The Power and Limitations of Uniform Samples in Testing Properties of Figures*

Piotr Berman† Meiram Murzabulatov‡ Sofya Raskhodnikova§

1st October 2018

Abstract

We investigate testing of properties of 2-dimensional figures that consist of a black object on a white background. Given a parameter $\epsilon \in (0, 1/2)$, a tester for a specified property has to accept with probability at least 2/3 if the input figure satisfies the property and reject with probability at least 2/3 if it is ϵ-far from satisfying the property. In general, property testers can query the color of any point in the input figure.

We study the power of testers that get access only to uniform samples from the input figure. We show that for the property of being a half-plane, the uniform testers are as powerful as general testers: they require only $O(\epsilon^{-1})$ samples. In contrast, we prove that convexity can be tested with $O(\epsilon^{-1})$ queries by testers that can make queries of their choice while uniform testers for this property require $\Omega(\epsilon^{-5/4})$ samples. Previously, the fastest known tester for convexity needed $\Theta(\epsilon^{-4/3})$ queries.

1 Introduction

We investigate testing of properties of 2-dimensional figures that consist of a black object and a white background. Sometimes the correctness of an algorithm depends on whether its input satisfies a certain property, e.g., it is a half-plane or a convex set. However, for a very large set, it is infeasible to determine whether it is indeed a half-plane or convex. How quickly is it possible to determine whether the input approximately satisfies the desired property? What access to the input is sufficient for this task?

Property testing [45, 26] (see also surveys on property testing [23, 42, 22] and a recent book by Goldreich [24]) studies algorithms that quickly determine whether the input has the desired property or is *far* from having it. Many types of objects have been investigated in the property testing framework, including graphs [26, 21, 1], functions [12, 25, 16, 20, 28, 40, 3, 35], distributions [6, 49, 44, 13, 24], strings [5, 32], arrays [18, 38, 34] and geometric objects [15, 14]. In this work, we study properties of 2-dimensional figures.

A *figure* (U, C) consists of a compact convex universe $U \subseteq \mathbb{R}^2$ and a measurable subset $C \subseteq U$. The set C can be thought of as a black object on a white background $U \setminus C$. A figure (U, C) is a *half-plane* if there is a line separating C from $U \setminus C$. A figure (U, C) is *convex* iff C is convex. The relative distance between two figures (U, C) and (U, C') over the same universe is the probability of the symmetric difference between them under the uniform distribution on U. A figure (U, C) is ϵ-far from a property (e.g., being a half-plane) if the relative distance from (U, C) to every figure (U, C') with the property over the same universe is at least ϵ. Otherwise, the figure is ϵ-close to the property.

Definition 1.1. *Given a proximity parameter $\epsilon \in (0, 1/2)$ and error probability $\delta \in (0, 1)$, an* ϵ-tester *for a given property accepts with probability at least $1 - \delta$ if the figure has the desired property and rejects with probability at least $1 - \delta$ if the figure is ϵ-far from the desired property*[1]*. A tester has* 1-sided error *if it always accepts inputs with the property. (Otherwise, it has 2-sided error). A tester is* nonadaptive *if it makes all of its queries in advance, before seeing any of the input. A tester is* uniform *if it accesses*

*A preliminary full version of this paper appears in Algorithmica [10], https://doi.org/10.1007/s00453-018-0467-9.

†Pennsylvania State University, USA; berman@cse.psu.edu.

‡Pennsylvania State University, USA; mzm269@cse.psu.edu. This author was supported by NSF award CCF-1422975.

§Boston University, Boston, USA; sofya@bu.edu. This author was supported by NSF award CCF-1422975.

[1] If δ is not specified, it is assumed to be 1/3. By standard arguments, the error probability can be reduced from 1/3 to an arbitrarily small δ by running the tester $O(\log 1/\delta)$ times.

Property	Previous Work	This work
Half-plane	$O\left(\frac{1}{\epsilon}\right)$ queries/running time adaptive, 1-sided error [37] --- $O\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$ samples uniform, 1-sided error implicitly follows from connection to PAC-learning in [26]	$\Theta\left(\frac{1}{\epsilon}\right)$ samples/running time uniform, 1-sided error
Convexity	$\Theta\left(\frac{1}{\epsilon^{4/3}}\right)$ samples/running time uniform, 1-sided error [8]	$\Theta\left(\frac{1}{\epsilon}\right)$ queries/running time adaptive, 1-sided error --- $\Omega\left(\frac{1}{\epsilon^{5/4}}\right)$ samples uniform, 2-sided error

Table 1: Best previously known and our bounds on query and time complexity of adaptive/uniform testing of the half-plane property and convexity.

its input only via uniform and independent samples from U, each labeled with a bit indicating whether it belongs to C.

In particular, a uniform tester is nonadaptive. In general, a tester can query the input at an arbitrary location. Such a strong assumption about the access model is not always realistic. For some classes of properties, e.g., linear properties (that is, properties that form linear spaces), adaptivity provably does not help [7], whereas for other classes of properties, e.g., graph properties in the bounded degree model, nonadaptive algorithms are not as powerful as the adaptive algorithms [41]). Uniform testers, in contrast to general adaptive testers, rely only on uniform samples from the input. One advantage of using uniform testers is that they are *universal* in the following sense: we can collect uniform samples from the data in advance, before we know what property of the data needs to be tested.

Uniform testers were first considered by Goldreich, Goldwasser, and Ron [26] and systematically studied by Goldreich and Ron [27] and Firscher, Lachish, and Vasudev [19] under the name of "sample-based testers". In particular, [27, 19] show that certain types of query-based testers yield uniform testers with sublinear (but dependent on the size of the input) sample complexity.

In the context of property testing and sublinear algorithms, visual properties of 2-dimensional figures and discretized images have been studied in [37, 36, 43, 29, 30, 31, 11, 8, 9]. In [37], *adaptive* ϵ-testers for the half-plane property and convexity were obtained. For the half-plane property, their query complexity[2] is $O(\epsilon^{-1})$ and for convexity their query complexity is $O(\epsilon^{-2})$. Currently, the best ϵ-tester known for convexity takes $O(\epsilon^{-4/3})$ samples and is uniform [8]. This tester has 1-sided error, and every uniform 1-sided error tester for convexity needs $\Omega(\epsilon^{-4/3})$ samples [8].

This motivates the following questions: What is the power of uniform samples? Specifically, can we test the half-plane property with $O(\epsilon^{-1})$ uniform samples? Are uniform testers as powerful as query-based testers for testing convexity?

1.1 Our Results

We show that for the property of being a half-plane, the uniform testers are as powerful as general testers: they require only $O(\epsilon^{-1})$ samples. This is not the case for convexity. We prove that convexity can be tested with $O(\epsilon^{-1})$ queries by testers that can make queries of their choice, improving the bound

[2]For any nontrivial property, including being a half-plane, $\Omega(\epsilon^{-1})$ is an easy lower bound on the complexity of an ϵ-tester.

of $O(\epsilon^{-4/3})$ in [8]. We also show that uniform testers for convexity, even with 2-sided error, require $\Omega(\epsilon^{-5/4})$ samples. Our results are summarized in Table 1.

1.2 Connection to Learning

An upper bound $O(\epsilon^{-1} \log \epsilon^{-1})$ on the number of uniform samples for testing the half-plane property can be obtained from a connection between (proper) PAC-learning and property testing, described in [26]. This bound follows from the fact that the VC dimension of the half-plane property is constant. Even though our tester has only slightly better sample complexity, its complexity is tight. Moreover, the running time of our tester is also optimal. For convexity, PAC-learning under the uniform distribution requires $\Theta(\epsilon^{-3/2})$ samples, as shown by Schmeltz [46]. (VC dimension of convexity is unbounded, so this result is specific to the uniform distribution.) For this property, however, as shown in [8], testing requires significantly fewer samples than learning when the object is accessed via uniform samples. Our tester for convexity can be viewed as an adaptive learner for the property, followed by a check that the learned convex object is close to the input.

1.3 Our Techniques

Our tester for the half-plane property is the natural one: it checks whether the convex hull of sampled black points intersects the convex hull of sampled white points and rejects if it is the case. In other words, it rejects only if it finds a violation of the half-plane property. To analyze the tester, we use the notion of *black-central* and *white-central* points defined in terms of the Ham Sandwich cut of black (respectively, white) points. (These central points are related to the well studied centerpoints [17] and Tukey medians [48]. The guarantee for a centerpoint is that every line that passes through it creates a relatively balanced cut.) Such cuts have been studied extensively (see, e.g., [17, p. 356] and [33]), for example, in the context of range queries. Specifically, a *black-central* (respectively, white-central) point is the intersection of two lines that partition the figure into four regions, each with black (respectively, white) area[3] at least $\epsilon/4$. Black-central points were defined in [8] in order to analyze a tester of convexity of figures. A black-central (respectively, white-central) point is overwhelmingly likely to end up in the convex hull of sampled black (respectively, white) points. We show that if the figure is ϵ-far from being a half-plane, the convex hull of its black-central points intersects the convex hull of its white-central points. A point in the intersection, even though is not likely to be sampled, is likely to be in the intersection of the convex hull of the black samples and the convex hull of the white samples. Thus, there is likely to be the intersection, and the tester is likely to reject.

Our tester for convexity samples points uniformly at random and constructs a rectangle R that with high probability contains nearly the entire black area and whose sides include sampled black points. Then it adaptively queries points of R in order to partition it into the candidate black and white regions, leaving only a small region unclassified. After completing this learning stage, it samples points in the classified regions and rejects iff it finds a mistake.

To prove our lower bound, we construct hard instances, for which every uniform tester needs to get a 2-point witness, with points coming from different specified regions, in order to distinguish between our hard instances that are convex from hard instances that are far from convex. The challenge here is to construct a figure with regions that can be manipulated independently to either keep convexity or to violate it.

2 The Uniform Tester for the Half-Plane Property

In this section, we give a uniform tester for the half-plane property.

Theorem 2.1. *There is a uniform (1-sided error) ϵ-tester for the half-plane property of figures with sample and time complexity $O(\epsilon^{-1})$ and with error probability $1/3$.*

Proof. Our uniform tester for the half-plane property is Algorithm 1. It takes $O(\epsilon^{-1})$ uniform samples and checks if the sampled black and white points are linearly separable. To do it efficiently, the algorithm computes the convex hull of sampled black points (by first sorting them and then computing the upper and the lower hulls) and then checks if it contains a sampled white point (by merging the sorted list of

[3]For the two properties we consider (being a half-plane and convexity), we assume w.l.o.g. that the input figure U has unit area. If it is not the case, U can be rescaled. Thus, the area of a region corresponds to the probability of sampling from it under the uniform distribution.

white sample with the upper and lower hulls of black samples). Similarly, it checks if the convex hull of sampled white points contains a sampled black point.

Definition 2.1 (Notation for convex hull, upper and lower hull)**.** *For a set of points S, let Hull(S), UHull(S), and LHull(S) denote its convex hull, upper convex hull, and lower convex hull, respectively.*

We will show that the expected running time of Algorithm 1 is $O(\epsilon^{-1})$ and its error probability is 0.3. A tester with the worst case running time $O(\epsilon^{-1})$ and error probability 1/3 can be obtained from Algorithm 1 by standard arguments.

Algorithm 1: Uniform tester for the half-plane property.

input : parameter $\epsilon \in (0, 1/2)$;
access to uniform and independent samples from (U, C).

1 Set $s \leftarrow \frac{18}{\epsilon}$. Sample s points from U uniformly and independently at random.
2 Let R be a rectangle of the smallest area that contains the set U. Rotate U, so that R is axis-aligned. `// The area of` R `is at most twice the area of` U `// (see, e.g., [8, Lemma A.1]) for a proof.`
3 Compute the list S_B of sampled black points sorted by the x-coordinate by using the Bucket Sort with s bins. Similarly, compute S_W for the sampled white points.

`// Check if the convex hull of` S_B `contains a point from` S_W`.`
4 Use Andrew's monotone chain convex hull algorithm [2] to compute UHull(S_B) and LHull(S_B), the upper and the lower hulls of S_B, respectively, sorted by the x-coordinate.
5 Merge sorted lists S_W, UHull(S_B) and LHull(S_B) to determine for each point w in S_W its left and right neighbors in UHull(S_B) and LHull(S_B). If w lies between the corresponding line segments of the upper and the lower hulls, **reject**.

`// Check if the convex hull of` S_W `contains a point from` S_B`.`
6 Repeat Steps 4–5 with the roles of S_B and S_W reversed.
7 **Accept**.

Consider a half-plane figure (U, C). Let S_B and S_W be the two lists obtained by Algorithm 1 in Step 3. Since the figure is a half-plane, Hull(S_B) and Hull(S_W) do not intersect, that is, they are linearly separable. Thus, the algorithm accepts the figure.

Now assume that (U, C) is ϵ-far from being a half-plane. We prove that the algorithm rejects the figure with probability at least 2/3. We consider two sets of points in U : *black-central* and *white-central*. We show that if the figure is ϵ-far from being a half-plane, then the convex hulls of the two sets intersect. In this case, the tester will detect this intersection, with probability at least 2/3, by only looking at the convex hull of sampled black points and the convex hull of sampled white points.

Next, we define white-central and black-central points. Black-central points were used in [8] to analyze a tester for convexity. In that work, they were called *central* points.

Definition 2.2 (White-central and black-central points)**.** *A point in the figure is* white-central *(respectively,* black-central*) if it is the intersection of two lines such that each of the quadrants formed by these lines has white (respectively, black) area at least $\epsilon/4$.*

Lemma 2.2. *There is no line that separates white-central points from black-central points in a figure that is ϵ-far from being a half-plane.*

Proof.: Let (U, C) be a figure that is ϵ-far from being a half-plane. For the sake of contradiction, suppose there is a line ℓ that separates white-central and black-central points in (U, C), i.e., it partitions the figure into two regions, W_ℓ and B_ℓ , such that W_ℓ contains only white-central points and B_ℓ contains only black-central points (see Figure 1). The sum of the black area in W_ℓ and the white area in B_ℓ is at least ϵ since the figure is ϵ-far from being a half-plane. W.l.o.g. assume that the black area in W_ℓ is at least $\epsilon/2$. Consider the line ℓ' that is parallel to ℓ and such that the black area in one of the two half-planes defined by ℓ' is equal to $\epsilon/2$. (See Figure 2. Note that the black area in the other half-plane is at least $\epsilon/2$.) Clearly, ℓ' lies in W_ℓ. Next, we show that there is a black-central point on ℓ', i.e., in

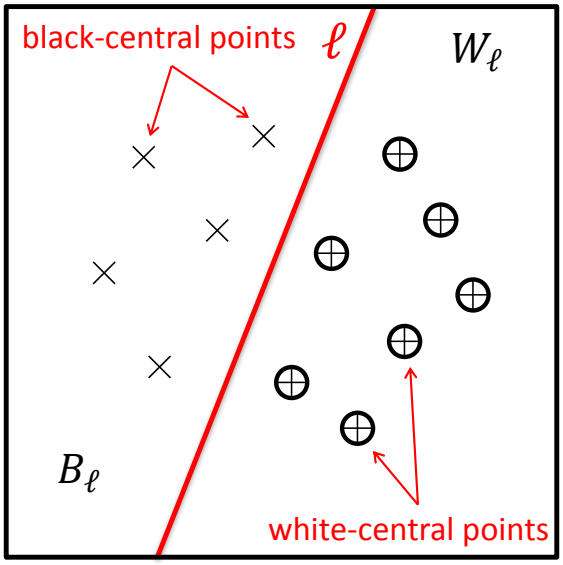

Figure 1: An illustration of black-central and white-central points separated by a line.

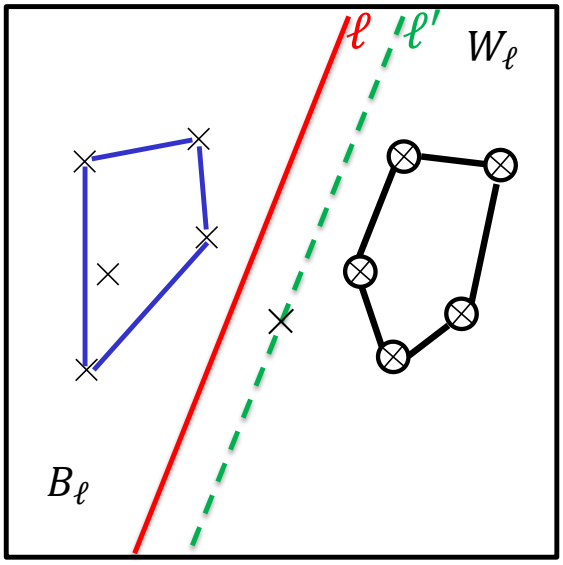

Figure 2: An illustration of the line ℓ' and a black-central point on it.

W_ℓ, thus arriving at a contradiction. Consider the two sets of black points, on either side of ℓ'. We have argued that each of them has area at least $\epsilon/2$. By the Ham Sandwich Theorem (in two dimensions, also known as the Pancake theorem), applied to the two sets, there is a line ℓ'' that bisects the two sets simultaneously, forming four sets black points of area at least $\epsilon/4$ each. The intersection point of ℓ' and ℓ'' is black-central and lies in W_ℓ. This is a contradiction, since ℓ is a line that separates white-central and black-central points. □ □

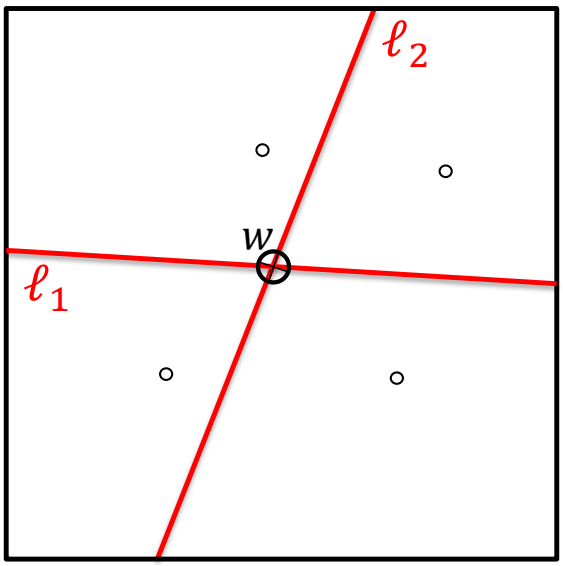

Figure 3: An illustration of a captured white central point.

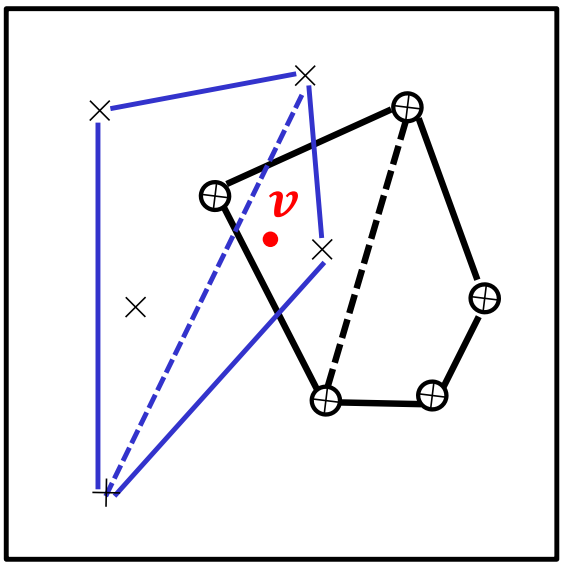

Figure 4: An illustration of the point v in the intersection of two convex hulls.

Consider a white-central point w which is the intersection of two lines ℓ_1 and ℓ_2, as shown in Figure 3. If four white points from four different quadrants determined by ℓ_1 and ℓ_2 are sampled by Algorithm 1, we say that the tester *captures* w. We define capturing a black-central point analogously.

By Lemma 2.2, the convex hull of all white-central points and the convex hull of all black-central

points intersect. Thus, there is a point v that lies in both convex hulls (see Figure 4). Moreover, there exists a set P_W of at most three white-central points such that point v lies in the convex hull of the points in P_W. Analogously, there exists a set P_B of at most three black-central points such that the point v lies in the convex hull of the points in P_B. If all points in $P_W \cup P_B$ are captured then v simultaneously lies in the convex hull of black samples and in the convex hull of white samples, i.e., the convex hull of black samples and the convex hull of white samples intersect, and the tester rejects the figure. The probability that the tester fails to capture a specific point in $P_W \cup P_B$ is, by the union bound, at most $4 \cdot (1-\epsilon/4)^{18/\epsilon} \leq 4 \cdot e^{-18/4}$. The probability that the tester fails to capture at least one point in $P_W \cup P_B$ is at most

$$6 \cdot 4 \cdot e^{-18/4} < 0.3.$$

Therefore, the failure probability of the tester is at most 0.3.

Sample and Time Complexity. Algorithm 1 samples $s = O(\epsilon^{-1})$ points. Next, we analyze its running time. Conduct the following mental experiment: Suppose we sample points from the rectangle R (defined in Algorithm 1) uniformly and independently at random until we collect s points from U; then we bucket sort sampled points by their x-coordinate into s bins. Let q be the number of points we sample. Then $\mathbb{E}[q] \leq 2s$. Since the x-coordinates of the sampled q points are distributed uniformly in the interval corresponding to the length of the rectangle R, they can be sorted in expected time $O(q)$ by subdividing this interval into s subintervals of equal length, and using them as buckets in the bucket sort. Thus, the expected running time of the algorithm in the mental experiment is $O(s)$.

Observe that Algorithm 1 has the same distribution on the s points sampled from U as the algorithm in the mental experiment. It sorts two (disjoint) subsets of the points sampled in the mental experiment. Thus, the expected running time of Step 3 of Algorithm 1 is $O(s)$. Recall the lists S_W and S_B defined in Step 3 of Algorithm 1. Andrew's monotone chain algorithm finds the upper and the lower hulls of a set of s sorted points in time $O(s)$. Thus, the lists UHull(S_W), LHull(S_W), UHull(S_B), and LHull(S_B) are found in time $O(s)$. Merging the lists S_W, UHull(S_B), and LHull(S_B) (and the lists S_B, UHull(S_W), and LHull(S_W)) in Step 5 also takes $O(s)$ time. In Step 5, checking whether a point lies between two line segments takes constant time. Overall, Algorithm 1 runs in expected time $O(s) = O(\epsilon^{-1})$. By standard arguments, we get a uniform tester with the worst case running time $O(\epsilon^{-1})$ and with a slightly larger error probability δ than in Algorithm 1, specifically, with $\delta = 1/3$. □ □

3 The Adaptive Tester for Convexity

In this section, we prove Theorem 3.1 that gives our adaptive convexity tester and state and prove Corollary 3.4 that gives our adaptive convexity learner.

Theorem 3.1. *Given $\epsilon \in (0, 1/2)$, convexity of a figure (U, C) can be ϵ-tested (adaptively) with 1-sided error in time $O(\epsilon^{-1})$ and with error probability $1/3$.*

Proof.: In [8] (see the proof of Theorem 3.1 in [8]), it was shown that testing convexity of figures (U, C) can be reduced to the special case when the universe U is an axis-aligned rectangle of unit area. Moreover, we can rescale the figure to obtain a unit square background from a unit rectangle background without affecting the distance of the figure to convexity. Therefore, we can assume w.l.o.g. that U is an axis-aligned unit square.

Our ϵ-tester for convexity (Algorithm 2) samples points uniformly at random and constructs a rectangle R that with high probability contains nearly the entire black area and whose sides include sampled black points. (See Figure 5.) Then it adaptively queries points of R in order to partition it into regions B, W and F. (See Figure 6.) To do it, the algorithm investigates the boundary of the set C by performing a walk of step size $\epsilon/12$. The walk is performed separately on the four corners of the rectangle R. To investigate the upper right corner, the algorithm performs a walk starting from a sampled black point on the top side of R until it reaches the right side of R as follows:

i) the algorithm starts by taking steps to the right;
ii) whenever a white point is detected during the walk, the algorithm changes the direction of the walk and goes down;
iii) whenever a black point is detected during the walk, the algorithm changes the direction of the walk and goes right.

Then, similarly, the algorithm investigates the remaining corners. The set B is the convex hull of all black points discovered during the walk. The walk defines squares that intersect the boundary of C and whose side length is equal to $\epsilon/12$. The set F is the "fence" region outside B formed by all such $(\epsilon/12) \times (\epsilon/12)$ squares. The set W is the remaining region in R.

As we will show later, the "fence" region F has a small area. If the image is convex, B is entirely black and W is entirely white. The algorithm queries a small number of random points in $B \cup W$ and rejects if it finds a misclassified point (i.e., a white point in B or a black point in W); otherwise, it accepts.

At the high level, since the black area outside R and the area of F are small, if the figure is ϵ-far from convexity then there are enough misclassified points in $B \cup W$, and the algorithm detects at least one of them with high probability.

Algorithm 2: *ϵ-tester* for convexity.

input : parameter $\epsilon \in (0, 1/2)$; access to a figure (U, C), where U is a unit square

1 Query $\frac{64}{\epsilon}$ points uniformly at random. If all sampled points are white, **accept**.

2 Let R be the smallest axis-parallel rectangle that contains all sampled black points. Let p_0 (respectively, p_1, p_2, p_3) be a sampled black point on the top (respectively, left, bottom, right) side of R.

3 **for** $i \leftarrow 0$ **to** 3 **do**

4 Let $(x, y) \leftarrow p_i$ and $P_i \leftarrow \emptyset$. `// Investigate the upper right corner of` R`.`
`// In line 8, we rotate` R `to reuse lines 4-8 of the`
`// pseudocode for investigating all four corners.`

5 **while** (x, y) *is in* R **do**

6 **if** (x, y) *is black or below the line through* p_i *and* $p_{(i+3) \bmod 4}$ **then**
$x \leftarrow x + \epsilon/12$. `// Move right.`

7 **else**
$P_i \leftarrow P_i \cup \{(x, y)\}$; $y \leftarrow y - \epsilon/12$. `// Move down.`

8 Let $W_i \leftarrow \{(u, v)$ inside $R \mid \exists (x, y) \in P_i$ such that $u \geq x, v \geq y$ with respect to the rotated coordinates$\}$. Rotate R clockwise by 90 degrees.

9 Let B be the convex hull of all black points discovered after Step 3 and $W \leftarrow \cup_{i=0}^{3} W_i$.

10 Query $\frac{8}{\epsilon}$ points uniformly at random. If a white point in B or a black point in W is detected, **reject**; otherwise, **accept**.

Now we prove that Algorithm 2 satisfies Theorem 3.1. First, we argue that Algorithm 2 always accepts if its input is a convex figure. If (U, C) has no black points (i.e., $C = \emptyset$), Step 1 always accepts. Otherwise, all points in B are black, by convexity of (U, C). We will show that all points in W are white.

Definition 3.1 (Triangle notation)**.** *We use notation $\triangle p_1 p_2 p_3$ to denote the triangle with vertices p_1, p_2, and p_3.*

For the sake of contradiction, suppose there is a black point $b = (u, v)$ in W_0. By definition of W_0, there is a white point $w = (x, y)$ in P_0 such that $u \geq x$ and $v \geq y$. Thus, the white point w is inside the triangle $\triangle p_0 b p_3$, formed by three black points, contradicting convexity of (U, C). Thus, there are no black points in W_0. Analogously, there are no black points in W_1, W_2 and W_3. Since there are no white points in B and no black points in $W = \cup_{i=0}^{3} W_i$, Step 10 of Algorithm 2 always accepts (U, C).

Now assume that (U, C) is ϵ-far from convex.

Lemma 3.2. *The probability that the black area outside the rectangle R (defined in Step 2 of Algorithm 2) is greater than $\frac{\epsilon}{4}$ is at most $1/9$.*

Proof.: Let L be a horizontal line with the largest y-coordinate such that the black area of the figure above L is at least $\frac{\epsilon}{16}$. The probability that no black points above L are sampled in Step 1 of Algorithm 2 (and, consequently R lies below L) is at most $(1 - \frac{\epsilon}{16})^{64/\epsilon} \leq e^{-4} < 1/36$. Thus, with probability at most $1/36$, the black area in the half-plane above R is greater than $\frac{\epsilon}{16}$. The same bound holds for the half-planes to the left, to the right and below R. By a union bound, the probability that the black area outside R is greater than $\frac{\epsilon}{4}$ is at most $(1/36) \cdot 4 = 1/9$. □ □

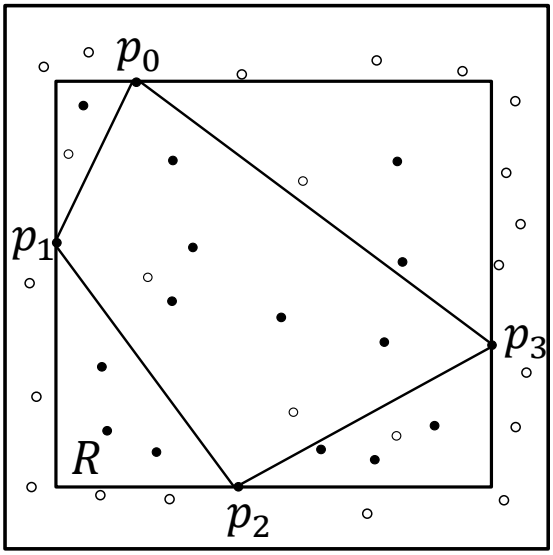

Figure 5: An illustration of Step 2 of Algorithm 2.

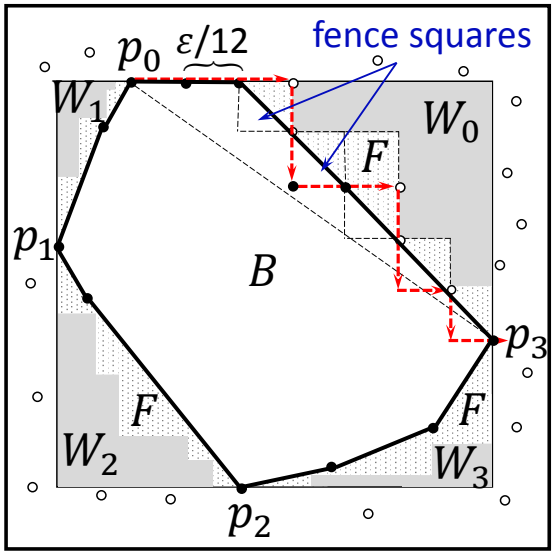

Figure 6: An illustration of Steps 3–9 of Algorithm 2.

Lemma 3.3. *Let $F = R - (B \cup W)$. Then the area of F is at most $\frac{\epsilon}{2}$.*

Proof.: Let $\gamma = \epsilon/12$ and (x_i, y_i) be p_i (as defined in Step 2 of Algorithm 2) for $i \in \{0, 1, 2, 3\}$. Call every region that consists of points $(x, y) + [0, \gamma]^2$ a *square*, where $\frac{x-x_i}{\gamma}, \frac{y-y_i}{\gamma} \in \mathbb{N}$. Call squares that contain points from F *fence squares*. Let $r = (x_3, y_0)$ and let $T = \triangle p_0 p_3 r$. We will find an upper bound on the number of fence squares inside T. Each point that Algorithm 2 queries in Step 5 results in at most one (new) fence square in T. The algorithm queries at most $\frac{x_3 - x_0 + y_0 - y_3}{\epsilon/12} + 2$ points in the triangle (thus, it discovers at most that many fence squares), since, in every iteration, it either increases the x-coordinate or decreases the y-coordinate of the queried point. Therefore, there are at most $\frac{x_3 - x_0 + y_0 - y_3}{\epsilon/12} + 2$ fence squares in this triangle. Similarly, we can find an upper bound on the number of discovered fence squares in the remaining triangles. Since the perimeter of R is at most 4, the sum of the upper bounds is at most $\frac{4}{\epsilon/12} + 8 = \frac{48}{\epsilon} + 8 \leq \frac{56}{\epsilon}$. The area of a single fence square is $(\frac{\epsilon}{12})^2 = \frac{\epsilon^2}{144}$ and thus the total area of F is at most $\frac{\epsilon^2}{144} \cdot \frac{56}{\epsilon} \leq \frac{\epsilon}{2}$. □ □

We call a point *misclassified* if it is black and belongs to W or if it is white and belongs to B. Let E denote the event that the black area outside R is less than $\epsilon/4$. By Lemma 3.2, the probability that E does not happen is at most $1/9$. Now assume that E happens. Recall that by Lemma 3.3, the area of F is at most $\epsilon/2$. Note that we obtain a convex figure if we make all area in B black and all area outside B white. Thus, since the figure (U, C) is ϵ-far from convex, the area of misclassified points in $B \cup W$ is at least $\epsilon/4$. Consequently, the probability that Step 10 of the algorithm fails to catch a misclassified point in $B \cup W$ is at most $(1 - \frac{\epsilon}{4})^{\frac{8}{\epsilon}} < e^{-2} < 2/9$. Therefore, the probability that the algorithm fails to detect nonconvexity of the figure is at most $1/9 + 2/9 = 1/3$, as desired.

Query Complexity. The algorithm queries points in Steps 1, 6 and 10. In Steps 1 and 10, the algorithm makes $O(\epsilon^{-1})$ queries. In Step 6, over all iterations, the algorithm also queries $O(\epsilon^{-1})$ points. Thus, the overall query complexity of the algorithm is $O(\epsilon^{-1})$.

Running Time. We show that the running time of Algorithm 2 is $O(\epsilon^{-1})$. By following the algorithm, one can see that the running time of Steps 1-8 is $O(\epsilon^{-1})$. We explain why Steps 9 and 10 run in time $O(\epsilon^{-1})$. Note that all black points used in Step 9 to compute B are sorted according to both x and y coordinates and there are $O(\epsilon^{-1})$ of them. Thus, in Step 9, we can use Andrew's algorithm [2] to compute B in time $O(\epsilon^{-1})$. Given a sampled point p from Step 10, in time $O(1)$ we identify whether it belongs to the quadrilateral $Q = p_0 p_1 p_2 p_3$ or to one of the triangles that form the region $R \setminus Q$ (see Figure 5). If p is inside Q then it is also in B. Now suppose p is inside the upper right triangle of the

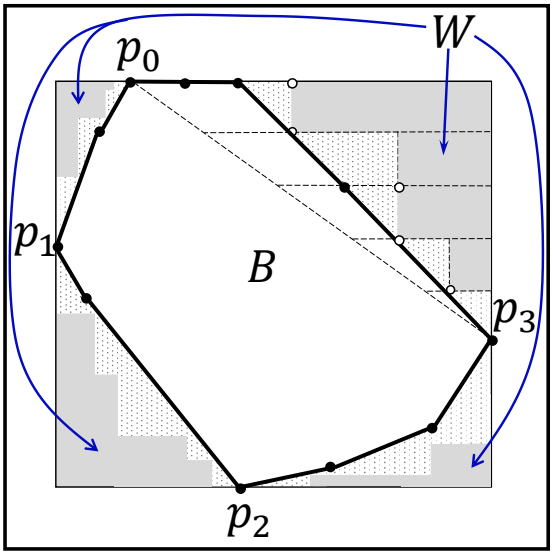

Figure 7: An illustration of triangle T partitioned into strips.

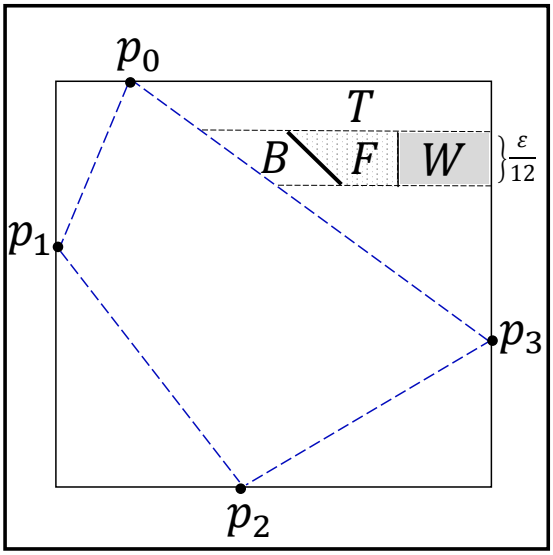

Figure 8: An illustration of one strip inside triangle T.

region $R \setminus Q$. Denote this triangle by T. Starting from the horizontal side of T consider a partition of T into horizontal strips with width $\epsilon/12$. Note that there are $O(\epsilon^{-1})$ such strips. Moreover, for each horizontal strip in the partition of T, there is one vertical line that defines the boundary between F and W and one line that defines the boundary between F and B. See Figures 7 and 8. For each strip, in time $O(1)$ we can compute and store such two lines while constructing B and W. In time $O(1)$ we identify the horizontal strip that point p belongs to. For each line ℓ computed and stored for this strip, in time $O(1)$ we identify the half-plane defined by ℓ that contains point p. Thus, in time $O(1)$ we determine whether p is in B or W or neither. Since we sample $O(\epsilon^{-1})$ points in Step 10, its running time is $O(\epsilon^{-1})$. Therefore, the running time of the algorithm is $O(\epsilon^{-1})$, as claimed. □ □

Corollary 3.4. *Given $\epsilon \in (0, 1/2)$, one can learn a convex figure (U, C) (specifically, obtain a figure that is ϵ-close to the input figure and is contained in the input figure) with $O(\epsilon^{-1})$ (adaptive) queries and in time $O(\epsilon^{-1})$ and with probability at least $2/3$.*

Proof.: Algorithm 2 constructs a convex set B in Steps 3-9 for a convex figure (U, C) in time $O(\epsilon^{-1})$. The set B is contained in C because C is convex and B is a convex hull of points in C. By Lemmas 3.2 and 3.3, the area of $C \setminus B$ is at most $\epsilon/4 + \epsilon/2 < \epsilon$ with probability at least $1 - 1/9 > 2/3$. □ □

4 The Lower Bound for Nonadaptive Convexity Testers

4.1 Preliminaries on Poissonization

The proof of our lower bound uses a technique called *Poissonization* [47]. In the context of property testing, this technique was first used in the analysis of properties of distributions by Batu et al. [4] (conference version only) and Raskhodnikova et al. [39]. It has also been used in the context of testing properties of figures in [8, 9]. Poissonization consists of modifying a probabilistic experiment by replacing a fixed quantity (e.g., the number of samples) with a variable one that follows a Poisson distribution. This breaks up dependencies between different events and makes the analysis tractable. The Poisson distribution with parameter $\lambda \geq 0$, denoted $\mathrm{Po}(\lambda)$, generates each value $x \in \mathbb{N}$ with probability $\frac{e^{-\lambda}\lambda^x}{x!}$. The expectation and variance of a random variable distributed according to $\mathrm{Po}(\lambda)$ are both λ.

Definition 4.1. *A* Poisson-s tester *is a uniform tester that takes a random number of samples distributed as* $\mathrm{Po}(s)$.

Lemma 4.1 (Poissonization Lemma [39, Lemma 5.3] and [8])**.**

(a) *Poisson algorithms can simulate uniform algorithms. Specifically, for every uniform tester $\mathcal{A}$ for property $\mathcal{P}$ that uses at most s samples and has error probability δ, there is a Poisson-$2s$ tester $\mathcal{A}'$ for $\mathcal{P}$ with error probability at most $\delta + 4/s$. Moreover,*

(b) *Let Ω be a sample space from which a Poisson-s algorithm makes uniform draws. Suppose we partition Ω into sets $\Omega_1, \ldots, \Omega_k$ (e.g., these sets can correspond to disjoint areas of the figure from which points are sampled), where each outcome is in set Ω_i with probability p_i for $i \in [k]$. Let X_i be the total number of samples in Ω_i seen by the algorithm. Then X_i is distributed as $\mathrm{Po}(p_i \cdot s)$. **Moreover, random variables X_i are mutually independent for all $i \in [k]$.***

4.2 The Lower Bound

Theorem 4.2. *Every 2-sided error uniform ϵ-tester for convexity requires $\Omega(\epsilon^{-5/4})$ samples.*

Proof.: By the Poissonization Lemma (Lemma 4.1), it is enough to prove the lower bound for Poisson algorithms. For sufficiently small ϵ, we define distributions $\mathcal{P}$ and $\mathcal{N}$ on figures, where $\mathcal{P}$ is supported only on convex figures whereas $\mathcal{N}$ is supported only on figures which are ϵ-far from convex. We show that every uniform Poisson-s tester, where $s = o(\epsilon^{-5/4})$, fails to distinguish $\mathcal{P}$ from $\mathcal{N}$ with sufficient probability.

Let $k = \lceil \frac{1}{2} \cdot \epsilon^{-1/2} \rceil$ and the universe $U = [0,1]^2$. Consider two regular convex k-gons G_1 and G_2, centered at $(1/2, 1/2)$, such that G_1 has side length $\sin(\frac{\pi}{k})$ and the vertices of G_2 are the midpoints of the sides of G_1 (see Figure 9). Call triangular regions inside G_1 but outside G_2 *teeth* (one such triangular region is a *tooth*). Let T be a tooth and b be its vertex which is also a vertex of G_1. Let the other two vertices of T be d and d' and let b_0 be a point on dd' such that bb_0 is the height of T from b to its base dd'. Call $\triangle bb_0d$ and $\triangle bb_0d'$ *half-teeth* (see Figure 10). Distributions $\mathcal{P}$ and $\mathcal{N}$ are defined next.

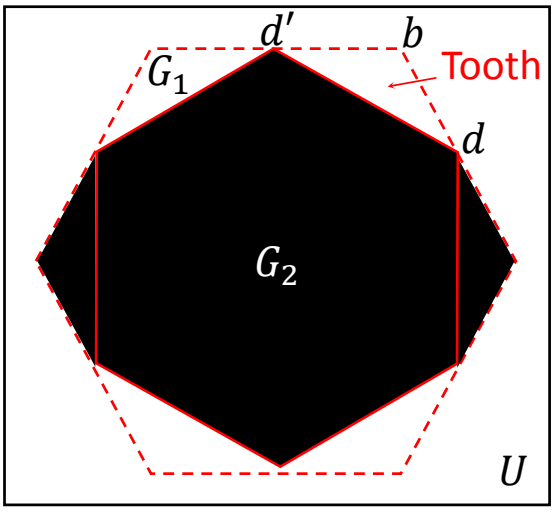

Figure 9: A figure from $\mathcal{P}$ for $k = 6$.

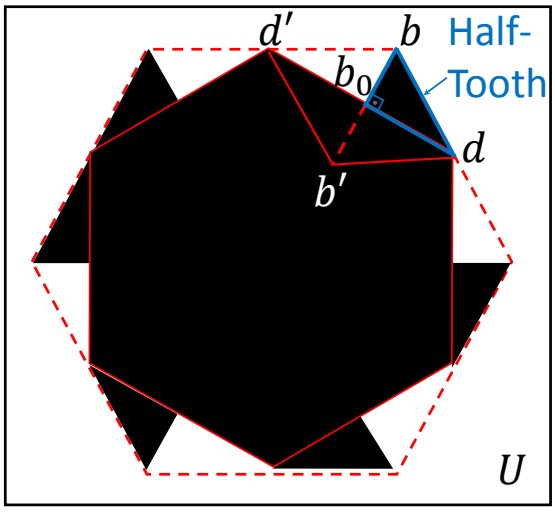

Figure 10: A figure from $\mathcal{N}$ for $k = 6$.

1. For all figures from both distributions, points outside G_1 are white and points in G_2 are black.
2. For a figure in $\mathcal{P}$, every tooth is independently colored white or black, each with probability $1/2$, as shown in Figure 9.
3. For a figure in $\mathcal{N}$, every tooth is independently colored as follows: one half-tooth is colored black or white, each with probability $1/2$, and the other half-tooth gets the opposite color, as shown in Figure 10.

Note that every figure in the support of $\mathcal{P}$ is convex.

Lemma 4.3. *For all $\epsilon \leq 3 \cdot 10^{-3}$, every figure in the support of $\mathcal{N}$ is ϵ-far from convex.*

Proof.: Consider a figure (U, C) in the support of $\mathcal{N}$. Let $\triangle bdd'$ be a tooth of (U, C). Consider the point b' that is symmetric to b with respect to the line dd', as shown in Figure 10. Call the quadrilateral $bb'dd'$ a *block*. Observe that there are k disjoint blocks. Let (U, C') be a convex figure that is closest to (U, C).

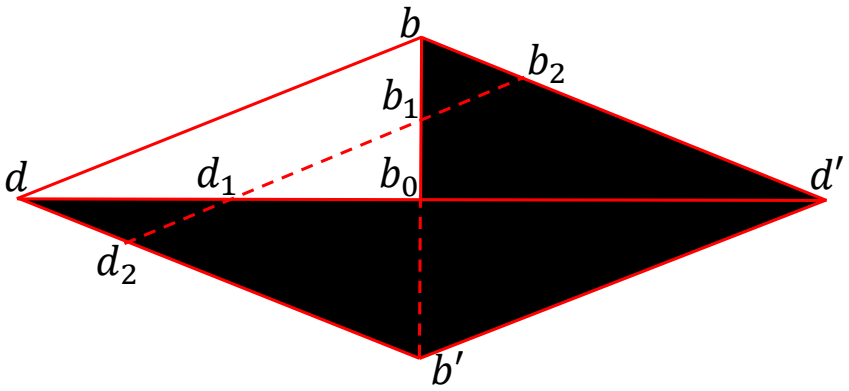

Figure 11: An illustration of a block.

Definition 4.2 (Notation for the area of a tooth)**.** *The area of a tooth is denoted* $A_{\triangle}$.

Claim 4.4. *In every block of* C*, an area at least* $\frac{A_{\triangle}}{16}$ *must be modified to obtain* C' *from* C.

Proof.: For a region R, let $A(R)$ denote the area of R.

Consider the block $bdb'd'$ illustrated in Figure 11. Let b_1 and d_1 be the midpoints of bb_0 and db_0, respectively. Let the line b_1d_1 intersect bd' and db' at b_2 and d_2, respectively.

Consider the white triangle $\triangle b_1b_0d_1$ and the three black triangles $\triangle dd_1d_2$, $\triangle bb_1b_2$, and $\triangle b_0b'd'$. If there is a point in each of these four triangles that has not changed color, then we have a white point in the convex hull of three black points, i.e., the figure is not convex. Therefore, in at least one of these four triangles, all points must change color in order to make the figure convex. Since the areas of the triangles are

$$A(\triangle b_0b'd') = \frac{A_{\triangle}}{2}, A(\triangle b_1b_0d_1) = \frac{A_{\triangle}}{8}, A(\triangle dd_1d_2) = A(\triangle bb_1b_2) = \frac{A_{\triangle}}{16},$$

the claim holds. □ □

Claim 4.5. $5.6 \cdot \frac{1}{k^3} < A_{\triangle} \leq 8 \cdot \frac{1}{k^3}$.

Proof.: Consider a tooth $\triangle bdd'$ (see Figure 9). Recall that the side length of the regular k-gon G_1 is $\sin(\frac{\pi}{k})$. Therefore,

$$|bd| = |bd'| = \frac{1}{2} \cdot \sin\left(\frac{\pi}{k}\right).$$

Let $\angle dbd'$ denote the angle of $\triangle bdd'$ at the vertex b. Note that

$$\angle dbd' = \frac{\pi \cdot (k-2)}{k} = \left(\pi - \frac{2\pi}{k}\right)$$

and that

$$A_{\triangle} = \frac{1}{2} \cdot |bd| \cdot |bd'| \cdot \sin\left(\angle dbd'\right) = \frac{1}{2} \cdot \frac{1}{4} \cdot \sin^2\left(\frac{\pi}{k}\right) \cdot \sin\left(\pi - \frac{2\pi}{k}\right) =$$

$$= \frac{1}{8} \cdot \sin^2\left(\frac{\pi}{k}\right) \cdot \sin\left(\frac{2\pi}{k}\right).$$

Since $0.9x \leq \sin x \leq x$ for $x \in [0, 0.78]$, we obtain that, for sufficiently large k (i.e., for $\epsilon \leq 3 \cdot 10^{-3}$),

$$A_{\triangle} \geq \frac{1}{8} \cdot \left(0.9 \cdot \frac{\pi}{k}\right)^2 \cdot \left(0.9 \cdot \frac{2\pi}{k}\right) > 5.6 \cdot \frac{1}{k^3};$$

$$A_{\triangle} \leq \left(\frac{1}{8}\right) \cdot \left(\frac{\pi}{k}\right)^2 \cdot \left(\frac{2\pi}{k}\right) \leq \frac{8}{k^3}.$$

□ □

There are k blocks and, by Claim 4.4, at least

$$k \cdot \frac{A_\triangle}{16} > k \cdot \frac{5.6}{16} \cdot \frac{1}{k^3} = \frac{7}{20} \cdot \frac{1}{k^2} \geq \epsilon$$

area needs to be modified to make C convex. (Recall that $k = \lceil \frac{1}{2} \cdot \epsilon^{-1/2} \rceil$.) This completes the proof of Lemma 4.3. □ □

Consider a Poisson-s algorithm $\mathcal{A}$ with $s = c_0 \cdot \epsilon^{-5/4}$. We will show that when c_0 is sufficiently small then $\mathcal{A}$ fails on $\mathcal{P}$ or $\mathcal{N}$ with probability greater than 1/3.

Definition 4.3. *A pair of points (p_1, p_2) is called a* red-flag pair *if p_1 and p_2 belong to different half-teeth of the same tooth.*

Let BAD denote the event that no red-flag pair is sampled by the algorithm $\mathcal{A}$.

Claim 4.6. *If c_0 is sufficiently small,* $\Pr[\overline{BAD}] < 1/10$.

Proof.: Let L_T and R_T be the random variables that count the number of points sampled by the tester in the left half-tooth and in the right half-tooth of a tooth T, respectively. Let X_T and X be the random variables that count the number of sampled red-flag pairs in a tooth T and in all teeth, respectively. By the Poissonization Lemma (Lemma 4.1), L_T and R_T are independent Poisson random variables with expectation $(A_\triangle/2) \cdot s$. Note that $X_T = L_T \cdot R_T$ and, therefore,

$$\mathbb{E}[X_T] = \mathbb{E}[L_T] \cdot \mathbb{E}[R_T] = (A_\triangle/2)^2 \cdot s^2 \leq \frac{16s^2}{k^6},$$

by Claim 4.5. Since all teeth are disjoint, then for sufficiently small c_0,

$$\mathbb{E}[X] = k \cdot \mathbb{E}[X_T] \leq k \cdot \frac{16s^2}{k^6} \leq 512 \cdot c_0^2 < 1/10.$$

By Markov's inequality, $\Pr[\overline{BAD}] = \Pr[X \geq 1] \leq \mathbb{E}[X] < 1/10$. □ □

Conditioned on BAD, the distribution on the answers to the queries made by $\mathcal{A}$ is the same whether the input is sampled from $\mathcal{P}$ or $\mathcal{N}$. Therefore,

$$\begin{aligned}&\Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x \mid BAD] \\ &\qquad = \Pr_{x\sim\mathcal{N}}[\mathcal{A} \text{ accepts } x \mid BAD] = 1 - \Pr_{x\sim\mathcal{N}}[\mathcal{A} \text{ rejects } x \mid BAD].\end{aligned}$$

Consequently,

$$\min(\Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x \mid BAD], \Pr_{x\sim\mathcal{N}}[\mathcal{A} \text{ rejects } x \mid BAD]) \leq 1/2.$$

Assume w.l.o.g. that $\Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x \mid BAD] \leq 1/2$. Then,

$$\begin{aligned}&\Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x] \\ &= \Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x \mid \overline{BAD}] \cdot \Pr[\overline{BAD}] \\ &\quad + \Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x \mid BAD] \cdot \Pr[BAD] \\ &< 1 \cdot \frac{1}{10} + \Pr_{x\sim\mathcal{P}}[\mathcal{A} \text{ accepts } x \mid BAD] \cdot 1 \\ &\leq \frac{1}{10} + \frac{1}{2} < \frac{2}{3}.\end{aligned}$$

Thus, every uniform algorithm requires $\Omega(\epsilon^{-5/4})$ samples to test convexity with error probability at most 1/3. □ □

5 Conclusion and Open Problems

We showed that uniform testers are as powerful as adaptive testers in the case of the half-plane property. Specifically, our uniform half-plane tester has 1-sided error and optimal running time. For convexity, the best previously known tester was uniform. However, we designed an adaptive tester with better (optimal) query complexity and showed that every uniform tester must have a significantly larger query complexity than our adaptive tester.

One remaining open problem is to resolve the sample complexity of an optimal (2-sided error) uniform tester for convexity. Our lower bound on this quantity is $\Omega(\epsilon^{-5/4})$, while the best upper bound is $O(\epsilon^{-4/3})$ [8]. The $O(\epsilon^{-4/3})$-sample uniform tester in [8] is the natural one: after taking the uniform samples, it checks if a white point was caught in the convex hull of sampled black points. However, the analysis is quite sophisticated. One of the ideas in the analysis is the following. The smallest witness of nonconvexity we can hope to get with $O(\epsilon^{-4/3})$ uniform samples has 4 points: 3 black points that form a triangle and a white point contained in that triangle. (We are very unlikely to sample 3 points on the same line, so we can't hope to reliably obtain a smaller witness of two black samples and a white sample on the line segment formed by them.) To save on the size of the witness, one of the black points in the witness is replaced with a black-central point. Even though we are not likely to sample such a point, we are likely to have a "proof" that such a point is black because it is likely to be in the convex hull of black samples. So, the analysis focuses on obtaining the other two black vertices of the triangle in the witness and the white sample contained in it. Since, for uniform 1-sided error testers, there is a matching lower bound of $\Omega(\epsilon^{-4/3})$ in [8], the only hope to beat the $O(\epsilon^{-4/3})$-sample upper bound for uniform testers is to allow for 2-sided error. That is, one would have to come up with a criterion to reject a figure that looks "statistically suspicious", even in the case when the tester found no witness of nonconvexity.

Another direction for research is to investigate the power of uniform samples in the context of tolerant property testing. Tolerant testing of 2-dimensional figures was investigated in [9]. The tolerant testers for half-plane and convexity in that work are uniform and have nearly optimal query complexity (as compared to any, even adaptive testers). However, it is open whether uniform samples are sufficient for achieving the optimal running time for tolerantly testing these properties. It is interesting to investigate the power of other restricted classes of testers, such as nonadaptive testers, in the context of testing of properties of geometric figures. Finally, this work only looks at 2-dimensional figures. Generalizing this study to higher dimensions is an intriguing open question.

References

[1] Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: It's all about regularity. *SIAM J. Comput.*, 39(1):143–167, 2009.

[2] A. M. Andrew. Another efficient algorithm for convex hulls in two dimensions. *Inf. Process. Lett.*, 9(5):216–219, 1979.

[3] Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. Optimal unateness testers for real-valued functions: Adaptivity helps. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 5:1–5:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[4] Tugkan Batu, Sanjoy Dasgupta, Ravi Kumar, and Ronitt Rubinfeld. The complexity of approximating the entropy. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity, Montréal, Québec, Canada, May 21-24, 2002*, page 17. IEEE Computer Society, 2002.

[5] Tugkan Batu, Funda Ergün, Joe Kilian, Avner Magen, Sofya Raskhodnikova, Ronitt Rubinfeld, and Rahul Sami. A sublinear algorithm for weakly approximating edit distance. In Lawrence L. Larmore and Michel X. Goemans, editors, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 316–324. ACM, 2003.

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

[7] Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF properties are hard to test. *SIAM J. Comput.*, 35(1):1–21, 2005.

[8] Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Testing convexity of figures under the uniform distribution. In *32nd International Symposium on Computational Geometry, SoCG 2016, June 14-18, 2016, Boston, MA, USA*, pages 17:1–17:15, 2016.

[9] Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant testers of image properties. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 90:1–90:14, 2016.

[10] Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. The power and limitations of uniform samples in testing properties of figures. *Algorithmica*, Jul 2018.

[11] Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 164–173, 2014.

[12] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comput. Syst. Sci.*, 47(3):549–595, 1993.

[13] Clément Canonne. A survey on distribution testing: Your data is big. But is it blue? *Electronic Colloquium on Computational Complexity (ECCC)*, 22:63, 2015.

[14] Artur Czumaj and Christian Sohler. Property testing with geometric queries. In *Algorithms - ESA 2001, 9th Annual European Symposium, Aarhus, Denmark, August 28-31, 2001, Proceedings*, pages 266–277, 2001.

[15] Artur Czumaj, Christian Sohler, and Martin Ziegler. Property testing in computational geometry. In *Algorithms - ESA 2000, 8th Annual European Symposium, Saarbrücken, Germany, September 5-8, 2000, Proceedings*, pages 155–166, 2000.

[16] Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Randomization, Approximation, and Combinatorial Algorithms and Techniques, Third International Workshop on Randomization and Approximation Techniques in Computer Science, RANDOM'99, Berkeley, CA, USA*, pages 97–108, 1999.

[17] Herbert Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1987.

[18] Funda Ergün, Sampath Kannan, Ravi Kumar, Ronitt Rubinfeld, and Mahesh Viswanathan. Spotcheckers. *J. Comput. Syst. Sci.*, 60(3):717–751, 2000.

[19] Eldar Fischer, Oded Lachish, and Yadu Vasudev. Trading query complexity for sample-based testing and multi-testing scalability. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1163–1182. IEEE Computer Society, 2015.

[20] Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 474–483. ACM, 2002.

[21] O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002.

[22] Oded Goldreich. Combinatorial property testing (a survey). *Randomization Methods in Algorithm Design*, 43:45–59, 1999.

[23] Oded Goldreich. *Property testing: current research and surveys*, volume 6390. Springer, 2010.

[24] Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.

[25] Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

[26] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

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

[28] Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of lipschitz functions with applications to data privacy. *SIAM J. Comput.*, 42(2):700–731, 2013.

[29] Igor Kleiner, Daniel Keren, Ilan Newman, and Oren Ben-Zwi. Applying property testing to an image partitioning problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(2):256–265, 2011.

[30] Simon Korman, Daniel Reichman, and Gilad Tsur. Tight approximation of image matching. *CoRR*, abs/1111.1713, 2011.

[31] 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, 2013.

[32] Oded Lachish and Ilan Newman. Testing periodicity. *Algorithmica*, 60(2):401–420, 2011.

[33] Nimrod Megiddo. Partitioning with two lines in the plane. *J. Algorithms*, 6(3):430–433, 1985.

[34] Ilan Newman, Yuri Rabinovich, Deepak Rajendraprasad, and Christian Sohler. Testing for forbidden order patterns in an array. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1582–1597. SIAM, 2017.

[35] Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin M. Varma. Parameterized property testing of functions. *TOCT*, 9(4):17:1–17:19, 2018.

[36] Luis Rademacher and Santosh Vempala. Testing geometric convexity. In *FSTTCS*, pages 469–480, 2004.

[37] Sofya Raskhodnikova. Approximate testing of visual properties. In *RANDOM-APPROX*, pages 370–381, 2003.

[38] Sofya Raskhodnikova. Testing if an array is sorted. In *Encyclopedia of Algorithms*, pages 2219–2222. 2016.

[39] Sofya Raskhodnikova, Dana Ron, Amir Shpilka, and Adam Smith. Strong lower bounds for approximating distribution support size and the distinct elements problem. *SIAM J. Comput.*, 39(3):813–842, 2009.

[40] Sofya Raskhodnikova and Ronitt Rubinfeld. Linearity testing/testing hadamard codes. In *Encyclopedia of Algorithms*, pages 1107–1110. 2016.

[41] Sofya Raskhodnikova and Adam D. Smith. A note on adaptivity in testing properties of bounded degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(089), 2006.

[42] Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends® in Theoretical Computer Science*, 5(2):73–205, 2010.

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

[44] Ronitt Rubinfeld. Taming big probability distributions. *ACM Crossroads*, 19(1):24–28, 2012.

[45] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

[46] Bernd Schmeltz. Learning convex sets under uniform distribution. In *Data Structures and Efficient Algorithms, Final Report on the DFG Special Joint Initiative*, pages 204–213, 1992.

[47] Wojciech Szpankowski. *Average Case Analysis of Algorithms on Sequences.* John Wiley & Sons, Inc., New York, 2001.

[48] John W Tukey. Mathematics and the picturing of data. In *Proceedings of the international congress of mathematicians*, volume 2, pages 523–531, 1975.

[49] Paul Valiant. Testing symmetric properties of distributions. *SIAM J. Comput.*, 40(6):1927–1968, 2011.