, each consisting of one application of the basic step of **BubbleSmooth** to the same positions $(k, k+1)$ on all lines in dimension i. To see this, observe that we can replace the **while** loop condition on Line 2 of Algorithm 2 with "repeat t times", where t should be large enough to guarantee that the line segment under consideration is Lipschitz after t iterations of **LinePass**. (E.g., $t = n \cdot \mathrm{ImgD}(f)$ repetitions suffices.) If this version of **BubbleSmooth** is run synchronously and in parallel on all lines in dimension i, the the basic step will be applied to the same positions $(k, k+1)$ on all lines.

Since in each parallel update step only two adjacent columns of h are affected, it is sufficient to prove the item for two adjacent columns of h. Accordingly, consider two adjacent columns C_1 and C_2 of h. Let M_1 and M_2 be Lipschitz columns that are closest in the ℓ_1 distance to C_1 and C_2, respectively. Thus, $\ell_1^{Lip}(C_1) = |C_1 - M_1|_1$ and $\ell_1^{Lip}(C_2) = |C_2 - M_2|_1$. Let C_1' and C_2' be the columns of the matrix resulting from applying the basic operator to the rows of the matrix (C_1, C_2). Similarly, define M_1' and M_2' to be the columns of the matrix resulting from applying the basic operator to the rows of (M_1, M_2). We prove in the full version that applying the basic operator to the rows of a matrix consisting of two Lipschitz columns results in a matrix whose columns are still Lipschitz, that is, M_1' and M_2' are Lipschitz. Therefore, $\ell_1^{Lip}(C_1') \le |C_1' - M_1'|_1$ and $\ell_1^{Lip}(C_2') \le |C_2' - M_2'|_1$. Finally, using the inequality $|C_1' - M_1'|_1 + |C_2' - M_2'|_1 \le |C_1 - M_1|_1 + |C_2 - M_2|_1$ whose proof is deferred to the full version, the proof of Item 3 is completed as follows: $\ell_1^{Lip}(C_1) + \ell_1^{Lip}(C_2) = |C_1 - M_1|_1 + |C_2 - M_2|_1 \ge |C_1' - M_1'|_1 + |C_2' - M_2'|_1 \ge \ell_1^{Lip}(C_1') + \ell_1^{Lip}(C_2')$. □

4 Algorithms for Testing the Lipschitz Property on Hypergrids

In this section, we present our testers for the Lipschitz property of functions $f : [n]^d \to \mathbb{Z}$. Theorem 1.2 relates the distance of a function f from the Lipschitz property to the (expected) distance of its lines to this property. The resulting bound, however, depends on the image diameter of f. The image diameter is small (at most nd) for Lipschitz functions, but can be arbitrarily large otherwise. The high-level description of our testers is the following: *(i) estimate the image diameter of f and* ***reject*** *if it is too large; (ii)* *repeatedly sample a line g of f at random, run one step of a Lipschitz tester for the line on g and* ***reject*** *if a violated pair is discovered; otherwise,* ***accept****.* Step (i) ensures that a small sample of lines is enough to succeed with constant probability. The testers differ only in one parameter which quantifies what "too large" means in Step (i).

4.1 Estimating the Effective Image Diameter

As mentioned before, a Lipschitz function on $[n]^d$ has image diameter at most nd, which can serve as a threshold for rejection in Step (i) of the informal procedure above. However (if we are willing to tolerate two-sided error), it is sufficient to use a smaller threshold, equal the *effective* diameter of the function. For a given $\epsilon \in (0,1]$, define $\mathrm{ImgD}_\epsilon(f)$ as the smallest value α such that f is ϵ-close to having image diameter α:

$$\mathrm{ImgD}_\epsilon(f) = \min_{U\subseteq[n]^d : |U|\geq(1-\epsilon)n^d} \{\max_{x\in U} f(x) - \min_{x\in U} f(x)\}.$$

Although the image diameter of a Lipschitz function f can indeed achieve value nd, the effective $\mathrm{ImgD}_\epsilon(f)$ is upper bounded by the potentially smaller quantity $O(n\sqrt{d\ln(1/\epsilon)})$. The next lemma makes this precise, and follows directly from McDiarmid's inequality.

Lemma 4.1 (Effective image diameter) *For all $\epsilon \in (0,1]$, each Lipschitz function $f : [n]^d \to \mathbb{R}$ is $(\epsilon/21)$-close to having image diameter at most $n\sqrt{d\ln(42/\epsilon)}$.*

Our testers use estimates of image diameter or effective diameter to reject functions. The next lemma, proved in the full version, shows that we can get such estimates efficiently. An algorithm satisfying parts (i) and (ii) of the lemma was obtained in [7].

Lemma 4.2 *There is a randomized algorithm* SAMPLE-DIAMETER *that, given a function $f : [n]^d \to \mathbb{R}$ and $\epsilon \in (0,1]$, outputs an estimate $r \in \mathbb{R}$ such that: (i) $\mathrm{ImgD}_\epsilon(f) \leq r$ with probability at least $5/6$; (ii) $r \leq \mathrm{ImgD}(f)$ (always) and (iii) $r \leq \mathrm{ImgD}_{\epsilon/21}(f)$ with probability at least $2/3$. Moreover, the algorithm runs in time $O(1/\epsilon)$.*

4.2 Tester for Hypergrid Domains

Our tester for functions on hypergrids uses a tester for functions on lines from [7].

Lemma 4.3 (Full version of [7]) *Consider a function $g : [n] \to \mathbb{R}$ and $r \geq \mathrm{ImgD}(g)$. Then there is a 1-sided error algorithm* LINE-TESTER *which on input g and r rejects with probability at least $\frac{\epsilon^{Lip}(g)}{6\log\min\{r,n\}}$.*

To analyze our testers, we also need to estimate the probability that a random line $g \leftarrow L_f$ is rejected by LINE-TESTER(g, r) with $r \geq \mathrm{ImgD}_{\epsilon/2}(f)$. Such bound r will be obtained via Lemma 4.2. Since r may be much smaller than $\mathrm{ImgD}(f)$, Lemma 4.3 does not apply directly. Nevertheless, the next lemma (proved in the full version) shows how to circumvent this difficulty.

Lemma 4.4 *Let $f : [n]^d \to \mathbb{Z}$ be ϵ-far from Lipschitz. Consider a real $r \geq \mathrm{ImgD}_{\epsilon/2}(f)$. For a random line $g \leftarrow L_f$, the probability that* LINE-TESTER(g, r) *rejects is at least* $\frac{\epsilon}{24dr \log \min\{r,n\}}$.

Algorithm 3 presents our tester for the Lipschitz property on hypergrid domains. One of its inputs is a threshold R for rejection in Step 1. The testers in Theorem 1.1 are obtained by setting R appropriately.

Algorithm 3: Tester for Lipschitz property on hypergrid.

input : function $f : [n]^d \to \mathbb{Z}$, $\epsilon \in (0, 1]$, and value $R \in \mathbb{R}$

1 Let $r \leftarrow$ SAMPLE-DIAMETER$(f, \epsilon/2)$. If $r > R$, **reject**.
2 **for** $i = 1$ **to** $\ell = \frac{48d \cdot r \log \min\{r,n\}}{\epsilon}$ **do**
3   Select a line g uniformly from L_f and **reject** if LINE-TESTER(g, r) does.
4 **Accept**.

Proof (of Theorem 1.1). We claim that Algorithm 3 run with $R = nd$ (respectively, $R = n\sqrt{d \ln(84/\epsilon)}$) gives the tester in part (1) (respectively, part (2)) of Theorem 1.1. Suppose that the input function f is Lipschitz. When $R = nd$, the algorithm accepts f with probability 1; when $R = n\sqrt{d \ln(84/\epsilon)}$, Lemmas 4.2 and 4.1 guarantee that it accepts with probability at least $2/3$. Now suppose that f is ϵ-far from Lipschitz. Conditioning on the event that $r \geq \mathrm{ImgD}_{\epsilon/2}(f)$ (which holds with probability at least $5/6$ by Lemma 4.2), we get from Lemma 4.4 that f is rejected with probability at least $4/5$ in Step 3. Removing the conditioning gives that f is rejected with probability at least $2/3$ (regardless of R). Further details and the analysis of the running time are omitted. □

References

[1] Awasthi, P., Jha, M., Molinaro, M., Raskhodnikova, S.: Testing lipschitz functions on hypergrid domains. Electronic Colloquium on Computational Complexity (ECCC) TR12-076 (2012)
[2] Chakrabarty, D., Seshadhri, C.: Optimal bounds for monotonicity and Lipschitz testing over the hypercube. Electronic Colloquium on Computational Complexity (ECCC) TR12-030 (2012)
[3] Dodis, Y., Goldreich, O., Lehman, E., Raskhodnikova, S., Ron, D., Samorodnitsky, A.: Improved testing algorithms for monotonicity. In: RANDOM. pp. 97–108 (1999)
[4] Goldreich, O., Goldwasser, S., Lehman, E., Ron, D., Samorodnitsky, A.: Testing monotonicity. Combinatorica 20(3), 301–337 (2000)
[5] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. J. ACM 45(4), 653–750 (1998)
[6] Gromov, M.: Metric Structures for Riemannian and non-Riemannian Spaces (1999)
[7] Jha, M., Raskhodnikova, S.: Testing and reconstruction of Lipschitz functions with applications to data privacy. In: IEEE FOCS. pp. 433–442 (2011), full version available at http://eccc.hpi-web.de/report/2011/057/.
[8] Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. SIAM J. Comput. 25(2), 252–271 (1996)