Property testing of data dimensionality*

Robert Krauthgamer†　　　Ori Sasson‡

Abstract

Data dimensionality is a crucial issue in a variety of settings, where it is desirable to determine whether a data set given in a high-dimensional space adheres to a low-dimensional structure. We study this problem in the framework of property testing: Given a query access to a data set S, we wish to determine whether S is low-dimensional, or whether it should be modified significantly in order to have the property. Allowing a constant probability of error, we aim at algorithms whose complexity does not depend on the size of S.

We present algorithms for testing the low-dimensionality of a set of vectors and for testing whether a matrix is of low rank. We then address low-dimensionality in metric spaces. For vectors in the metric space l_1, we show that low-dimensionality is not testable. For l_2, we show that a data set can be tested for having a low-dimensional structure, but that the property of approximately having such a structure is not testable.

1 Introduction

The analysis of large volumes of complex data is required in various disciplines. Such complex data is frequently represented by vectors in a high-dimensional space, e.g., by applying feature extraction. High-dimensional data is notoriously difficult to work with, as the complexity of many commonly used operations is highly dependent (e.g. exponentially) on the dimension.

Real-life data sets often adhere to a low-dimensional structure, whose extraction is of practical importance. This gives rise to (the family of) *dimensionality reduction* problems, where we want to find, for a given set S of points, a good representation in a space of low dimension $d > 0$. Throughout, we consider the dimension d to be fixed (with respect to the size of S).

One prominent interpretation for the aforementioned low-dimensional structure is as a linear space. That is, the data set S is given in a (high-dimensional) vector space and one wishes to find whether the vectors of S lie (exactly or approximately) in a low-dimensional linear (or affine) subspace. For example, Singular Value Decomposition (SVD) computes a low-rank matrix that approximates an input (real-valued) matrix optimally (in a certain sense), thus finding in the input matrix an "almost" low-dimensional linear structure. SVD has many applications in information retrieval, see e.g. [DDL$^+$90, Kle98]. Two related techniques that are commonly used in practice for dimensionality reduction are Principle Component Analysis (PCA) and Multidimensional Scaling (MDS).

---

*This is a preprint of a paper that will appear in SODA 2003. (Copyright owned as specified in the conference proceedings.)

†International Computer Science Institute, Berkeley, CA 94704, USA and Computer Science Division, University of California, Berkeley, CA 94720, USA. Part of this work done while the author was at the Hebrew University of Jerusalem. Supported in part by NSF grants CCR-9820951 and CCR-0121555 and DARPA cooperative agreement F30602-00-2-0601. Email: `robi@cs.berkeley.edu`

‡School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem 91904, Israel. Supported in part by the Israeli National Science Foundation. Email: `ori@cs.huji.ac.il`

Another way to look at the issue of low-dimensional structure is through the prism of finite metric spaces.[1] That is, given a data set S that consists of points in a metric space (such as l_p^m, i.e. $\mathbb{R}^m$ equipped with the l_p-norm), one wishes to realize the distances among the points of S (exactly or approximately) by points of a low-dimensional space (say l_p^d). For example, the "flatenning lemma" of Johnson and Lindenstrauss [JL84] (see also [Ind01]) shows that every n-point l_2-metric can be embedded in $l_2^{O(\log n)}$ with an arbitrarily small (fixed) distortion of distances.

Interestingly, when the data consists of vectors in a Euclidean space l_2, there is a close relation between linear and metric low-dimensional spaces, but it is not robust. It is clear that a set of points in l_2 lies in a d-dimensional linear subspace if and only if it can be embedded isometrically in l_2^d. However, the situation is completely different when an almost low-dimensional structure is considered. For instance, a small perturbation to the vectors is not necessarily related to a small distortion of distances.

An open question in the area of low-dimensional metric spaces is that of a finite point criterion for l_p^d-embeddability. Namely, what is the minimum integer $f_p(d)$ (called the order of congruence of l_p^d) such that any metric space is l_p^d-embeddable if and only if all subspaces of it on $f_p(d)$ points are l_p^d-embeddable. For $p = 2$, Menger [Men28] shows that $f_2(d) = d + 3$ for all $d \geq 1$. For $p = 1$ and every $d > 2$, Bandelt et al. [BCL98] show that $f_1(d) \geq d^2 - 1$ (see also [DL97, Chapter 11]) but it is not even known whether $f_1(d)$ is finite. Our results for l_1 and l_2 spaces establish somewhat similar bounds for a relaxed version of this question.

Property testing. We study relaxed versions of dimensionality reduction decision problems, in the framework of *property testing*. Instead of determining whether S has a certain low-dimensionality property P *or not*, the goal is to determine (with high probability) whether S has the property P, *or it should be modified significantly* in order to have P. That is, given query access to the entries of (a representation of) S, we wish to determine whether S has the property P, or whether S should be modified in at least an ϵ-fraction of the entries in order to have the property. In the latter case, S is called *ϵ-far* from having the property P.[2]

For these relaxed problems, it is desirable to obtain algorithms that are significantly more efficient (in terms of query complexity and running time) than those required for exactly deciding the property. In particular, we seek an algorithm whose complexity does not depend on the size of the data set but only on ϵ (and on the property P), and thus has a sublinear complexity. If such an algorithm exists, we say that the property P is *testable*.[3]

1.1 Our results

Linear structure of low dimension. We show that the (data set) property of having a low-dimensional linear (or affine) structure is testable in rather general settings. Specifically, we show the following in Section 2.

- Testing whether vectors $v_1, \ldots, v_n$ (in a vector space V) lie in a linear (or affine) subspace of dimension at most d can be achieved by an algorithm that queries $O(d/\epsilon)$ vectors.

[1]Throughout, when we refer to a metric we mean a *semi-metric*, i.e., the distance between distinct elements in the metric space is allowed to be 0.

[2]Property testing investigates the interplay between a global property P of the input and the local entries of its representation. This relationship is exploited in several areas such as secret sharing and probabilistically checkable proofs (PCPs).

[3]Some papers define a property to be testable if it has a testing algorithm with sublinear (query) complexity.

- Testing whether a matrix $A_{m\times n}$ (over a field F) has rank at most d can be achieved by an algorithm that queries the entries of an $O(d/\epsilon) \times O(d/\epsilon)$ submatrix.

These results may be of independent interest, as they consider fundamental algebraic properties. In particular, they are related to (but different from) testing multi-linear and low-degree codes, which has found many applications in secret sharing and in probablistically checkable proofs (PCPs).

Metric structure of low dimension. We address testing for low-dimensionality in the metric spaces l_2 and l_1. First, testing algorithms for low-dimensionality of l_2-metrics follow immediately by specializing our results regarding linear structure.[4] We then show that, in contrast, a similar property in l_1-metrics is not testable, and that the property of *approximately* having a low-dimensional structure in l_2-metrics is also not testable. Specifically, we show the following in section 3.

- Testing whether vectors $v_1, \ldots, v_n \in l_1^m$ can be embedded into l_1^d, requires querying $\Omega(\sqrt[4]{n})$ vectors, even for $d = 1$, $m = 2$.
- Testing whether vectors $v_1, \ldots, v_n \in l_2^m$ can be perturbed by $\delta > 0$ so that they embed isometrically in l_2^d, requires querying $\Omega(\min\{\sqrt{n}, \sqrt{m/\log m}\})$ vectors, even for $d{=}0$, $\epsilon{=}1/2$.
- Testing whether the vectors $v_1, \ldots, v_n \in l_2^m$ can be embedded into l_2^d with distortion $\Delta \geq 1$ requires querying $\Omega(\sqrt{n/\Delta})$ vectors, even for $d = 1$, $\epsilon = O(1/\Delta)$, and arbitrary fixed Δ.
- Testing whether a matrix $M_{n\times n}$ is the distance matrix of a d-dimensional Euclidean metric can be achieved by an algorithm that queries the entries of an $O(d/\epsilon) \times O(d/\epsilon)$ submatrix. (This offers a slight improvement over the query complexity of $O(d\log(d)/\epsilon \times d\log(d)/\epsilon)$ shown in [PR01].)

Small norm. In the context of testing dimensionality, the norm of a matrix may be interesting when evaluating the difference of an input matrix and a low-dimensional approximation of it, see e.g. [FKV98, AM01]. In Section 4 we show the following for testing whether the *Frobenius norm* $(\sum_{ij} A_{ij}^2)^{1/2}$ of a matrix A is small.

- Testing whether a matrix $A_{m\times n}$ has Frobenius norm at most r can be achieved by an algorithm that queries $O(\epsilon^{-3}\log 1/\epsilon)$ entries of A.

This result applies also to testing whether the l_p-norm of a vector is small (for any fixed $p \geq 1$), which may be of independent interest.

Approximation of relaxed search problems. Our testing algorithms can be extended to solve relaxations of the search problems that correspond to the tested properties. Consider, for example, the problem of finding a basis for the vector space spanned by the rows of a low-rank matrix $A_{m\times n}$. Our analysis in the proof of Theorem 2.3 implies an algorithm that finds, with high probability, a basis for the span of a matrix that agrees with A on all but at most an ϵ-fraction of the entries. The running time of this algorithm is $O(\min\{m,n\} \cdot (\text{rank}A)^2/\epsilon^2)$, and is thus sublinear in the matrix size $n \cdot m$. Details omitted from this version of the paper.

[4]Note that in Euclidean spaces the metric dimension coincides with the affine dimension.

1.2 Related work

Property testing was first defined by Rubinfeld and Sudan [RS96] in the context of algebraic properties of functions. Goldreich, Goldwasser and Ron [GGR98] initiated the study of this notion for combinatorial objects, focusing mainly on graph properties. Recently, property testing was studied in various other settings such as clustering, metric spaces, geometric objects, strings, and distributions, see e.g. [Ron01, Fis01] and the references therein.

Our work is particularly related to that of Parnas and Ron [PR01] on testing metric properties. They consider the problem of testing whether a distance matrix represents a tree metric, an ultra-metric, an approximate ultrametric, or a low-dimensional Euclidean metric. The work in this paper touches upon the latter family of metrics but we focus mainly on other input representations. Also related is testing of other properties of matrices and vectors; for instance, testing the monotonicity of a matrix is studied in [EKK+00, DGL+99, FN01].

Random sampling methods are used in [FKV98, AM01] for fast computation of a low-rank approximation of a matrix; their results suggest that the structure of a low-rank matrix is typically revealed by sampling a relatively few entries. However, there are some crucial differences between these low-rank approximations and our work. One major difference is in the sampling requirements. The algorithms we study are essentially bound to oblivious uniform sampling, while the algorithms of [FKV98] use a sample according to an input-dependent distribution (which may be computed in linear time by preprocessing), and the algorithms of [AM01] require a sample whose size depends (at least polylogarithmically) on the input. Another difference is in the measure of farness from a low-rank matrix. In low-rank approximations, two matrices are considered close to each other if the norm of their difference is small. In contrast, our definitions measure the fraction of entries in which the two matrices differ, bearing no significance to the magnitude of the differences.[5]

Random sampling is also used in [DGGZ02] to determine the dimension of a geometric shape. These algorithms use sampling to explore the dimensionality of the input, similar to our work, but from a different perspective; the objects of investigation in [DGGZ02] are shapes and not arbitrary finite sets of points from a metric or linear space, and the problems they study are not property testing problems.

We point out that there other notions of dimensionality reduction. For instance, combinatorial feature selection is the problem of projecting on a subset of the coordinates (i.e., a subset of the features), and the goal is to preserve the data properties, such as clustering and entropy, see e.g. [CGK+00].

1.3 Preliminaries

We consider several natural representations of high-dimensional data sets, but we always assume that both the querying mechanism (i.e., what queries are available to the algorithm) and the farness measure (i.e., how far is an instance from having the property) correspond to the same representation. Below we formally define the notions of ϵ-farness and of a testing algorithm.

Definition 1 (Distance from a property). *Let P be a property of objects that consist of entries (such as matrices or vectors). An object is called ϵ-*far from having the property *P if an ϵ-fraction of the object's entries should be modified for the object to have the property.*

[5] The "right" measure of farness clearly depends on the application. For instance, measuring the norm of the difference may prevail in real-valued data where an additive noise is expected, while measuring the fraction of entries that differ may be more suitable in discrete data where an arbitrary (or even adversarial) partial corruption is considered.

Definition 2 (Property testing algorithm). *Let P be a property of objects that consist of entries. A* property testing algorithm *for P is an algorithm that, given a query access to the entries of an input object and a distance $\epsilon > 0$, accepts with probability at least $2/3$ if the object has the property P, and rejects with probability at least $2/3$ if the object is ϵ-far from having the property.*

The above definition allows the algorithm to have a two-sided error. A more restricted definition is one where the algorithm's error is one-sided, and then the algorithm must always accept inputs having the property P. We generally consider two-sided error algorithms.

1.4 A technical lemma

The following lemma is used several times in the analysis of our testing algorithms.

Lemma 1.1. *Let $d \geq 0$ and $\epsilon > 0$. Assume that $0 \leq X_0 \leq X_1 \leq X_2 \leq \ldots$ is a sequence of random variables satisfying that for all $t \geq 0$,*

$$\Pr\left[X_{t+1} \geq X_t + 1 \mid X_t \leq d\right] \geq \epsilon. \tag{1}$$

Then for $t^* \geq 8(d+1)/\epsilon$,

$$\Pr\left[X_{t^*} \leq d\right] < 1/3. \tag{2}$$

Proof. We first show that the variables X_t can be modified into binomially distributed random variables $Z_t \sim B(t, \epsilon)$ such that $\Pr[X_t \leq d] \leq \Pr[Z_t \leq d]$ for all $t \geq 0$. Observe that when X_t is larger than d, its exact value is irrelevant for the last inequality, so we assume without loss of generality that once X_t is larger than d, it increases by 1 with probability ϵ, independently of all other events.

When X_t is not larger than d, We have by (1) that X_t increases by at least 1 with probability at least ϵ. Hence, we can define random variables $Z_0 \leq Z_1 \leq \ldots$ such that $Z_t \sim B(t, \epsilon)$ and Z_t is dominated by X_t, i.e. $\Pr[X_t \leq y] \leq \Pr[Z_t \leq y]$ for all y.

Our choice of $t^* = 8(d+1)/\epsilon$ implies $\mathbb{E}[Z_{t^*}] \geq 8d$. Using the Chernoff bound (see e.g. [MR95]) and that $d \geq 0$ we have

$$\Pr[Z_{t^*} \leq d] \leq e^{-(7/8)^2 \cdot 8(d+1)/2} < e^{-3(d+1)} < 1/3.$$

It follows that $\Pr[X_{t^*} \leq d] < 1/3$, as claimed. $\square$

2 Low-dimensional linear structure

In this section we show algorithms for testing whether a data set has a low-dimensional linear structure. Very briefly, these algorithms query a random sample of the input and accept if the sample satisfies the tested property P. The proofs follow by considering the sample to be an iterative process, which is a relatively standard technique in this area.

2.1 A low-dimensional vector subspace

The following theorem shows that it is possible to test whether a set of vectors has a low linear (or affine) dimension. We note that the result holds for either finite or infinite vector space V. As was pointed out to us by Dick Karp, this result easily extends also to matroids.[6]

[6] Let $M = (E, I)$ be a matroid with ground set E and (set of) independent subsets I. The rank of $S \subseteq E$ is the size of the largest independent set contained in S (see e.g. [Oxl92]). The analog formulation of theorem 2.1 for

Theorem 2.1. *There is an algorithm for testing whether a set S of n vectors in a vector space V lie in a linear (or affine) subspace of dimension d, or whether S is ϵ-far from having this property (in the sense at least ϵn vectors need to be modified/removed). This algorithm queries $O(d/\epsilon)$ randomly chosen vectors of S.*

Proof. The testing algorithm works as follows. Given a set S and an integer d, the algorithm queries $O(d/\epsilon)$ randomly chosen vectors of S, and accepts if and only if they lie in a linear (or affine, respectively) subspace of dimension at most d. Clearly, if the dimension of the vectors of S is at most d, then the algorithm always accepts. (It follows that this algorithm has one-sided error.)

Consider next a set of n vectors that is ϵ-far from residing in a subspace of dimension d. For the purpose of our analysis, we think of the algorithm as if it has $O(d/\epsilon)$ iterations. Starting with an empty sample, the algorithm iteratively augments the sample with one additional vector from S. (For affine dimension, assume that we start with one vector.) Denote by U_t the set of sampled vectors that is obtained after $t \geq 0$ iterations, and let X_t be the dimension of the linear (or affine) subspace spanned by the vectors of U_t.

Lemma 2.2. *Let S be a set of n vectors that is ϵ-far from residing in a subspace of dimension d. Then,* $\Pr[X_{t+1} = X_t + 1 \mid X_t \leq d] \geq \epsilon$.

Proof. Consider U_t as above and suppose its dimension is $X_t \leq d$. Since S is ϵ-far from residing in a subspace of dimension d, we have that removal (or modification) of ϵn vectors of S cannot result in a set of vectors that lies in a d-dimensional subspace, and obviously not in a subspace of dimension $X_t \leq d$. It follows that at least ϵn vectors of S lie outside the subspace spanned by U_t. Thus, with probability at least ϵ one of these vectors is chosen to augment U_t, yielding U_{t+1} of dimension $X_{t+1} = X_t + 1$. ☐

We can now complete the proof of Theorem 2.1. By combining Lemmas 1.1 and 2.2, we have that if S is ϵ-far from residing in a subspace of dimension d, then for $t^* = 8(d+1)/\epsilon$ we have $\Pr[X_{t^*} \geq d+1] > 2/3$. It follows that with probability at least $2/3$ the testing algorithm rejects S after $t^* = O(d/\epsilon)$ iterations. ☐

2.2 Testing matrix rank

The next theorem shows that the matrix property of having a low rank is testable. We note that the result holds for either finite or infinite fields F.

Theorem 2.3. *There is an algorithm for testing whether a matrix $A_{m \times n}$ over a field F has rank at most d or whether A is ϵ-far from having this property (in the sense that at least an ϵ-fraction of entries of A need to be modified). This algorithm queries a randomly chosen $O(d/\epsilon) \times O(d/\epsilon)$ submatrix of A.*

Proof. The testing algorithm works as follows. Given a matrix A and an integer d, the algorithm selects at random $O(d/\epsilon)$ rows and $O(d/\epsilon)$ columns, queries the resulting submatrix, and accepts if and only if the rank of this submatrix is at most d. Clearly, if A has rank at most d then every submatrix of A has rank at most d and the testing algorithm always accepts. (It follows that this algorithm has one-sided error.)

---

matroids states that there is an algorithm for testing whether a set S of n elements in the ground set of a matroid M has rank at most d, or whether S is ϵ-far from having this property.

Consider next an input matrix A that is ϵ-far from any matrix of rank at most d. For the sake of analysis, we think of the algorithm as if it starts with an "empty" 0×0 submatrix, and iteratively augments the current submatrix by one random row and by one random column (i.e., the choice is without replacement).

To analyze a single iteration in the algorithm, we use the lemma below. Denote by B_t the $t \times t$ submatrix of A that is considered at iteration $t \geq 0$ (i.e., after t augmentations), and let $X_t = \text{rank}(B_t)$ for $t \geq 0$.

Lemma 2.4. *Let A be ϵ-far from any matrix of rank at most d. Then*

$$\Pr\left[X_{t+1} > X_t) \mid X_t \leq d\right] \geq \epsilon/3. \tag{3}$$

Proof. Consider an arbitrary submatrix B_t that satisfies $\text{rank}(B_t) \leq d$. We say that a row is an *augmenting row* for the submatrix B_t if this row was not chosen in the first t iterations. An augmenting row for the submatrix B_t is said to be *consistent* with B_t if the augmentation of B_t to this row does not increase the rank of B_t. It is straightforward that if at least $\epsilon m/3$ augmenting rows are not consistent with B_t, then the probability that the algorithm augments B_t with one of these rows is at least $\epsilon/3$, and thus (3) holds.

Using analogous definitions for the columns, we have that if at least $\epsilon n/3$ augmenting columns are not consistent with B_t, then the probability that the algorithm augments B_t with one of these columns is larger than $\epsilon/3$, and thus (3) holds.

We say that entry A_{ij} is a *strongly-augmenting entry* for the submatrix B_t if each of row i and column j (separately) is both augmenting and consistent for B_t. A strongly-augmenting entry A_{ij} is said to be *consistent* with B_t if the augmentation of the submatrix B_t with row i and column j (simultaneously) does not increase the rank of B_t. If the number of strongly-augmenting entries that are not consistent with B_t is at least $\epsilon nm/3$, then the probability that the algorithm augments B_t with one of these entries A_{ij} (i.e. chooses its row i and its column j), is larger than $\epsilon/3$, and thus (3) holds.

We complete the proof of Lemma 2.4 by showing that at least one of the three cases mentioned above must hold. Assume to the contrary that (i) less than $\epsilon m/3$ augmenting rows are not consistent with B_t, (ii) less than $\epsilon n/3$ augmenting columns are not consistent with B_t, and (iii) less than $\epsilon nm/3$ strongly-augmenting entries are not consistent with B_t. Suppose that we change to zero all the entries in rows and columns that are augmenting and not consistent with B_t, and that we change the value of every strongly-augmenting entry A_{ij} that is not consistent with B_t to a value that is consistent with B_t. (Such a value always exists, because augmenting B_t with row i adds a row that is a linear combination of the rows already in B_t, so we can take the same linear combination also in column j.) It is straightforward that the rank of the resulting matrix is exactly $\text{rank}(B_t) \leq d$, while the total number of entries changed is less than ϵnm, which contradicts the assumption that A is ϵ-far from any matrix of rank at most d. □

We can now complete the proof of Theorem 2.3. By combining Lemmas 1.1 and 2.4 (with $\epsilon' = \epsilon/3$), we have that A is ϵ-far from any matrix of rank at most d, then $\Pr\left[X_{t^*} \geq d+1\right] > 2/3$ for $t^* = 24(d+1)/\epsilon$. It follows that with probability at least $2/3$ the testing algorithm rejects A after $t^* = O(d/\epsilon)$ iterations. □

3 Metric structure of low dimension

In this section we investigate testing for low-dimensionality from the perspective of metric spaces. Observe that a set of points in a Euclidean space l_2^m can be embedded isometrically into l_2^d if and only if they lie in a d-dimensional affine space. Hence, applying Theorem 2.1 on the vector space $\mathbb{R}^m$ gives an algorithm for testing the dimensionality of a set of vectors in l_2^m. Below, we show that a similar result does not hold for l_1 metrics. We then consider two notions of being an "approximate" l_2^d-metric. Finally, we examine the property of being a low-dimensional l_2-metric, when the input is given as a distance matrix.

3.1 Low-dimensional l_1-metric

We turn to the problem of testing whether a set of n points in l_1 can be embedded into l_1^d for a fixed $d \geq 0$. The next lemma shows that no (two-sided error) testing algorithm for this problem can query a number of points that is independent of n.

Lemma 3.1. *Any algorithm for testing whether a set S of n vectors in l_1^m can be embedded into l_1^d, or whether S is ϵ-far from having this property (in the sense that at least an ϵ-fraction of the vectors need to be modified/removed), has to query $\Omega(\sqrt[4]{n})$ vectors.*

Proof. Consider first the case where $d = 1$, $m = 2$. We construct sets S and S', each consisting of n vectors in l_1^2, as follows. S' "forms" three diagonal lines parallel to each other in the plane $\mathbb{R}^2$, and is given by

$$S' = \Big\{(i, i+j) \ : \ i = 1, \ldots, n/3 \ ; \ j = 0, 1, 2\Big\}.$$

To construct S, choose at random $t = \sqrt{n}/10$ points of S' and add n/t copies of each one of these points to S.

We claim that with probability at least $8/9$ the set S can be embedded isometrically in l_1^1. Indeed, the probability that S contains two points (x, y) and (x', y') with either $|x - x'| \leq 1$ or $|y - y'| \leq 1$ is at most $\frac{8t^2}{n-t} \leq 1/9$ (since each point added to S has probability at most $\frac{8t}{n-t}$ to form such a pair with one of the points already placed in S). Thus, S has no such pairs with probability at least $8/9$. When this event happens, ordering the points of S by their first coordinate and by their second coordinate yield the same order, yielding a geodesic line in l_1^2 that goes through all the points of S. This shows that S can be embedded isometrically into l_1^1 (i.e., into $\mathbb{R}$), and the claim follows.

Let us show that S' is $1/6$-far from being embeddable in l_1^1. Any three points that form the endpoints of a "$\perp$", namely $\{(x, y), (x+1, y+1), (x+2, y)\}$, cannot be embedded into l_1^1 (since in l_1^1, which is identical to $\mathbb{R}$, one of every three points must be on a geodetic line between the other two.) There are (at least) $n/6$ disjoint triplets of this type in S', and in order for S to be embeddable into l_1^1 at least one vector out of each triplet must be modified/removed. Hence, S' is $1/6$-far from being embeddable in l_1^1.

Assume for contradiction that there exists a (possibly two-sided error) testing algorithm that queries at most $s \leq \sqrt[4]{n}/10$ vectors. Let the algorithm's input be the set of vectors S, permuted at random. The random permutation implies that the s queried vectors are s random vectors from S, and thus the probability (over the randomness in choosing the permutation and in the testing algorithm) of querying more than one vector from at least one of the t groups (of copies) in S is at most $\frac{s^2 n/t}{n-s} \leq 1/9$ (each of the s queries has probability at most $\frac{sn/t}{n-s}$ to be from the same group as a previously queried vector). Similar to the above, the probability that the t points chosen for S are not distinct is also at most $1/9$, so by the union bound we have that the probability that two of the queried vectors are identical is at most $2/9$. It follows that there is a difference in the algorithm's view (in terms of queried vectors) between input sets S and S' (permuted) with probability at most $2/9$. Therefore, the probability that the algorithm accepts a (permuted) input S differs from that of a (permuted) input S' by at most $2/9$. However, by our analysis above the probability of accepting S is at least $8/9 \cdot 2/3 = 16/27$, while the probability of accepting S' is at most $1/3$, and we arrive at a contradiction.

The proof extends to any fixed $d > 1$ and $m \geq d+1$, as we now sketch. Let S' consist of many parallel copies of $W = \{0, \vec{e}_1, \ldots, \vec{e}_{d+1}, -\vec{e}_1, \ldots, -\vec{e}_{d+1}\} \subset \mathbb{R}^{d+1}$, where $\vec{e}_i$ is the ith standard unit vector in $\mathbb{R}^{d+1}$. namely $S' = \{\vec{u} + (3i, 3i, \ldots, 3i) \ : \ \vec{u} \in W \ ; \ i = 1, \ldots, \frac{n}{2d+3}\}$. Observe that the l_1 metric on the points of W is just the path metric of a complete bipartite graph $K_{1,2d+1}$, and thus cannot be embedded isometrically into l_1^d, see e.g. [HH78] or [DL97, Prop. 11.1.4]. It follows that S' is $\Omega(1/d)$-far from being isometrically embeddable into l_1^d. Similar to the above, a set S of random points from S' (with many copies) has, with high probability, the same ordering according to each of the coordinates, and thus can be embedded isometrically into l_1^1, and in particular into l_1^d. $\square$

3.2 Almost low-dimensional l_2-metric

We consider next the problem of testing whether a set of n points in Euclidean space can be perturbed by a small distance so that they reside in an affine subspace of low dimension (i.e. they can be embedded isometrically into l_2^d), as follows. This property is relevant in settings where the input may contain some additive error. The next lemma shows that no (two-sided error) testing algorithm for this problem can query a number of vectors that is independent of n. Note that m below is the (high) dimension in which the input data set is given.

Lemma 3.2. *Any algorithm for testing whether a set of n vectors $v_1, \ldots, v_n \in l_2^m$ can be perturbed by a distance of at most δ in order to reside in an affine subspace of dimension d, or ϵ-far from having this property (in the sense that at least an ϵ-fraction of the vectors need to be modified/removed), has to query $\Omega(\min\{\sqrt{n}, \sqrt{m/\log m}\})$ vectors, even for $d = 0$, $\epsilon = 1/2$, and arbitrary $\delta > 0$.*

Proof. Consider first the case where $d = 0$. Let S be a set of n vectors chosen at random from a sphere of radius $\delta_+ = \delta/(1 - \frac{1}{2n})$ (centered at the origin) in $\mathbb{R}^m$. Let S' be a set of n vectors from this sphere, formed by $n/2$ vectors chosen at random from the same sphere together with their $n/2$ antipodal vectors (i.e. for each randomly chosen vector v we take also $-v$).

We claim that with probability at least $5/6$ the set S has the property that its vectors can be perturbed by a distance of at most δ in order to reside in an affine subspace of dimension $d = 0$. Obviously, this property is equivalent to saying that the vectors of S reside in some ball of radius δ. Denote the vectors of S by $u_1, \ldots, u_n$. Then every two vectors u_i, u_j for $i \neq j$ are random vectors from the sphere, and so by the concentration of measure in a sphere in $\mathbb{R}^m$ (see e.g. [MS86, IM99, Fei00]) we have that for all $\kappa > 1$,

$$\Pr\left[|u_i^T u_j| > |u_i| \cdot |u_j| \cdot \sqrt{\kappa/m}\right] \leq e^{-\kappa/4}.$$

For a suitable $\kappa = \Omega(\log n)$ we get $e^{-\kappa/4} \leq \frac{1}{6n^2}$. By a union bound, with probability at least $5/6$ every u_i, u_j satisfy that $|u_i^T u_j| \leq \delta_+^2 \sqrt{\kappa/m}$. Assuming that the latter event happens, consider a ball centered at $\frac{1}{n}\sum_{j=1}^n u_j$ that contains all the vectors $u_1, \ldots, u_n$. For every i we have $\left\|u_i - \frac{1}{n}\sum_j u_j\right\|^2 = (1 - \frac{1}{n})^2\|u_i\|^2 + \frac{1}{n^2}\sum_{j\neq i}\|u_j\|^2 - (1-\frac{1}{n})\frac{1}{n}\sum_{j\neq i} u_i^T u_j \leq \delta_+^2\left(1 - \frac{1}{n} + 2\sqrt{\kappa/m}\right)$. When $m \geq 16\kappa n^2 = \Omega(n^2 \log n)$ we have that $\left\|u_i - \frac{1}{n}\sum_j u_j\right\|^2 \leq \delta_+^2(1 - \frac{1}{2n}) \leq \delta^2$, and then all the vectors $u_1, \ldots, u_n$ reside in a ball of radius δ^2 (centered at $\frac{1}{n}\sum_{j=1}^n u_j$). The claim follows.

Let us now show that S' is always 1/2-far from having the above property. The distance between a vector v and its antipodal vector $-v$ is $2\delta_+$, and thus any ball of radius δ (and thus diameter $2\delta < 2\delta_+$) contains at most one of v and $-v$. It follows that any such ball contains at most half the vectors of S', i.e. at least half the vectors need to be modified/removed in order for them to reside in a ball of radius δ.

Consider an arbitrary (possibly two-sided error) testing algorithm, and assume for contradiction it queries at most $s \leq \sqrt{n/10}$ vectors. Let the algorithm's input be the set of vectors S', permuted at random. The random permutation implies that the s queried vectors are s random vectors from S', and thus the probability (over the randomness in choosing the permutation and in the testing algorithm) of querying a vector v and its antipodal vector $-v$ is at most $\frac{s^2}{n-s} \leq 1/9$ (each of the s queries has probability at most $\frac{s}{n-s}$ to be the antipodal of a previously queried vector). It follows that there is a difference in the algorithm's view (in terms of queried vectors) between input sets S and S' (permuted) with probability at most 1/9. Therefore, the probability that the algorithm accepts a (permuted) input S differs from that of a (permuted) input S' by at most 1/9. However, by our analysis above the probability of accepting S is at least $5/6 \cdot 2/3 = 5/9$, while the probability of accepting S' is at most 1/3, and we arrive at a contradiction.

The proof extends to any fixed $d > 0$; we now sketch the proof for $d = 1$. Let S, S' be sets of (random) vectors in $\mathbb{R}^m$ as above. Let $\hat{S}, \hat{S}'$ be each a set with $2n$ vectors in $\mathbb{R}^{m+1}$ as follows. For every vector $v \in S$ place $(v; 0)$ and $(v; \delta_+)$ in $\hat{S}$, and similarly for $\hat{S}'$. We showed above that with probability at least 5/6, there exists a point $p \in \mathbb{R}^m$ such that all n vectors of S are within distance δ from p. When this events happens, all vectors of $\hat{S}$ are within distance δ from the line that goes through the two points $(p; 0)$ and $(p; \delta_+)$, i.e. $\hat{S}$ has the property that its vectors can be perturbed by a distance of at most δ in order to reside in an affine subspace of dimension $d = 1$. To see that $\hat{S}'$ is 1/4-far from having this property, break $\hat{S}'$ into quadruples $(v; 0), (-v; 0), (v; 2\delta_+), (-v; 2\delta_+)$, and observe that there is no line that intersects all four balls of radius $\delta < \delta_+$ centered at these four points. □

3.3 Distorted low-dimensional l_2-metric

We now consider the problem of testing whether a set of points in Euclidean space has a low-distortion embedding into a low-dimensional Euclidean space. We say that the vectors $v_1, \ldots, v_n \in l_2^m$ can be embedded into l_2^d with distortion at most $\Delta \geq 1$ if there exist vectors $v'_1 \ldots, v'_n \in l_2^d$ such that $1 \leq \frac{\|v_i - v_j\|}{\|v'_i - v'_j\|} \leq \Delta$ for all $1 \leq j < i \leq n$. The next lemma shows that no (two-sided error) testing algorithm for this problem can query a number of vectors that is independent of n.

Lemma 3.3. *Any algorithm for testing whether a set of n vectors in l_2^m can be embedded into l_2^d with distortion at most some fixed $\Delta > 0$, or whether it is ϵ-far from having this property (in the sense that at least an ϵ-fraction of the vectors need to be modified/removed), has to query $\Omega(\sqrt{n/\Delta})$ vectors, even for $d = 1$, $\epsilon = O(1/\Delta)$.*

Proof. Consider first the case $d = 0$. We construct sets S and S' each with n vectors in l_2^3, as follows. Consider a unit circle in l_2^2 and place on it $t = \Omega(\Delta)$ points, so that the distance between every two consecutive points is the same. Now place n/t parallel copies of these discrete circles in l_2^3, so that every two consecutive copies are at distance $r > 0$ from each other, where $r > 0$ is a constant to be determined later. Formally, let

$$S' = \left\{ \begin{pmatrix} \sin(2\pi i/t) \\ \cos(2\pi i/t) \\ j \cdot r \end{pmatrix} : i = 1, \ldots, t \; ; \; j = 1, \ldots, n/t \right\}.$$

To construct S, choose at random one point of S' from each circle and add t copies of it to S.

The set S can be embedded in $\mathbb{R}$ with distortion at most Δ. To see this, map the points of each circle to the center of that circle, i.e., project on the third coordinate. The distance between points from circles $i \neq j$ is at least $|i-j| \cdot r$ and at most $((|i-j| \cdot r)^2 + 4)^{1/2}$, while the distance between their embeddings in $\mathbb{R}$ is $|i-j| \cdot r$. For a suitable choice of r, the distortion in this embedding is at most Δ, as claimed.

We next claim that the set S' is $1/t$-far from having an embedding into $\mathbb{R}$ with distortion at most Δ. Indeed, if less than $1/t$-fraction of the vectors of S' are modified/removed then in at least one circle C, less than $1/t$-fraction of the vectors are modified/removed, i.e. all the t vectors from C remain intact. Embedding into $\mathbb{R}$ the t points on the circle C requires distortion $\Omega(t)$, due to a result of Rabinovich and Raz [RR98]. (In fact, they consider embedding a cycle graph on t vertices into $\mathbb{R}$, but such a cycle can be embedded into a suitable scaling of the circle C with distortion 2π.) The $\Omega(t)$ distortion that is required for the circle C is larger than Δ, for a suitable choice of t, and the claim follows.

Consider an arbitrary (possibly two-sided error) testing algorithm, and assume for contradiction that it queries $s \leq \sqrt{n/10t}$ points. Let the algorithm's input be the set S', permuted at random. The random permutation implies that the s queried points are s random points from S', and thus the probability (over the randomness in choosing the permutation and in the testing algorithm) of querying two points from the same circle is at most $\frac{s^2 t}{n-s} \leq 1/9$ (each of the s queries has probability at most $\frac{st}{n-s}$ to be a point from the same circle as a previously queried point). It follows that there is a difference in the algorithm's view (in terms of queried points) between input sets S and S' (permuted) with probability at most $1/9$. Therefore, the probability that the algorithm accepts a (permuted) input S differs from that of a (permuted) input S' by at most $1/9$. However, by our analysis above the probably of accepting S is at least $2/3$ while the probability of accepting S' is at most $1/3$, and we arrive at a contradiction.

The proof extends to any fixed $d > 1$, by replacing the circles above with a structure that cannot be embedded into l_2^d with a small distortion. For example, for t unit vectors orthogonal to each other in $\mathbb{R}^t$ (or t nearly-orthogonal unit vectors in $\mathbb{R}^{O(\log t)}$) there is a $t^{\Omega(1/d)}$ lower bound on the distortion, due to Euclidean volume considerations. Details omitted from this version of the paper. □

3.4 Distance matrix for low-dimensional l_2-metrics

We next consider testing whether a given matrix is the distance matrix of a low-dimensional Euclidean metric, i.e. whether the given distances can be realized by vectors in l_2^d. We show for this problem a testing algorithm whose complexity slightly improves over a result of Parnas and Ron [PR01]. Our algorithm is similar to that of [PR01]; it chooses a random subset U of points in the metric space, queries the distances between every two points in U (so it queries $O(|U|^2)$ entries of M), and accepts if and only if the metric induced on points of U can be embedded (isometrically) in a Euclidean space of dimension d. Our analysis shows that it suffices to have $|U| = O(d/\epsilon)$, while that of Parnas and Ron [PR01] requires $|U| = O(d \log d/\epsilon)$. The improvement in the sample size follows from a tighter analysis of the underlying random process.

Theorem 3.4. *There is an algorithm for testing whether a real matrix $M_{n\times n}$ is the distance matrix of a d-dimensional Euclidean metric or ϵ-far from it (in the sense that at least ϵ-fraction of the entries of M need to be modified). This algorithm queries the pairwise distances between $O(d/\epsilon)$ points chosen at random.*

Proof. The testing algorithm works as follows. Given M, the algorithm selects at random $O(d/\epsilon)$ points, and accepts if and only if the metric induced by M on these points can be embedded (isometrically) in a Euclidean space of dimension d. As noted in [PR01], one can decide in polynomial time whether such a Euclidean embedding exists (by computing the rank and positive semidefiniteness of a related matrix). Clearly, if M is the distance matrix of a d-dimensional Euclidean metric, then the metric induced by M on any sample U is also a d-dimensional Euclidean metric, and the algorithm always accepts. (It follows that this algorithm has one-sided error.)

Consider a matrix M that is ϵ-far from the distance matrix of any d-dimensional Euclidean metric. We can assume that M is nonnegative and symmetric, since as noted in [PR01, Section 2], these properties are easily testable with complexity $O(1/\epsilon)$. For the sake of analysis, we think of the algorithm as if it starts with a sample U of one point, and iteratively augment the sample with two random points chosen without replacement.

To analyze a single iteration in the algorithm, we use the lemma below, which follows from Lemma 6.1 in the full version of [PR01]. Denote by U_t the set of sampled points that is obtained after $t \geq 0$ iterations (i.e., after t augmentations, so $|U_t| = 1 + 2t$), and let X_t denote the minimum dimension that is required to embed (the metric induced by M on) U_t in a Euclidean metric. If no such dimension exists (i.e., this metric is not Euclidean), let $X_t = \infty$.

Lemma 3.5 (Parnas and Ron [PR01]). *Let M be ϵ-far from the distance matrix of any d-dimensional Euclidean metric. Then* $\Pr[X_{t+1} > X_t \mid X_t \leq d] \geq \epsilon/2$.

By combining Lemmas 1.1 and 3.5 (with $\epsilon' = \epsilon/2$), we have that if M is ϵ-far from the distance matrix of any d-dimensional Euclidean metric, then $\Pr[X_{t^*} \geq d+1] > 2/3$ for $t^* = 16(d+1)/\epsilon$. It follows that with probability at least $2/3$ the testing algorithm rejects U after $t^* = O(d/\epsilon)$ iterations. This completes the proof of Theorem 3.4. □

4 Small norm

In this section we show an algorithm for testing whether a matrix has a small Frobenius norm. It is straightforward that this problem is equivalent to testing whether the l_2-norm of a vector is small. In fact, this equivalence extends to testing the l_p-norm of a vector, for any fixed $p \geq 1$. Our algorithm has a two-sided error. A simple argument shows that any one-sided error testing algorithm for this property must query $\Omega(n)$ entries; details omitted from this version of the paper.

Theorem 4.1. *There is an algorithm for testing whether a vector $v \in \mathbb{R}^n$ has l_1-norm (or l_2-norm) at most b, or whether it is ϵ-far from having this property (in the sense that at least ϵ-fraction of the entries of v need to be modified). This algorithm queries $O(\epsilon^{-3}\log(1/\epsilon))$ randomly chosen entries of v.*

Proof. The testing algorithm works as follows. Given a vector $v \in \mathbb{R}^n$ and $b > 0$, the algorithm queries $s = O(\epsilon^{-3}\log(1/\epsilon))$ entries of v chosen at random (with replacement), discards the $\epsilon s/7$ samples whose absolute values are the largest, and accepts if and only if the average of the absolute values of the remaining $(1-\epsilon/7)s$ samples is at most $(1+\epsilon/2)\frac{b}{n}$. (For testing l_2-norm absolute value is replaced with squared value.) For simplicity, we assume that all terms involving $1/\epsilon$ (such as $\epsilon s/7$) are integers, and that $\epsilon < 1/10$.

For the sake of analyzing our testing algorithm, we can assume that the input vector $v \in \mathbb{R}^n$ has only nonnegative entries (as replacing them with their absolute values does not affect the algorithm or the vector's norm). Similarly, we may assume that the entries of v are in nondecreasing order, i.e. $v_1 \leq v_2 \leq \cdots \leq v_n$. We then partition the n entries of v into $10/\epsilon$ *blocks*, each consisting of

$\epsilon n/10$ entries of v. It follows that for every $1 \le i < j \le 10/\epsilon$, every entry in block i is not larger than every entry in block j. We denote by T_i the sum of the entries in block i. The next lemma characterizes the typical number of samples from a block.

Lemma 4.2. *With probability at least $2/3$, each of the $10/\epsilon$ blocks is sampled by the algorithm between $(1-\epsilon/7)\epsilon s/10$ and $(1+\epsilon/7)\epsilon s/10$ times.*

Proof. Let X_i denote the number of samples from block i. This random variable has a binomial distribution $B(\epsilon/10, s)$, and thus its expectation is $\mathbb{E}[X_i] = \epsilon s/10$. For a suitable $s = \Theta(\epsilon^{-3}\log(1/\epsilon))$ we have by the Chernoff bound (see e.g. [MR95]) that

$$\Pr\Big[X_i \ge (1+\epsilon/7)\mathbb{E}[X_i]\Big] \le e^{-(\epsilon s/10)(\epsilon/7)^2/2} = e^{-\epsilon^3 s/980} \le \epsilon/60,$$

and a similar bound for the probability of the event $X_i \le (1-\epsilon/7)\mathbb{E}[X_i]$. Applying a union bound on the $10/\epsilon$ blocks, the lemma follows. $\square$

Consider first a vector v whose l_1-norm is at most b. From Lemma 4.2 we have that with probability at least $2/3$ every block is sampled between $(1-\epsilon/7)\epsilon s/10$ and $(1+\epsilon/7)\epsilon s/10$ times. Suppose that this event indeed happens. We then have that the samples discarded by the algorithm include all the samples from the highest block $i = 10/\epsilon$ (since $\epsilon s/7 > (1+\epsilon/7)\epsilon s/10$). The number of samples from a block $i < 10/\epsilon$ is at most $(1+\epsilon/7)\epsilon s/10$, and each of them is no larger than the average of the entries in block $i+1$; note that this average is $T_{i+1}/(\epsilon n/10)$. We can thus upper bound the sum of the samples that are not discarded by

$$\begin{aligned}\sum_{i<10/\epsilon} \frac{(1+\epsilon/7)\epsilon s}{10} \cdot \frac{T_{i+1}}{\epsilon n/10} &= \frac{(1+\epsilon/7)s}{n} \sum_{i<10/\epsilon} T_{i+1} \\ &\le \frac{(1+\epsilon/7)sb}{n}.\end{aligned}$$

It follows that the average of the $(1-\epsilon/7)s$ samples that were not discarded is at most

$$\frac{1}{(1-\epsilon/7)s} \cdot \frac{(1+\epsilon/7)sb}{n} < (1+\epsilon/2)\frac{b}{n}.$$

We conclude that with probability at least $2/3$ the aforementioned event happens and the algorithm accepts.

Consider next a vector v that is ϵ-far from having l_1-norm at most b, namely at least ϵn entries of v need to be changed in order to yield a vector of norm at most b. The sum of the $(1-0.9\epsilon)n$ smallest entries of v is larger than b, or otherwise we could zero the largest $0.9\epsilon n$ entries of v (i.e. highest 9 blocks) and obtain a vector of norm at most b, which contradicts our assumption on v. From Lemma 4.2 we have that with probability at least $2/3$ every block is sampled between $(1-\epsilon/7)\epsilon s/10$ and $(1+\epsilon/7)\epsilon s/10$ times. Suppose that this event indeed happens. We then have that all the samples discarded by the algorithm are from the two highest blocks $i = 10/\epsilon$ and $i = 10/\epsilon - 1$ (since $\epsilon s/7 < 2\cdot(1-\epsilon/7)\epsilon s/10$). The number of samples from a block $1 < i < 10/\epsilon - 1$ is at least $(1-\epsilon/7)\epsilon s/10$, and each of them is no smaller than the average of the entries in block $i-1$; note that this average is $T_{i-1}/(\epsilon n/10)$. We can thus lower bound the sum of the samples that are not discarded by

$$\sum_{i=2}^{10/\epsilon-2} \frac{(1-\epsilon/7)\epsilon s}{10} \cdot \frac{T_{i-1}}{\epsilon n/10} = \frac{(1-\epsilon/7)s}{n} \sum_{j=1}^{10/\epsilon-3} T_j.$$

By the nondecreasing order of the entries of v we have that each of $T_{10/\epsilon-8}, \dots, T_{10/\epsilon-3}$ is no smaller than the average of $T_1, \dots, T_{10/\epsilon-9}$. It follows that $\sum_{j=1}^{10/\epsilon-3} T_j \geq \left(1 + \frac{6}{10/\epsilon-9}\right) \sum_{j=1}^{10/\epsilon-9} T_j$. Recall that the sum of the $(1-0.9\epsilon)n$ smallest entries of v is larger than b, i.e., $\sum_{j=1}^{10/\epsilon-9} T_j > b$. Therefore, the average of the $(1-\epsilon/7)s$ samples that are not discarded is larger than

$$\frac{1}{(1-\epsilon/7)s} \cdot \frac{(1-\epsilon/7)s}{n} \cdot \left(1 + \frac{6}{10/\epsilon - 9}\right) \cdot b > (1+\epsilon/2)\frac{b}{n}.$$

We conclude that with probability at least 2/3 the aforementioned event happens and then the algorithm rejects. This proves the correctness of our testing algorithm, which completes the proof of Theorem 4.1. □

Acknowledgements. We thank Ziv Bar-Yossef, Uri Feige, Dick Karp, Guy Kortsarz, Nati Linial, Avner Magen, and Kobbi Nissim for valuable discussions and comments. We also thank the anonymous reviewers for their helpful comments.

References

[AM01] D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations. In *33rd Annual ACM Symposium on the Theory of Computing*, pages 611–618, July 2001.

[BCL98] H. J. Bandelt, V. Chepoi, and M. Laurent. Embedding into rectilinear spaces. *Discrete Comput. Geom.*, 19(4):595–604, 1998.

[CGK+00] M. Charikar, V. Guruswami, R. Kumar, S. Rajagopalan, and A. Sahai. Combinatorial feature selection problems. In *41st Annual IEEE Symposium on Foundations of Computer Science*, pages 631–640, November 2000.

[DDL+90] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[DGL+99] Y. Dodis, O. Goldreich, E. Lehman, S. Raskhodnikova, D. Ron, and A. Samorodnitsky. Improved testing algorithms for monotonicity. In *Randomization, approximation, and combinatorial optimization (RANDOM)*, pages 97–108, Berlin, 1999. Springer.

[DL97] M. Deza and M. Laurent. *Geometry of cuts and metrics*. Springer-Verlag, Berlin, 1997.

[DGGZ02] T. K. Dey, J. Giesen, S. Goswami, and W. Zhao. Shape dimension and approximation from samples. In *13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 772–780, 2002.

[EKK+00] F. Ergün, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Viswanathan. Spot-checkers. *J. Comput. System Sci.*, 60(3):717–751, 2000.

[Fei00] U. Feige. Approximating the bandwidth via volume respecting embeddings. *J. Comput. System Sci.*, 60(3):510–539, 2000.

[FKV98] A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo algorithms for finding low rank approximation. In *39th Annual IEEE Symposium on Foundations of Computer Science*, pages 370–378, November 1998.

[Fis01] E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of the European Association for Theoretical Computer Science. EATCS*, (75):97–126, 2001.

[FN01] E. Fischer and I. Newman. Testing of matrix properties. In *33rd Annual ACM Symposium on the Theory of Computing*, pages 286–295, July 2001.

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

[HH78] F. Hadlock and F. Hoffman. Manhattan trees. *Utilitas Math.*, 13:55–67, 1978.

[IM99] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *30th Annual ACM Symposium on Theory of Computing*, pages 604–613, May 1999.

[Ind01] P. Indyk. Algorithmic applications of low-distortion geometric embeddings. In *42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 10–33, October 2001.

[JL84] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Conn., 1982)*, pages 189–206. Amer. Math. Soc., Providence, RI, 1984.

[Kle98] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677. ACM, 1998.

[Men28] K. Menger. Untersuchungen uber allgemeine metrik. *Mathematische Annalen*, 100:75–163, 1928.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[MS86] V. D. Milman and G. Schechtman. *Asymptotic theory of finite-dimensional normed spaces.* Springer-Verlag, Berlin, 1986.

[Oxl92] J. G. Oxley. *Matroid theory.* Oxford Science Publications. The Clarendon Press Oxford University Press, New York, 1992.

[PR01] M. Parnas and D. Ron. Testing metric properties. In *33rd Annual ACM Symposium on the Theory of Computing*, pages 576–585, July 2001.

[Ron01] D. Ron. Property testing. In P. Pardalos, S. Rajasekaran, J. Reif, and J. Rolim, editors, *Handbook of Randomized Computing.* Kluwer Academic, 2001.

[RR98] Y. Rabinovich and R. Raz. Lower bounds on the distortion of embedding finite metric spaces in graphs. *Discrete Comput. Geom.*, 19(1):79–94, 1998.

[RS96] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.