Tolerant Testers of Image Properties[*][†]

Piotr Berman[1], Meiram Murzabulatov[2], and Sofya Raskhodnikova[3]

1 **Pennsylvania State University, University Park, USA**
`berman@cse.psu.edu`

2 **Pennsylvania State University, University Park, USA**
`mzm269@psu.edu`

3 **Pennsylvania State University, University Park, USA**
`sofya@cse.psu.edu`

Abstract

We initiate a systematic study of tolerant testers of image properties or, equivalently, algorithms that approximate the distance from a given image to the desired property (that is, the smallest fraction of pixels that need to change in the image to ensure that the image satisfies the desired property). Image processing is a particularly compelling area of applications for sublinear-time algorithms and, specifically, property testing. However, for testing algorithms to reach their full potential in image processing, they have to be tolerant, which allows them to be resilient to noise. Prior to this work, only one tolerant testing algorithm for an image property (image partitioning) has been published.

We design efficient approximation algorithms for the following fundamental questions: What fraction of pixels have to be changed in an image so that it becomes a half-plane? a representation of a convex object? a representation of a connected object? More precisely, our algorithms approximate the distance to three basic properties (being a half-plane, convexity, and connectedness) within a small additive error ϵ, after reading a number of pixels polynomial in $1/\epsilon$ and independent of the size of the image. The running time of the testers for half-plane and convexity is also polynomial in $1/\epsilon$. Tolerant testers for these three properties were not investigated previously. For convexity and connectedness, even the existence of distance approximation algorithms with query complexity independent of the input size is not implied by previous work. (It does not follow from the VC-dimension bounds, since VC dimension of convexity and connectedness, even in two dimensions, depends on the input size. It also does not follow from the existence of non-tolerant testers.)

Our algorithms require very simple access to the input: uniform random samples for the half-plane property and convexity, and samples from uniformly random blocks for connectedness. However, the analysis of the algorithms, especially for convexity, requires many geometric and combinatorial insights. For example, in the analysis of the algorithm for convexity, we define a set of reference polygons P_ϵ such that (1) every convex image has a nearby polygon in P_ϵ and (2) one can use dynamic programming to quickly compute the smallest empirical distance to a polygon in P_ϵ. This construction might be of independent interest.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Computational geometry, convexity, half-plane, connectedness, property testing, tolerant property testing

Digital Object Identifier 10.4230/LIPIcs.ICALP.2016.90

[*] A full version of the paper is available at `https://arxiv.org/abs/1503.01363`.

[†] M.M. and S.R. were supported by NSF CAREER award CCF-0845701 and NSF award CCF-1422975. S.R. was also supported by Boston University's Hariri Institute for Computing and Center for Reliable Information Systems and Cyber Security and, while visiting the Harvard Center for Research on Computation & Society, by a Simons Investigator grant to Salil Vadhan.

1 Introduction

Image processing is a particularly compelling area of applications for sublinear-time algorithms and, specifically, property testing. Images are huge objects, and our visual system manages to process them very quickly without examining every part of the image. Moreover, many applications in image analysis have to process a large number of images online, looking for an image that satisfies a certain property among images that are generally very far from satisfying it. Or, alternatively, they look for a subimage satisfying a certain property in a large image (e.g., a face in an image where most regions are part of the background.) There is a growing number of proposed *rejection-based* algorithms that employ a quick test that is likely to reject a large number of unsuitable images (see, e.g., citations in [15]).

Property testing [21, 10] is a formal study of fast algorithms that accept objects with a given property and reject objects that are far. Testing image properties in this framework was first considered in [19]. Ron and Tsur [20] initiated property testing of images with a different input representation, suitable for testing properties of sparse images. Since these models were proposed, several sublinear-time algorithms for visual properties were implemented and used: namely, those by Kleiner et al. and Korman et al. [15, 16, 17].

However, for sublinear-time algorithms to reach their full potential in image processing, they have to be resilient to noise: images are often noisy, and it is undesirable to reject images that differ only on a small fraction of pixels from an image satisfying the desired property. Tolerant testing was introduced by Parnas, Ron and Rubinfeld [18] exactly with this goal in mind—to deal with noisy objects. It builds on the property testing model and calls for algorithms that accept objects that are close to having a desired property and reject objects that are far. Another related task is approximating distance of a given object to a nearest object with the property within additive error ϵ. (Distance approximation algorithms imply tolerant testers in a straightforward way.) The only image problem for which tolerant testers were studied is the image partitioning problem investigated by Kleiner et al. [15].

Our results. We design efficient approximation algorithms for the following fundamental questions: What fraction of pixels have to be changed in an image so that it becomes a half-plane? a representation of a convex object? a representation of a connected object? In other words, we design algorithms that approximate the distance to being a half-plane, convexity and connectedness within a small additive error or, equivalently, tolerant testers for these properties. These problems were not investigated previously in the tolerant testing framework. For all three properties, we give ϵ-additive distance approximation algorithms that run in constant time (i.e., dependent only on ϵ, but not the image size). We remark that even though it was known that these properties can be tested in constant time [19], this fact does not necessarily imply constant-query tolerant testers for these properties. E.g., Fischer and Fortnow [9] exhibit a property (of objects representable with strings of length n) which is testable with a constant number of queries, but for which every tolerant tester requires $n^{\Omega(1)}$ queries. For convexity and connectedness, even the existence of distance approximation algorithms with query (or time) complexity independent of the input size does not follow from previous work. It does not follow from the VC-dimension bounds, since VC dimension of convexity and connectedness, even in two dimensions, depends on the input size[1]. Implications of the VC dimension bound on convexity are further discussed below.

[1] For $n \times n$ images, the VC dimension of convexity is $\Theta(n^{2/3})$ (this is the maximum number of vertices of a convex lattice polygon in an $n \times n$ lattice [1]); for connectedness, it is $\Theta(n)$.

Table 1 Our results on distance approximation. To get complexity of (ϵ_1, ϵ_2)-tolerant testing, substitute $\epsilon = (\epsilon_2 - \epsilon_1)/2$.

Property	Sample Complexity	Run Time	Access to Input
Half-plane	$O\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon}\right)$	$O\left(\frac{1}{\epsilon^3}\log\frac{1}{\epsilon}\right)$	uniformly random pixels
Convexity	$O\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon}\right)$	$O\left(\frac{1}{\epsilon^8}\right)$	uniformly random pixels
Connectedness	$O\left(\frac{1}{\epsilon^4}\right)$	$\exp\left(O\left(\frac{1}{\epsilon}\right)\right)$	uniformly random blocks of pixels

Our results on distance approximation are summarized in Table 1. Our algorithm for convexity is the most important and technically difficult of our results, requiring a large number of new ideas to get running time polynomial in $1/\epsilon$. To achieve this, we define a set of reference polygons P_ϵ such that (1) every convex image has a nearby polygon in P_ϵ and (2) one can use dynamic programming to quickly compute the smallest empirical distance to a polygon in P_ϵ. It turns out that the empirical error of our algorithm is proportional to the sum of the square roots of the areas of the regions it considers in the dynamic program. To guarantee (2) and keep our empirical error small, our construction ensures that the sum of the square roots of the areas of the considered regions is small. This construction might be of independent interest.

Our algorithms do not need sophisticated access to the input image: uniformly randomly sampled pixels suffice for our algorithms for the half-plane property and convexity. For connectedness, we allow our algorithms to query pixels from a uniformly random block. (See the end of Section 2 for a formal specification of the input access.)

Our algorithms for convexity and half-plane work by first implicitly learning the object[2]. PAC learning was defined by Valiant [23], and agnostic learning, by Kearns et al. [14] and Haussler [12]. As a corollary of our analysis, we obtain fast proper agnostic PAC learners of half-planes and of convex sets in two dimensions that work under the uniform distribution. The sample and time complexity[3] of the PAC learners is as indicated in Table 1 for distance approximation algorithms for corresponding properties.

While the sample complexity of our agnostic half-plane learner (and hence our distance approximation algorithm for half-planes) follows from the VC dimension bounds, its running time does not. Agnostically learning half-spaces under the uniform distribution has been studied by [13], but only for the hypercube $\{-1, 1\}^d$ domains, not the plane. Our PAC learner of convex sets, in contrast to our half-plane learner, dimension lower bounds on sample complexity. (The sample complexity of a PAC learner for a class is at least proportional to the VC dimension of that class [8].) Since VC dimension of convexity of $n \times n$ images is $\Theta(n^{2/3})$, proper PAC learners of convex sets in two dimensions (that work under arbitrary

[2] There is a known implication from learning to testing. As proved in [10], a proper PAC learning algorithm for property $\mathcal{P}$ with sampling complexity $q(\epsilon)$ implies a 2-sided error (uniform) property tester for $\mathcal{P}$ that takes $q(\epsilon/2) + O(1/\epsilon)$ samples. There is an analogous implication from proper agnostic PAC learning to distance approximation with an overhead of $O(1/\epsilon^2)$ instead of $O(1/\epsilon)$. We choose to present our testers first and get learners as corollary because our focus is on testing and because we want additional features for our testers, such as 1-sided error, that do not automatically follow from the generic relationship.

[3] All our results are stated for error probability $\delta = 1/3$. To get results for general δ, by standard arguments, it is enough to multiply the complexity of an algorithm by $\log 1/\delta$.

distributions) must have sample complexity $\Omega(n^{2/3})$. However, one can do much better with respect to the uniform distribution. Schmeltz [22] showed that a non-agnostic learner for that task needs $\Theta(\epsilon^{-3/2})$ samples. Surprisingly, it appears that this question has not been studied at all for agnostic learners. Our agnostic learner for convex sets in 2D under the uniform distribution needs $O\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon}\right)$ samples and runs in time $O\left(\frac{1}{\epsilon^8}\right)$.

Finally, we note that for connectedness, we take a different approach. Our algorithms do not try to learn the object first; instead they rely on a combinatorial characterization of distance to connectedness. We show that distance to connectedness can be represented as an average of distances of sub-images to a related property.

Comparison to other related work. Property testing has rich literature on graphs and functions, however, properties of images have been investigated very little. Even though superficially the inputs to various types of testing tasks might look similar, the problems that arise are different. In the line of work on testing dense graphs, started by Goldreich et al. [10], the input is also an $n \times n$ binary matrix, but it represents an adjacency matrix of the dense input graph. So, the problems considered are different than in this work. In the line of work on testing geometric properties, started by Czumaj, Sohler, and Ziegler [7] and Czumaj and Sohler [6], the input is a set of points represented by their coordinates. The allowed queries and the distance measure on the input space are different from ours.

A line of work potentially relevant for understanding connectedness of images is on connectedness of bounded-degree graphs. Goldreich and Ron [11] gave a tester for this property, subsequently improved by Berman et al. [3]. Campagna et al. [5] gave a tolerant tester for this problem. Even though we view our image as a graph in order to define connectedness of images, there is a significant difference in how distances between instances are measured (see [19] for details). We also note, that unlike in [5], our tolerant tester for connectedness is fully tolerant, i.e., it works for all settings of parameters.

The only previously known tolerant tester for image properties was given by Kleiner et al. [15]. They consider the following class of image partitioning problems, each specified by a $k \times k$ binary template matrix T for a small *constant* k. The image satisfies the property corresponding to T if it can be partitioned by $k-1$ horizontal and $k-1$ vertical lines into blocks, where each block has the same color as the corresponding entry of T. Kleiner et al. prove that $O(1/\epsilon^2)$ samples suffice for tolerant testing of image partitioning properties. Note that VC dimension of such a property is $O(1)$, so by Footnote 2, we can get a $O(1/\epsilon^2 \log 1/\epsilon)$ bound. Our algorithms required numerous new ideas to significantly beat VC dimension bounds (for convexity and connectedness) and to get low running time.

For the properties we study, distance approximation algorithms and tolerant testers were not investigated previously. In the standard property testing model, the half-plane property can be tested in $O(\epsilon^{-1})$ time [19], convexity can be tested in $O(\epsilon^{-4/3})$ time [2], and connectedness can be tested in $O(\epsilon^{-2}\log\epsilon^{-1})$ time [19, 3]. As we explained, property testers with running time independent of ϵ do not necessarily imply tolerant testers with that feature. Many new ideas are needed to obtain our tolerant testers. In particular, the standard testers for half-plane and connectedness are adaptive while the testers here need only random samples from the image, so the techniques used for analyzing them are different. The tester for convexity in [2] uses only random samples, but it is not based on dynamic programming.

Open questions. In this paper we give tolerant testers for several important problems on images. It is open whether these testers are optimal. No nontrivial lower bounds are known for these problems. (For any non-trivial property, an easy lower bound on the query complexity of a distance approximation algorithm is $\Omega(1/\epsilon^2)$. This follows from the fact that $\Omega(1/\epsilon^2)$ coin flips are needed to distinguish between a fair coin and a coin that lands heads with probability $1/2+\epsilon$.) Thus, our testers for half-plane and convexity are nearly optimal in terms of query complexity (up to a logorithmic factor in $1/\epsilon$). But it is open whether their running time can be improved.

Organization. We give formal definitions and notation in Section 2, deferring some standard definitions to the full version of this article. Algorithms for being a half-plane, convexity, and connectedness are given in Sections 3, 4, and 5, respectively. We view our half-plane result as a good preparation for our distance approximation algorithm for convexity, the most technically difficult result in the paper. Corollaries about PAC learners as well as all omitted proofs and numerous figures can be found in the full version of this article.

2 Definitions and Notation

We use $[0..n)$ to denote the set of integers $\{0,1,\dots,n-1\}$ and $[n]$ to denote $\{1,2,\dots,n\}$.

Image representation. We focus on black and white images. For simplicity, we only consider square images, but everything in this paper can be easily generalized to rectangular images. We represent an image by an $n \times n$ binary matrix M of pixel values, where 0 denotes white and 1 denotes black. We index the matrix by $[0..n)^2$. The object is a subset of $[0..n)^2$ corresponding to black pixels; namely, $\{(i,j) \mid M[i,j]=1\}$.

The *absolute distance*, $Dist(M_1,M_2)$, between matrices M_1 and M_2 is the number of the entries on which they differ. The *relative distance* between them is $dist(M_1,M_2) = Dist(M_1,M_2)/n^2$. A property $\mathcal{P}$ is a subset of binary matrices.

Access to the input. A *query-based* algorithm accesses its $n \times n$ input matrix M by specifying a query pixel (i,j) and obtaining $M[i,j]$. A *uniform* algorithm accesses its $n \times n$ input matrix by drawing independent samples (i,j) from the uniform distribution over the domain (i.e., $[0..n)^2$) and obtaining $M[i,j]$. A *block-uniform algorithm* accesses its $n \times n$ input matrix by specifying a block length $r \in [n]$. For a block length r of its choice, the algorithm draws $x, y \in [\lceil n/r \rceil]$ uniformly at random and obtains set $\{(i,j) \mid \lfloor i/r \rfloor = x \text{ and } \lfloor j/r \rfloor = y\}$ and $M[i,j]$ for all (i,j) in this set. The sample complexity of a *uniform* or a *block-uniform* algorithm is the number of pixels of the image it examines.

▶ Remark 2.1. Uniform algorithms have access to independent (labeled) samples from the uniform distribution over the domain. *Bernoulli algorithms* only have access to (labeled) Bernoulli samples from the image: namely, each pixel appears in the sample with probability s/n^2, where s is the sample parameter that controls the expected sample complexity. By standard arguments, a Bernoulli algorithm with the sample parameter s can be used to obtain a uniform algorithm that takes $O(s)$ samples and has the same guarantees as the original algorithm (and vice versa).

3 Distance Approximation to the Nearest Half-Plane

An image is a *half-plane* if there exist an angle $\varphi \in [0, 2\pi)$ and a real number c such that $M[x,y] = 1$ (i.e., pixel (x,y) is black) iff $x\cos\varphi + y\sin\varphi \geq c$. In other words, an image is a half-plane if there is a line, called a *separating line*, that separates black and white pixels of the image. For all φ and c, let M_c^φ denote the half-plane that satisfies the above inequality with parameters φ and c, and let L_c^φ be the segment of the separating line that belongs to the image. We call φ the *direction* of M_c^φ (and L_c^φ). Note that φ is the oriented angle between the x-axis and a line perpendicular to L_c^φ.

▶ **Theorem 3.1.** *There is a uniform ϵ-additive distance approximation algorithm for the half-plane property with sample complexity $O(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon})$ and time complexity $O(\frac{1}{\epsilon^3}\log\frac{1}{\epsilon})$.*

Proof. At a high level, our algorithm (Algorithm 1) constructs a small set H_ϵ of reference half-planes. It samples pixels uniformly at random and outputs the empirical distance to the closest reference half-plane. The core property of H_ϵ is that the smallest empirical distance to a half-plane in H_ϵ can be computed quickly.

▶ **Definition 3.2** (Reference directions and half-planes)**.** Given $\epsilon \in (0, \frac{1}{4})$, let $a = \epsilon n/\sqrt{2}$. Let D_ϵ be the set of directions of the form $i\epsilon$ for $i \in [0..\lceil 2\pi/\epsilon \rceil)$, called *reference directions*. The set of *reference half-planes*, denoted H_ϵ, consists of half-planes of the form M_c^φ, where $\varphi \in D_\epsilon$, the reference line intersects $[0, n-1]^2$, and c is an integer multiple of a.

In other words, for every reference direction, we space reference half-planes distance a apart. By definition, there are at most $\sqrt{2}n/a = 2/\epsilon$ reference half-planes for each direction in D_ϵ and, consequently, $|H_\epsilon| \leq 2\pi/\epsilon \cdot (2/\epsilon) < 13/\epsilon^2$.

Algorithm 1: Distance approximation to being a half-plane.

```
Input : parameters $n \in \mathbb{N}$, $\epsilon \in (0, 1/4)$; Bernoulli access to an $n \times n$ binary matrix $M$.
1 Sample a set $S$ of $s = \frac{4}{\epsilon^2}\ln\frac{9}{\epsilon}$ pixels uniformly at random with replacement.
2 Let $D_\epsilon$ and $H_\epsilon$ be the sets of reference directions and half-planes, respectively (see
    Definition 3.2) and let $a = \epsilon n/\sqrt{2}$.
  // Compute $\hat{d} = \min_{M' \in H_\epsilon} \hat{d}(M')$, where $\hat{d}(M') = \frac{1}{s} \cdot |\{p \in S : M[p] \neq M'[p]\}|$:
3 foreach $\varphi \in D_\epsilon$ do
    // Lines with direction $\varphi$ partition the image. Bucket sort samples by
       position in the partition:
4   Assign each sample $(x, y) \in S$ to bucket $j = \lfloor (x\cos\varphi + y\sin\varphi)/a \rfloor$.
5   For each bucket $j$, compute $w_j$ and $b_j$, the number of white and black pixels it has.
6   For each $j$, where $M_{ja}^\varphi \in H_\epsilon$, compute $\hat{d}(M_{ja}^\varphi) = \frac{1}{s}\sum_{k<j} b_k + \frac{1}{s}\sum_{k\geq j} w_k$.
7 Output $\hat{d}$, the minimum of the values computed in Step 6.
```

▶ **Lemma 3.3.** *For every half-plane matrix M, there is $M' \in H_\epsilon$ such that $dist(M, M') \leq \epsilon/2$.*

Proof. Consider a half-plane M_c^φ. Let φ' be a reference direction closest to φ. Then $|\varphi - \varphi'| \leq \epsilon/2$. We consider two cases. See Figures 1 and 2.

Case 1: Suppose that there is a reference half-plane $M_{c'}^{\varphi'}$ such that the separating line segments L_c^φ and $L_{c'}^{\varphi'}$ intersect. Note that the length of every line segment that belongs to the image is at most $\sqrt{2}n$. The symmetric difference of M_c^φ and $M_{c'}^{\varphi'}$ is contained in two regions formed by line segments L_c^φ and $L_{c'}^{\varphi'}$. Each of these regions is either a triangle or (if it contains a corner of the image) a quadrilateral. First, suppose both regions are triangles. The sum of lengths of their bases, that lie on the same line, is at most $\sqrt{2}n$, whereas the

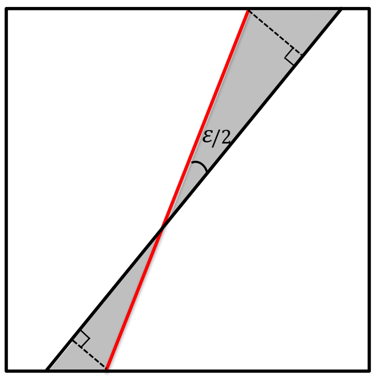

Figure 1 Proof of Lemma 3.3: triangular regions.

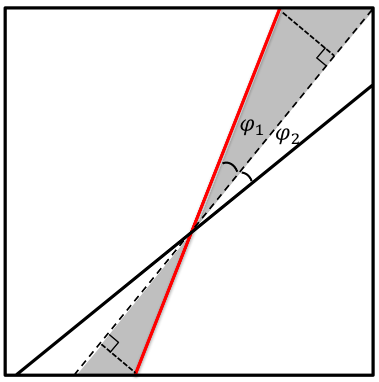

Figure 2 Proof of Lemma 3.3: triangular and quadrilateral regions.

sum of their heights is at most $\sin(\epsilon/2) \times \sqrt{2}n \le \epsilon n/\sqrt{2}$. Hence, the sum of their areas[4] is at most $\epsilon n^2/2$.

If exactly one of the regions is a quadrilateral, we add a line through the corner of the image contained in the quadrilateral and the intersection point of L_c^{φ} and $L_{c'}^{\varphi'}$. It partitions the symmetric difference of M_c^{φ} and $M_{c'}^{\varphi'}$ into two pairs of triangular regions. Let φ_1 (respectively, φ_2) be the angle between the new line and L_c^{φ} (respectively, $L_{c'}^{\varphi'}$). Then $\varphi_1 + \varphi_2 \le \epsilon/2$. Applying the same reasoning as before to each pair of regions, we get that the sum of their areas is at most $\varphi_1 n^2 + \varphi_2 n^2 \le \epsilon n^2/2$. If both regions are quadrilaterals, we add a line as before for each of them and apply the same reasoning as before to the three resulting pairs of regions. Again, the area of the symmetric difference of M_c^{φ} and $M_{c'}^{\varphi'}$ is at most $\epsilon n^2/2$. Thus[4], $M_{c'}^{\varphi'}$ is the required M'.

Case 2: There exist reference half-planes with separating line segments $L = L_{c'}^{\varphi'}$ and $L' = L_{c'+a}^{\varphi'}$ such that the line segment L_c^{φ} is between L and L'. The region between L and L' has length at most $\sqrt{2}n$ and width a. Thus, its area is at most ϵn^2. Partition it into two regions: between L and L_c^{φ} and between L' and L_c^{φ}. One of the two regions has area at most $\epsilon n^2/2$. Thus, $M_{c'}^{\varphi'}$ or $M_{c'+a}^{\varphi'}$ is the required M'. ◀

Analysis of Algorithm 1. Let d_M be the distance of M to being a half-plane. Then there exists a half-plane matrix M^* such that $dist(M, M^*) = d_M$. By a uniform convergence bound (see, e.g., [4]), since $s \ge (2/\epsilon^2)(\ln |H_\epsilon| + \ln 6)$ for all $\epsilon \in (0, 1/4)$, we get that with probability at least $2/3$, $|dist(M, M') - \hat{d}(M')| \le \epsilon/2$ for all $M' \in H_\epsilon$. Suppose this event happened. Then $\hat{d} \ge d_M - \epsilon/2$ because $dist(M, M') \ge d_M$ for all half-planes M'. Moreover, by Lemma 3.3, there is a matrix $\hat{M} \in H_\epsilon$ such that $dist(M, \hat{M}) \le dist(M, M^*) + dist(M^*, \hat{M}) \le d_M + \epsilon/2$. For this matrix, $\hat{d}(\hat{M}) \le dist(M, \hat{M}) + \epsilon/2 \le d_M + \epsilon$. Thus, $d_M - \epsilon/2 \le \hat{d} \le d_M + \epsilon$. That is, $|d_M - \hat{d}| \le \epsilon$ with probability $2/3$, as required.

Sample and time complexity. The number of samples, s, is $O(1/\epsilon^2 \log 1/\epsilon)$. To analyze the running time, recall that $|D_\epsilon| = O(1/\epsilon)$. For each direction in D_ϵ, we bucket sort

[4] For simplicity of presentation, we equate the area of a convex region and the number of pixels in it, thus ignoring additional small-order terms. By Pick's theorem, the number of pixels could be at most the area plus $2n$. It does not affect the asymptotic analysis of our algorithms.

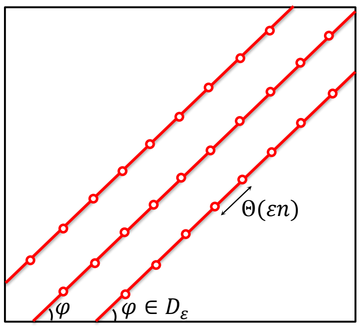

Figure 3 An illustration of reference lines and reference points.

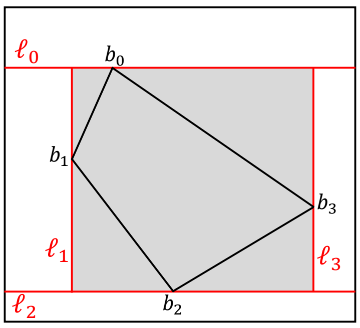

Figure 4 An illustration of a reference box and triangles of $\mathbf{T}_0$.

all samples in expected $O(s)$ time. The remaining steps in the **foreach** loop of Step 3 can be implemented to run in $O(s)$ time. The expected run time of Algorithm 1 is thus $O(1/\epsilon \cdot s) = O(1/\epsilon^3 \log 1/\epsilon)$. Remark 2.1 implies a tester with the same worst case run time. ◀

4 Distance Approximation to the Nearest Convex Image

An image is *convex* if the convex hull of all black pixels contains only black pixels.

▶ **Theorem 4.1.** *There is a uniform ϵ-additive distance approximation algorithm for convexity with sample complexity $O(\frac{1}{\epsilon^2} \log \frac{1}{\epsilon})$ and running time $O(\frac{1}{\epsilon^8})$.*

Proof. The starting point for our algorithm for approximating the distance to convexity (Algorithm 2) is similar to that of Algorithm 1 that approximates the distance to a nearest half-plane. We define a small set P_ϵ of reference polygons. Algorithm 2 implicitly learns a nearby reference polygon and outputs the empirical distance from the image to that polygon. The key features of P_ϵ is that (1) every convex image has a nearby polygon in P_ϵ, and (2) one can use dynamic programming (DP) to quickly compute the smallest empirical distance to a polygon in P_ϵ.

We start by defining reference directions, lines, points, and line-point pairs that are later used to specify our DP instances. Reference directions are almost the same as in Definition 3.2.

▶ **Definition 4.2** (Reference lines, line-point pairs). Fix $\epsilon_0 = \epsilon/144$. The set of *reference directions* is $D_\epsilon = \{\pi/2\} \cup \{i\epsilon_0 : i \in [0, \lceil 2\pi/\epsilon_0 \rceil)\}$. For every $\varphi \in D_\epsilon$, define the set of *reference lines* $L_\varphi = \{\ell : \ell$ passes through the image and satisfies the equation $x \cos\varphi + y \sin\varphi = c$, where c is an integer multiple of $\epsilon_0 n\}$. For each reference line, the set of *reference points on ℓ* contains points w.r.t. ℓ, which are inside $[0, n-1]^2$, spaced exactly $\epsilon_0 n$ apart (it does not matter how the initial point is picked). A *line-point pair* is a pair (ℓ, b), where ℓ is a reference line and b is a reference point w.r.t. ℓ. (Note that there could be reference points on ℓ that were defined w.r.t. some other reference line. This is why we say "a reference point w.r.t. ℓ", and not "a reference point on ℓ".)

Roughly speaking, a reference polygon is a polygon whose vertices are defined by line-point pairs. There are additional restrictions that stem from the fact that we need to be able to efficiently find a nearby reference polygon for an input image. The actual definition specifies which actions we can take while constructing a reference polygon. Reference polygons are built starting from reference boxes, which are defined next.

▶ **Definition 4.3** (Reference box). A *reference box* is a set of four line-point pairs (ℓ_i, b_i) for $i = 0, 1, 2, 3$, where ℓ_0, ℓ_2 are distinct horizontal lines, such that ℓ_0 is above ℓ_2, and (ℓ_1, ℓ_3) are distinct vertical lines, such that ℓ_1 is to the left of ℓ_3. The reference box defines a vertex set $B_0 = \{b_0, b_1, b_2, b_3\}$ and a triangle set $\mathbf{T}_0$, formed by removing the quadrilateral $b_0b_1b_2b_3$ from the rectangle delineated by the lines $\ell_0, \ell_1, \ell_2, \ell_3$.

Intuitively, by picking a reference box, we decide to keep the area inside the quadrilateral $b_0b_1b_2b_3$ black, the area outside the rectangle formed by $\ell_0, \ell_1, \ell_2, \ell_3$ white, and the triangles in $\mathbf{T}_0$ gray, i.e., undecided for now.

▶ **Definition 4.4.** For points x, y, let $\ell(x, y)$ denote the line that passes through x and y. Let xy denote the line segment between x and y.

Reference polygons are defined next. Intuitively, to obtain a reference polygon, we keep subdividing "gray" triangles in $\mathbf{T}_0$ into smaller triangles and deciding to color the smaller triangles black or white or keep them gray (i.e., undecided for now). We also allow "cutting off" a quadrilateral that is adjacent to black and coloring it black (a.k.a. "the base change operation"). Even though the definition of reference polygons is somewhat technical, the readers can check their understanding of this concept by following Algorithm 2, as it chooses the best reference polygon to approximate the input image.

▶ **Definition 4.5** (Reference polygon). A *reference polygon* is an image of a polygon Hull(B), where the set B can be obtained from a reference box with a vertex set B_0 and a triangle set $\mathbf{T}_0$ by the following recursive process. Initially, $\mathbf{T}_{\text{end}} = \emptyset$ and $B = B_0$. While $\mathbf{T}_0 \neq \emptyset$, move a triangle T from $\mathbf{T}_0$ to $\mathbf{T}_{\text{end}}$ and perform the following steps:

1. (Base Change). Let $T = \triangle b'b''v$, where $b', b'' \in B$. Select reference point b'_0 on $b'v$ w.r.t. line $\ell(b', v)$, and reference point b''_0 on $b''v$ w.r.t. line $\ell(b'', v)$. Add b'_0, b''_0 to B. (This corresponds to coloring the quadrilateral $b'b'_0b''_0b''$ black.) Let h be the height of $\triangle b'_0b''_0v$ w.r.t. the base $b'_0b''_0$.
2. (Subdivision Step) If $h > 6\epsilon_0 n$, choose whether to proceed with this step or go to Step 3 (both choices correspond to a legal reference polygon); otherwise, go to Step 3. Let φ be the angle between $\ell(b'_0, b''_0)$ and the x-axis, and $\hat{\varphi} \in D_\epsilon$ be such that $|\hat{\varphi} - \varphi| \leq \epsilon_0/2$. Select a reference line-point pair (ℓ, b), where the line $\ell \in L_{\hat{\varphi}}$ crosses b'_0v and b''_0v, whereas b is in the triangle $\triangle b'_0b''_0v$. Let v' (resp., v'') be the point of intersection of ℓ and b'_0v (resp., ℓ and b''_0v). Let $T' = \triangle b'_0bv'$, $T'' = \triangle b''_0bv''$. Add b to B and triangles T', T'' to $\mathbf{T}_0$. (This represents coloring $\triangle b'_0b''_0b$ black and keeping T' and T'' gray.)
3. (End of Processing) Do nothing. (This represents coloring $\triangle b'_0b''_0v$ white).

By Remark 2.1, to prove Theorem 4.1, it suffices to design a Bernoulli tester that takes $s = O(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon})$ samples in expectation and runs in time $O(\frac{1}{\epsilon^8})$. Our Bernoulli tester is Algorithm 2. In Algorithm 2, we use the following notation for the (relative) empirical error with respect to an input image M, a set of sampled pixels S, and the size parameter s. For an image M', let $\hat{d}(M') = \frac{1}{s} \cdot |\{u \in S : M[u] \neq M'[u]\}|$. For every region $R \subseteq [0..n)^2$, we let $\hat{d}_+(R) = \frac{1}{s} \cdot |\{u \in S \cap R : M[u] = 0\}|$, and $\hat{d}_-(R) = \frac{1}{s} \cdot |\{u \in S \cap R : M[u] = 1\}|$, i.e., the empirical error if we make R black/white, respectively.

Subroutine Best chooses the option with the smallest empirical relative error among those given in Definition 4.5, items 1-3. Its pseudocode is in the full version of this article.

Our set of reference polygons has two critical features. First, for each convex image there is a nearby reference polygon. It turns out that the empirical error for a region is proportional

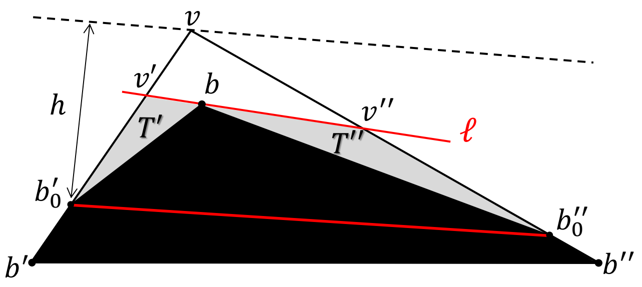

Figure 5 An illustration to Definition 4.5: Triangle $\triangle b'b''v$.

Algorithm 2: Bernoulli approximation algorithm for distance to convexity.

```
Input : parameters $n \in \mathbb{N}$, $\epsilon \in (0, 1/4)$; Bernoulli access to an $n \times n$ binary matrix $M$.
1  Set $s = \Theta(\frac{1}{\epsilon^2} \log \frac{1}{\epsilon})$. Include each image pixel in the sample $S$ w.p. $p = s/n^2$.
   // Run the algorithm to find $\hat{d}$, the smallest fraction of samples
      misclassified by a reference polygon in $P_\epsilon$. A dynamic programming
      implementation of the algorithm is given in the full version.
2  Let $W_{\ell_0}$ (resp., $W_{\ell_2}$) be the set of pixels of the image $M$ that lie either above $\ell_0$ or to
     the left of $b_0$ on $\ell_0$ (resp., either below $\ell_2$ or to the left of $b_2$ on $\ell_2$). Let $W_{\ell_1}$ (resp.,
     $W_{\ell_3}$) be the set of pixels of $M - W_{\ell_0} - W_{\ell_2}$ to the left of $\ell_1$ (resp., to the right of $\ell_3$).
3  Set $\hat{d} = 1$.
4  forall line-point pairs $(\ell_0, b_0), (\ell_2, b_2)$, where $\ell_0, \ell_2$ are horizontal lines do
5      Set $\hat{d}_{\text{left}} = 1$.
       // The best error for the region to the left of $b_0b_2$, between $\ell_0$ and $\ell_2$.
6      foreach line-point pair $(\ell_1, b_1)$, where $\ell_1$ is a vertical line do
7          Let $v_0$ (resp., $v_2$) be the point where $\ell_1$ intersects $\ell_0$ (resp., $\ell_1$ intersects $\ell_2$).
8          $\hat{d}_{\text{left}} = \min(\hat{d}_{\text{left}}, \hat{d}_-(W_{\ell_1}) + \hat{d}_+(\triangle b_0b_1b_2) + \mathsf{Best}(\triangle b_0b_1v_0) + \mathsf{Best}(\triangle b_1b_2v_2))$
9      Similarly to Steps 5–8, compute $\hat{d}_{\text{right}}$.
       // The best error for the region to the right of $b_0b_2$, between $\ell_0$ and $\ell_2$.
10     Compute $\hat{d} = \min(\hat{d}, \hat{d}_-(W_{\ell_0} \cup W_{\ell_2}) + \hat{d}_{\text{left}} + \hat{d}_{\text{right}})$.
11 return $\hat{d}$.
```

to the square root of its area. The second key feature of our reference polygons is that, for each of them, the set of considered triangles, $\mathbf{T}_{\text{end}}$, has small $\sum_{T \in \mathbf{T}_{\text{end}}} \sqrt{A(T)}$, where $A(T)$ denotes the area of triangle T. The proofs of both features, as well as the analysis of the empirical error, are quite technical and appear in the full version of this article. ◀

Here, we state and partially prove a lemma that puts together different parts of the analysis. It makes it clear why the empirical error of each region is proportional to the square root of its area which is, as explained in Footnote 4, a proxy for the number of pixels in it.

▶ **Lemma 4.6.** *With probability at least $2/3$ over the choice of the samples taken by Algorithm 2, $|\hat{d}(M') - dist(M, M')| \leq 5\epsilon/6$ for all reference polygons M'.*

Proof. Consider a region $R = (R_+, R_-)$, partitioned into two regions R_+ and R_-, such that in some step, the algorithm checks the assumption that R_+ is black and R_- is white, i.e., evaluates $\hat{d}_+(R_+) + \hat{d}_-(R_-)$. Let $\mathbf{R}$ be the set of all such regions R. We show that with probability at least $2/3$, the estimates $\hat{d}_+(R_+) + \hat{d}_-(R_-)$ are accurate on all regions in $\mathbf{R}$.

Fix $R = (R_+, R_-) \in \mathbf{R}$. Let Γ be the set of misclassified pixels in R, i.e., pixels in R_+ which are white in M and pixels in R_- which are black in M. Define $\gamma = |\Gamma|/n^2$. Algorithm 2 approximates γ by $\hat{d}_+(R_+) + \hat{d}_-(R_-) = \frac{1}{s}|\Gamma \cap S|$. Equivalently, it uses the estimate $\frac{1}{p}|\Gamma \cap S|$ for $|\Gamma|$ (recall that $p = s/n^2$). The error of the estimate is $err_S(R) = \frac{1}{p}|\Gamma \cap S| - |\Gamma|$.

▶ **Claim 4.7.** $\Pr[|err_S(R)| > \sqrt{\gamma} \cdot c\epsilon n^2] \leq 2\exp(-\frac{3}{8}c^2\epsilon^2 s)$*, where* $c = 1/21$.

Proof. For each pixel u, define random variables χ_u and X_u, where χ_u is the indicator for the event $u \in S$ (i.e., a Bernoulli variable with the probability parameter p), whereas $X_u = \frac{\chi_u}{p} - 1$. Then our estimate of $|\Gamma|$ is $\frac{1}{p}|\Gamma \cap S| = \frac{1}{p}\sum_{u\in\Gamma}\chi_u$, whereas $err_S(R) = \sum_{u\in\Gamma} X_u$. We use Bernstein inequality to bound $\Pr[\sum_{u\in\Gamma} X_u > \sqrt{\gamma} \cdot c\epsilon n^2]$. The variables X_u are identically distributed. The maximum value of $|X_u|$ is $a = \frac{1-p}{p}$. Note that $\mathbb{E}[X_u^2] = \frac{1}{p^2}\mathbb{E}[(\chi_u - p)^2] = \frac{1}{p^2}\operatorname{Var}[\chi_u] = \frac{1-p}{p} = a$. Assume w.l.o.g. that $z < |\Gamma|$. (If $z \geq |\Gamma|$ then $\sum_{u\in\Gamma} X_u$ cannot exceed z, and the probability we are bounding is 0.) By Bernstein inequality,

$$\begin{aligned}\Pr\left[\sum_{u\in\Gamma} X_u > z\right] &\leq \exp\left(\frac{-z^2/2}{a|\Gamma| + a\cdot z/3}\right) < \exp\left(-\frac{3}{8}\cdot\frac{z^2\cdot p}{|\Gamma|}\right) = \exp\left(-\frac{3}{8}\frac{\gamma c^2\epsilon^2 n^4}{\gamma n^2}\frac{s}{n^2}\right)\\ &= \exp(-\frac{3}{8}c^2\epsilon^2 s).\end{aligned}$$

The second inequality holds because $a < 1/p$ and $z < |\Gamma|$. The equalities are obtained by substituting the expressions for $z, |\Gamma|$, and p, and simplifying. By symmetry, $\Pr[|err_S(R)| \geq z] \leq 2\exp(-\frac{3}{8}c^2\epsilon^2 s)$. ◀

The rest of the proof appears in the full version of this article. ◀

5 Distance Approximation to the Nearest Connected Image

To define *connectedness*, we consider *the image graph* G_M of an image M. The vertices of G_M are $\{(i,j) \mid M[i,j] = 1\}$, and two vertices (i,j) and (i',j') are connected by an edge if $|i-i'| + |j-j'| = 1$. In other words, the image graph consists of black pixels connected by the grid lines. The image is *connected* if its image graph is connected.

▶ **Theorem 5.1.** *There is a block-uniform ϵ-additive distance approximation algorithm for connectedness with sample complexity $O(\frac{1}{\epsilon^4})$ and running time* $\exp\left(O\left(\frac{1}{\epsilon}\right)\right)$.

The first idea in our algorithms for connectedness is that we can modify an image by superimposing a grid on it, and as a result obtain a nearby image whose distance to connectedness is determined by the properties of individual squares into which the grid lines partition the image. The squares and the relevant property of the squares are defined next.

For a set $S \subset [0..n)^2$ and $(i,j) \in [0..n)^2$, we define $S + (i,j) = \{(x+i, y+j) : (x,y) \in S\}$.

▶ **Definition 5.2** (Squares and grid pixels). Fix a side length $n \equiv 1 \pmod{r}$. For all integers $i, j \in [0..n-r)$, where i and j are divisible by r, the $(r-1)\times(r-1)$ image that consists of all pixels in $[r-1]^2 + (i,j)$ is called an *r-square* of M. The set of all r-squares of M is denoted S_r.

The pixels that do not lie in any squares of S_r, i.e., pixels (i,j) where i or j is divisible by r, are called *grid pixels*. The set of all grid pixels is denoted by GP_r.

▶ **Claim 5.3.** $|GP_r| \leq 2n^2/r$.

Note that a square consists of pixels of an r-block, with the pixels of the first row and column removed. Therefore, a block-uniform algorithm can obtain a uniformly random r-square.

Recall the definition of the border of an image from Section 2.

▶ **Definition 5.4** (Border connectedness). A (sub)image S is *border-connected* if for every black pixel (i,j) of S, the image graph G_S contains a path from (i,j) to a pixel on the border. The property *border connectedness,* denoted $\mathcal{C}'$, is the set of all border-connected images.

The main idea behind Algorithm 3, used to prove Theorem 5.1, is to relate the distance to connectedness to the distance to another property, which we call *grid connectedness.* The latter distance is the average over squares of the distances of these squares to *border connectedness.* The average can be easily estimated by looking at a sample of the squares.

W.l.o.g. assume that $n \equiv 1 \pmod{4/\epsilon}$. (Otherwise, we can pad the image with white pixels without changing whether it is connected and adjust the accuracy parameter.)

```
Algorithm 3: Distance approximation to connectedness.
  Input  : $n \in \mathbb{N}$ and $\epsilon \in (0, 1/4)$; block-sample access to an $n \times n$ binary matrix $M$.
1 Sample $s = 4/\epsilon^2$ squares uniformly and independently from $S_{4/\epsilon}$ (see Definition 5.2).
  // Tho do this draw random blocks from the $4/\epsilon$-partition of $[0..n)^2$.
2 For each such square $S$, compute $dist(S, \mathcal{C}')$, where $\mathcal{C}'$ is border connectedness (see
   Definition 5.4). Let $\hat{d}_{\text{squares}}$ be the average of computed distances $dist(S, \mathcal{C}')$.
3 return $\hat{d} = \left((1 - \frac{\epsilon}{4})(1 - \frac{1}{n})\right)^2 \cdot \hat{d}_{\text{squares}}$.
```

▶ **Definition 5.5.** Fix $\epsilon \in (0, 1/4)$. Let image M_ϵ be a *gridded image* obtained from image M as follows:

$$M_\epsilon[i,j] = \begin{cases} 1 & \text{if } (i,j) \text{ is a grid pixel from } \mathrm{GP}_{4/\epsilon}; \\ M[i,j] & \text{otherwise.} \end{cases}$$

Let $\mathcal{C}$ be the set of all connected images. For $\epsilon \in (0, 1/4)$, define *grid connectedness* $\mathcal{C}_\epsilon = \{M \mid M \in \mathcal{C}, \text{ and } M[i,j] = 1 \text{ for all } (i,j) \in \mathrm{GP}_{4/\epsilon}\}$.

▶ **Lemma 5.6.** *Let $d_M = dist(M, \mathcal{C})$ and $d_\epsilon = dist(M_\epsilon, \mathcal{C}_\epsilon)$. Then $d_M - \frac{\epsilon}{2} \le d_\epsilon \le d_M$. Moreover,*

$$d_\epsilon = \Big((1 - \frac{\epsilon}{4})(1 - \frac{1}{n})\Big)^2 \cdot \frac{1}{|S_{4/\epsilon}|} \sum_{S \in S_{4/\epsilon}} dist(S, \mathcal{C}').$$

Proof. First, we prove that $d_\epsilon \le d_M$. Let M' be a connected image such that $dist(M, M') = d_M$. Then M'_ϵ, the gridded image obtained from M', satisfies $\mathcal{C}_\epsilon$. Since $dist(M_\epsilon, M'_\epsilon) \le d_M$, it follows that $d_\epsilon \le d_M$. Now we show that $d_M - \frac{\epsilon}{2} \le d_\epsilon$. Let $M''_\epsilon \in \mathcal{C}_\epsilon$ be such that $dist(M_\epsilon, M''_\epsilon) = d_\epsilon$. Then $M''_\epsilon \in \mathcal{C}$ and, by Claim 5.3, $dist(M, M''_\epsilon) \le |\mathrm{GP}_{4/\epsilon}|/n^2 + d_\epsilon \le \epsilon/2 + d_\epsilon$, implying $d_M \le \epsilon/2 + d_\epsilon$, as required.

Finally, observe that to make M_ϵ satisfy $\mathcal{C}_\epsilon$, it is necessary and sufficient to ensure that each square satisfies $\mathcal{C}'$. In other words,

$$d_\epsilon n^2 = \sum_{S \in S_{4/\epsilon}} Dist(S, \mathcal{C}') = (4/\epsilon - 1)^2 \sum_{S \in S_{4/\epsilon}} dist(S, \mathcal{C}').$$

Since $|S_{4/\epsilon}| = (\frac{n-1}{4/\epsilon})^2$, the desired expression for d_ϵ follows. ◀

The rest of the analysis is completed in the full version of this article.

References

1 Imre Barany. Extremal problems for convex lattice polytopes: a survey. *Contemporary Mathematics*, 2000.

2 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Testing convexity of figures under the uniform distribution. In *SoCG*, 2016.

3 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *STOC*, pages 164–173, 2014. `doi:10.1145/2591796.2591887`.

4 Avrim Blum. Machine learning theory. Lecture notes. URL: `http://www.cs.cmu.edu/~avrim/ML12/lect0201.pdf`.

5 Andrea Campagna, Alan Guo, and Ronitt Rubinfeld. Local reconstructors and tolerant testers for connectivity and diameter. In *APPROX-RANDOM*, pages 411–424, 2013. `doi:10.1007/978-3-642-40328-6_29`.

6 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. `doi:10.1007/3-540-44676-1_22`.

7 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. `doi:10.1007/3-540-45253-2_15`.

8 Andrzej Ehrenfeucht, David Haussler, Michael J. Kearns, and Leslie G. Valiant. A general lower bound on the number of examples needed for learning. *Inf. Comput.*, 82(3):247–261, 1989.

9 Eldar Fischer and Lance Fortnow. Tolerant versus intolerant testing for boolean properties. *Theory of Computing*, 2(1):173–183, 2006. `doi:10.4086/toc.2006.v002a009`.

10 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. `doi:10.1145/285055.285060`.

11 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002. `doi:10.1007/s00453-001-0078-7`.

12 David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inf. Comput.*, 100(1):78–150, 1992. `doi:10.1016/0890-5401(92)90010-D`.

13 Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio. Agnostically learning halfspaces. *SIAM J. Comput.*, 37(6):1777–1805, 2008.

14 Michael J. Kearns, Robert E. Schapire, and Linda Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994. `doi:10.1007/BF00993468`.

15 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. `doi:10.1109/TPAMI.2010.165`.

16 Simon Korman, Daniel Reichman, and Gilad Tsur. Tight approximation of image matching. *CoRR*, abs/1111.1713, 2011. URL: `http://arxiv.org/abs/1111.1713`.

17 Simon Korman, Daniel Reichman, Gilad Tsur, and Shai Avidan. Fast-match: Fast affine template matching. In *CVPR*, pages 2331–2338, 2013. `doi:10.1109/CVPR.2013.302`.

18 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. Syst. Sci.*, 72(6):1012–1042, 2006. `doi:10.1016/j.jcss.2006.03.002`.

19 Sofya Raskhodnikova. Approximate testing of visual properties. In *RANDOM-APPROX*, pages 370–381, 2003. `doi:10.1007/978-3-540-45198-3_31`.

20 Dana Ron and Gilad Tsur. Testing properties of sparse images. *ACM Trans. Algorithms*, 10(4):17:1–17:52, 2014. `doi:10.1145/2635806`.

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

22 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.

23 Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.