Polynomial Time Approximation Schemes for Geometric k-Clustering*

Rafail Ostrovsky† Yuval Rabani‡

July 1, 2001

Abstract

The Johnson-Lindenstrauss lemma states that n points in a high dimensional Hilbert space can be embedded with small distortion of the distances into an $O(\log n)$ dimensional space by applying a random linear transformation. We show that similar (though weaker) properties hold for certain random linear transformations over the Hamming cube. We use these transformations to solve NP-hard clustering problems in the cube as well as in geometric settings.

More specifically, we address the following clustering problem. Given n points in a larger set (for example, $\mathbb{R}^d$) endowed with a distance function (for example, L^2 distance), we would like to partition the data set into k disjoint clusters, each with a "cluster center", so as to minimize the sum over all data points of the distance between the point and the center of the cluster containing the point. The problem is provably NP-hard in some high dimensional geometric settings, even for $k = 2$. We give polynomial time approximation schemes for this problem in several settings, including the binary cube $\{0,1\}^d$ with Hamming distance, and $\mathbb{R}^d$ either with L^1 distance, or with L^2 distance, or with the square of L^2 distance. In all these settings, the best previous results were constant factor approximation guarantees.

We note that our problem is similar in flavor to the k-median problem (and the related facility location problem), which has been considered in graph-theoretic and fixed dimensional geometric settings, where it becomes hard when k is part of the input. In contrast, we study the problem when k is fixed, but the dimension is part of the input.

---

*A preliminary version of this paper appeared in the Proceedings of the IEEE Symposium on Foundations of Computer Science, November 2000.

†Telcordia Technologies, MCC-1C357B, 445 South Street, Morristown, New Jersey 07960 6438, USA. e-mail: rafail@research.telcordia.com. URL: http://www.argreenhouse.com/bios/rafail/index.shtml

‡Computer Science Department, Technion — IIT, Haifa 32000, Israel. Part of this work was done while visiting Telcordia Technologies. This work was supported by grant number 386/99-1 of the Israel Science Foundation founded by the Israeli Academy of Sciences and Humanities, and by the Fund for the Promotion of Research at the Technion. Email: rabani@cs.technion.ac.il. URL: http://www.cs.technion.ac.il/~rabani

1 Introduction

Suppose we are given a set X of n data points in $\mathbb{R}^d$ and we wish to find a "good" partition of the points into two non-empty sets X_1 and X_2 (called clusters). There could be many different measures of the quality of the partition. The measure we adopt here is the following: Assign to the set X_i a center point $c_i \in \mathbb{R}^d$, for $i = 1, 2$. Then, sum up the (Euclidean) distances between each data point and the center of the set that contains it. The smaller the sum, the better we deem the partition.

Another way to interpret the problem is the following. If the centers are known, then obviously the best partition is to assign each point to the closest center. Thus, our problem is to find centers $c_1, c_2 \in \mathbb{R}^d$ so as to minimize the quantity

$$\sum_{x \in X} \min\{\|x - c_1\|_2, \|x - c_2\|_2\}.$$

More generally, we also consider variations of this problem with k centers, for a fixed $k > 2$, using other distance measures (such as the L^1 norm, and the square of Euclidean distance), and in other vector spaces (such as the binary cube). We refer to these problems as (geometric) k-clustering. In this paper, we give polynomial time approximation schemes for k-clustering in several high dimensional geometric settings, including the binary cube with Hamming distance, and $\mathbb{R}^d$ with either Euclidean distance, or the square of Euclidean distance, or L^1 distance. As discussed in detail below, previous results provided constant approximation guarantees, or were limited to fixed dimension.

Clustering of data has significant importance in many fields, including operations research, data mining, statistics, computer vision and pattern recognition (see, for example, [37, 10, 38] and references therein). The recent interest in clustering problems can be attributed to applications such as the classification of web pages retrieved by a search engine [40, 33, 30], or the study of gene expression in computational molecular biology [31]. In many applications, the goal is to cluster data into several clusters according to some measure, where the data has many incomparable attributes and thus can be cast as a high dimensional clustering problem [33, 18, 7, 39]. In this paper, we consider the case where the dimension is very large but the number of clusters that we need to produce is relatively small. A typical case is when a large collection of documents must be clustered according to a small number of topics that can be inspected by a person in order to assist further classification and searching. Representation of documents using methods such as latent semantic indexing [20, 16, 19, 9] leads to high dimensional data. Systems like the "scatter/gather"

project [14, 13] or the "Manjara" project [30, 24] (a web meta-search engine that clusters the results of a search according to several topics) require clustering of such data into a relatively small number of clusters.

When k is part of the input, the problem is also known as the k-median problem. In graph-theoretic settings (where the points are placed in a finite metric space which is part of the input), the k-clustering problem (fixed k) trivially has a polynomial time solution: Simply enumerate over all possible choices for the centers. For arbitrary k, in finite metrics, the k-median problem was shown to be APX-hard by Guha and Khuller [25]. A breakthrough result by Charikar, Guha, Shmoys, and Tardos [11] gave a constant factor approximation algorithm, based on a rounding procedure for a natural linear programming relaxation. The constant has been improved by Jain and Vazirani [28], and further by Charikar and Guha [12], using the primal-dual method.

Similarly, in fixed dimension d, the k-clustering problem has a polynomial time solution. To illustrate this for $k = 2$ (in $\mathbb{R}^d$ with Euclidean distances), notice that the clusters must be separated by a hyperplane. In fixed dimension, the number of combinatorially distinct separations is polynomial in n, and we can check each of them efficiently. However, the combinatorial complexity of the problem grows exponentially with the dimension. Indeed, the k-clustering problem was shown to be NP-hard even for $k = 2$ in several cases. Kleinberg, Papadimitriou, and Raghavan [34] show it for the binary cube, and Drineas, Frieze, Kannan, Vempala, and Vinay show it for $\mathbb{R}^d$ with squared Euclidean distances. The NP-hardness of the Euclidean distances case is still open. We note that in fixed dimension, for arbitrary k, Arora, Raghavan, and Rao [6] give a polynomial time approximation scheme, using dynamic programming.

Our measure of the quality of our clustering is by no means the obvious choice. In fact, other measures have been proposed in the literature. The most common alternatives are min-sum clustering, and min-max clustering (or k-center). In min-sum clustering, the quality of the clustering is measured by the sum of intra-cluster distances (so there are no centers associated with the clusters). In min-max clustering, the quality is measured by the maximum distance of a data point to the center of the cluster containing it. None of these measures seem to produce the intuitively "best" clustering on all instances.

For example, Figure 1 at the end of the paper shows two instances of points in the Euclidean plane (requiring a partition into two clusters). Figures 2 and 3 show the results of min-sum and 2-clustering, respectively, on these two instances. Clearly, min-sum is intuitively better on the bottom instance, whereas 2-clustering is intuitively better on the top instance. (Notice that in the case of squared L^2 distances, if C is a cluster, then the cost of C under the min-sum measure is

$\frac{1}{2}\sum_{x,y\in C}\|x-y\|_2^2$ and the cost of C under the 2-clustering measure is $\min_c \sum_{x\in C}\|x-c\|_2^2$. The latter expression is minimized at $c = \frac{1}{n}\sum_{x\in C} x$, and is proportional to the former expression. However, the factor of proportionality is $|C|$, so the two measures do not necessarily produce the same optimal clustering.)

Clustering problems, and in particular min-sum clustering, have been considered recently by several authors. A prevalent technique is sampling: One takes a small (random) sample of the data points, enumerates over all possible partitions of the sample, extends each partition to a partition of the entire data set, and outputs the best solution. Schulman [39] gives a polynomial (linear) time approximation scheme for min-sum clustering in geometric settings (including squared Euclidean distances), provided that the dimension $d = o(\log n/\log\log n)$. His algorithm works in higher dimension too, but the running time degrades to $n^{O(\log\log n)}$. Indyk [26] gives a polynomial time approximation scheme for min-sum clustering in finite metric spaces (when two clusters are needed), based on the polynomial time approximation scheme of de la Vega and Kenyon [17] for metric MAX CUT. Alon and Sudakov [4] give a polynomial time approximation scheme for the maximization version of our problem in the binary cube (i.e., when the objective is to find a partition and centers that maximize the sum over all data points of the overlap between the point and the center of the cluster containing it). Notice that an optimal solution to their problem is also an optimal solution to our problem. However, this is not the case with near-optimal solutions (so their approximately optimal solution could be far from optimal by our measure). All these results use one form or another of sampling.

Sampling is not a common tool in the design of polynomial time approximation schemes. It has been used successfully in the context of dense graphs [5, 23]. In geometric settings (and in general), the ubiquitous method is dynamic programming (see [10]). One example in our context is the k-center algorithms of Agarwal and Procopiuc [1]. They give an $n^{O(k^{1-1/d})}$-time exact algorithm and a polynomial time approximation scheme with running time $O(n\log k) + (k/\epsilon)^{O(k^{1-1/d})}$ for the k-center problem in $\mathbb{R}^d$ with L^p distances, for all p, using dynamic programming. (See also the survey of Agarwal and Sharir [2] for previous and related work.)

A different idea is advocated by Drineas, Frieze, Kannan, Vempala, and Vinay [18]. They give a 2-approximation for k-clustering (fixed k) for the case of squared Euclidean distances, using methods from linear algebra (specifically, singular value decomposition, see also [20, 16, 19, 9, 24] for its uses in information retrieval and clustering). Prior to our work, this was the best result for arbitrary dimension. Notice that there is a trivial 2-approximation algorithm for the case of metric distances (such as Euclidean distances), because if we restrict the centers to be data points, we

lose at most a factor of 2 in the quality of the solution (thanks to the triangle inequality). This immediately implies a 4-approximation in the case of squared Euclidean distances. The advantage of the Drineas et al. method is that the clustering can be computed very quickly using methods for approximating the singular value decomposition (which in turn use sampling).

Our results use neither sampling of the data points, nor dynamic programming, nor the singular value decomposition. For the Hamming cube, we use random linear transformations to reduce the dimension. More specifically, Kushilevitz, Ostrovsky, and Rabani [35] show that a certain random linear transformation into a low dimensional cube can be used to test for a specific Hamming distance. We strengthen their analysis to show that this transformation guarantees low distortion for a *range* of distances, while for distances outside the range it doesn't shrink large distances too much and it doesn't expand small distances too much. We believe that this observation might be of independent interest. We note that in Hilbert space (e.g., $(\mathbb{R}^d, L^2)$) the Johnson-Lindenstrauss Lemma [29] uses a random linear transformation (a projection onto a random subspace) for nearly isometric dimension reduction of finite subsets. This lemma has found recent applications in combinatorics [22], graph algorithms [36], nearest neighbor search [27], and learning mixtures of Gaussians [15]. It does not seem to be useful in our case.

In the low dimensional cube, we can enumerate over the possible center locations and compute a candidate clustering for each possibility. The value of a candidate clustering in the low dimensional cube is not necessarily proportional to its value in the original space. However, we can check the value of each candidate clustering in the original space and output the best solution. This procedure is relatively simple for $k = 2$. For larger k, computing the clustering from the choice of cluster centers in low dimension is more complicated. The location of cluster centers induces for every data point a tournament among the clusters. We assign a data point to an *apex* of its tournament, an idea previously used by Kleinberg [32] in the context of nearest neighbor search. Our other results are derived essentially by reducing the problem to clustering in the Hamming cube. This is not a "black-box" reduction, as we have to modify the cube algorithm to test the candidate clusterings in the original space. Thus, our results imply that in all the settings we consider, in order to get a close to optimal clustering we only need to consider a polynomial number of possible cluster centers. The set of centers to consider can be generated efficiently from a distribution that depends (in a complicated fashion) on the input data points. In other words, we are able to generate a small ϵ-net for the space of partitions into clusters (where proximity is measured by the cost of the partition), thereby reducing significantly the combinatorial complexity of the approximation versions of the (NP-hard) problems we consider.

2 Low Dimensional Embeddings

Recall that a *metric space* is a pair (P, d) where P is a set (whose elements are called *points*), and d is a function $d : P \times P \to \mathbb{R}$ (called a *metric* or a *distance*), such that for every $p_1, p_2, p_3 \in P$ the following hold: (i) $d(p_1, p_2) \geq 0$; (ii) $d(p_1, p_2) = 0 \leftrightarrow p_1 = p_2$; (iii) $d(p_1, p_2) = d(p_2, p_1)$; and (iv) $d(p_1, p_2) + d(p_2, p_3) \geq d(p_1, p_3)$. The last property is called the triangle inequality. If P is a vector space and $\|\cdot\|$ is a norm; then, defining $d(p, q) = \|p - q\|$ we get a metric space, which we denote by $(P, \|\cdot\|)$.

Definition. Let $\mathcal{M} = (P, d)$ and $\mathcal{M}' = (P', d')$ be two metric spaces. Let $X, Y \subseteq P$. A mapping $\varphi : P \to P'$ is (δ, ϵ, ℓ)-*distorted* on (X, Y) [1] iff there exists ℓ' such that for every $x \in X$ and $y \in Y$, the following holds:

1. If $d(x, y) < \epsilon\ell$ then $d'(\varphi(x), \varphi(y)) < (1 + \delta)\epsilon\ell'$.
2. If $d(x, y) > \ell/\sqrt{\epsilon}$ then $d'(\varphi(x), \varphi(y)) > (1 - \delta)\ell'/\sqrt{\epsilon}$.
3. If $\epsilon\ell \leq d(x, y) \leq \ell/\sqrt{\epsilon}$ then
   $(1 - \delta)\ell'/\ell \leq d'(\varphi(x), \varphi(y))/d(x, y) \leq (1 + \delta)\ell'/\ell$.

Intuitively, a (δ, ϵ, ℓ)-distorted mapping approximately preserves distances close to ℓ, and furthermore it doesn't shrink too much large distances and doesn't expand too much small distances.

The following lemma is central to the analysis of our algorithms:

Lemma 1. Let $\delta, \epsilon, \ell > 0$, with $\epsilon/(1 - \epsilon) \leq (1 - \delta)/(1 + \delta)$. Let $\mathcal{M} = (P, d)$ and $\mathcal{M}' = (P', d')$ be two metric spaces. Let $x, y, z \in P$, with $\ell \leq d(y, z) \leq 2\ell$. Let φ be (δ, ϵ, ℓ)-distorted on $(\{x\}, \{y, z\})$. If $d'(\varphi(x), \varphi(y)) \leq d'(\varphi(x), \varphi(z))$; then, $d(x, y) \leq (1 + \zeta)d(x, z)$, where $\zeta = \max\{(2\epsilon - 1)/(1 - \epsilon), 2\sqrt{\epsilon}, 2\delta/(1 - \delta)\}$.

Proof. We consider four cases:
Case 1: If $d(x, y) < \epsilon\ell$, then by the triangle inequality, $d(x, z) > (1 - \epsilon)\ell$. Therefore, the claim holds in this case.
Case 2: If $d(x, z) < \epsilon\ell$, then $d(x, y) > (1 - \epsilon)\ell$. However, because φ is (δ, ϵ, ℓ)-distorted on $(\{x\}, \{y, z\})$, then for some $\ell' > 0$,

$$d'(\varphi(x), \varphi(z)) \quad < \quad (1 + \delta)\epsilon\ell'$$

[1] If $X = Y$ we simply say that φ is (δ, ϵ, ℓ)-distorted on X.

$$\begin{aligned} &\leq (1-\delta)(1-\epsilon)\ell' \\ &< d'(\varphi(x),\varphi(y)), \end{aligned}$$

in contradiction to the assumption of the lemma.

Case 3: If $d(x,y) > \ell/\sqrt{\epsilon}$, then by the triangle inequality $d(x,z) \leq d(x,y) + 2\ell < (1+2\sqrt{\epsilon})d(x,y)$, so the claim holds in this case too.

Case 4: Otherwise, $\epsilon\ell \leq d(x,y) \leq \ell/\sqrt{\epsilon}$, $\epsilon\ell \leq d(x,z)$, and we may assume that $d(x,z) \leq \ell/\sqrt{\epsilon}$ (otherwise the lemma is clearly true). Thus we have

$$\begin{aligned} d(x,y) &\leq \frac{\ell \cdot d'(\varphi(x),\varphi(y))}{(1-\delta)\ell'} \\ &\leq \frac{\ell \cdot d'(\varphi(x),\varphi(z))}{(1-\delta)\ell'} \\ &\leq \frac{1+\delta}{1-\delta} d(x,z) \qquad \blacksquare \end{aligned}$$

Notation. The field with two elements is denoted $\mathbb{Z}_2$. A d-dimensional vector space over $\mathbb{Z}_2$ is Denoted $\mathbb{Z}_2^d$. The d-dimensional Hamming distance (i.e., the L^1 norm in $\mathbb{Z}_2^d$) is denoted H_d. The Hamming cube Q^d is the metric space $(\mathbb{Z}_2^d, H_d)$.

The Hamming cube. Consider a probability distribution $\mathcal{A}_{d,d'}(p)$ on $d' \times d$ matrices over $\mathbb{Z}_2$ (i.e., linear transformations from $\mathbb{Z}_2^d$ into $\mathbb{Z}_2^{d'}$), where the entries are independent, identically distributed random 0/1 variables with $\Pr[1] = p$. The following lemma is an extension of a lemma in [35].

Lemma 2. For every $\gamma > 0$ there exists $\lambda > 0$ such that for every $\epsilon > 0$, and for every positive integers n, d, and ℓ, with $\ell \in [1,d]$, the following holds: Let $X \subseteq \mathbb{Z}_2^d$, with $|X| = n$. Let $d' = \lambda \ln n/\epsilon$, and let A be a random matrix drawn from $\mathcal{A}_{d,d'}(\epsilon/\ell)$. Then the linear mapping $x \mapsto Ax$ is $(\sqrt{\epsilon}, \epsilon, \ell)$-distorted on X (with respect to the Hamming distance in both spaces) with probability at least $1 - n^{-\gamma}$.

Proof. Let $x, y \in X$. Consider a probability distribution $\mathcal{D}$ over vectors $r \in \mathbb{Z}_2^d$, where the entries of r are independent, identically distributed, random 0/1 variables with $\Pr[1] = \epsilon/\ell$. We estimate the probability of the event $r \cdot (x-y) \neq 0$, denoted in what follows as E.

We first notice that the probability of E is monotonically increasing in $H_d(x,y)$ (assuming $\epsilon/\ell \leq \frac{1}{2}$). To see this, pick r by selecting coordinates independently with probability $2\epsilon/\ell$ each, and then setting each selected coordinate independently as 1 with probability $\frac{1}{2}$ (with all the remaining coordinates being set to 0). The probability of E is precisely half the probability that in the

first step we select at least one coordinate where x and y differ. The latter probability is clearly monotonically increasing in $H_d(x,y)$.

Let S be the set of coordinates where x and y differ (so $|S| = H_d(x,y)$), and let (the random variable) $X = |\{i \in S; r_i = 1\}|$. Then,

$$\Pr[E] = \Pr[X \equiv 1 \bmod 2] \geq \Pr[X = 1] \geq \tag{1}$$
$$\geq |S| \cdot \frac{\epsilon}{\ell} - \binom{|S|}{2} \cdot \left(\frac{\epsilon}{\ell}\right)^2 \geq \left(1 - \frac{\epsilon|S|}{2\ell}\right) \cdot \frac{\epsilon|S|}{\ell},$$

where the second inequality follows from the Bonferroni Inequalities. On the other hand,

$$\Pr[E] \leq \Pr[X \geq 1] = 1 - \left(1 - \frac{\epsilon}{\ell}\right)^{|S|}. \tag{2}$$

If $H_d(x,y)$ is in the interval $[\epsilon\ell, \ell/\sqrt{\epsilon}]$, then (1) is at least $(1-\sqrt{\epsilon}/2)\epsilon H_d(x,y)/\ell$, and (2) is at most $\epsilon H_d(x,y)/\ell$. By the monotonicity of $\Pr[E]$, if $H_d(x,y) > \ell/\sqrt{\epsilon}$, then $\Pr[E] > (1-\sqrt{\epsilon}/2)\sqrt{\epsilon}$, and if $H_d(x,y) < \epsilon\ell$, then $\Pr[E] < \epsilon^2$.

Now, picking a random matrix A from $\mathcal{A}_{d,d'}(\epsilon/\ell)$ amounts to picking the rows of A as d' independent samples from $\mathcal{D}$. The value of $H_{d'}(Ax, Ay)$ is precisely the number of times the event E happens for the d' samples. The expectation is $d'\Pr[E]$. By standard Chernoff bounds (see [3, Appendix A]), The probability that we deviate from the expectation by more than $\sqrt{\epsilon}d'\Pr[E]/2$ (either way) is at most $2e^{-\epsilon d'/9} < n^{-2-\gamma}$, assuming λ is sufficiently large. Summing up this probability over all $\binom{n}{2}$ pairs $x, y \in X$ completes the proof. ■

Other metric spaces. For instances in $(\mathbb{R}^d, L^1)$ and $(\mathbb{R}^d, L^2)$, we use embeddings into the Hamming cube. Let $B_p(x,\ell)$ denote the L^p ball of radius ℓ around $x \in \mathbb{R}^d$. The following lemma was proven in [35]:

Lemma 3 (Kushilevitz, Ostrovsky, and Rabani). Let $p \in \{1,2\}$, and consider the metric space $(\mathbb{R}^d, L^p)$. For every $\beta, \epsilon, \ell > 0$, and for every positive integer d, there exist $\delta = \delta(\epsilon) > 0$ and a positive integer $d' = \text{poly}(d, \delta^{-1}, \log\beta^{-1})$, such that $\delta \to 0$ as $\epsilon \to 0$, and such that for every $x \in \mathbb{R}^d$ there is a distribution $\Phi = \Phi(x, \ell, \delta, \beta)$ over mappings $\varphi : \mathbb{R}^d \to \mathbb{Z}_2^{d'}$ with the following properties:

1. Every mapping ϕ in the distribution Φ is defined by $\text{poly}(d')$ rationals; given these rationals, for every $y \in \mathbb{R}^d$, $\phi(y)$ can be computed using $\text{poly}(d')$ arithmetic operations; and, it is possible to generate the rationals defining a random sample ϕ of Φ in $\text{poly}(d')$ time.

2. If ϕ is drawn from Φ, then with probability at least $1-\beta$, ϕ is (δ, ϵ, ℓ)-distorted on $(B_p(x,\ell), \mathbb{R}^d)$ (with respect to the L^p distance in $\mathbb{R}^d$, and the Hamming distance in $\mathbb{Z}_2^{d'}$).

For our approximation schemes we need the following stronger claim:

Lemma 4. Let $X = \{x^1, x^2, \ldots, x^n\} \subseteq \mathbb{R}^d$. For every β, ϵ, ℓ, d, there exist $\delta = \delta(\epsilon) > 0$ and a positive integer $d' = \text{poly}(n, d, \delta^{-1}, \log \beta^{-1})$, such that there is a distribution $\Phi = \Phi(X, \ell, \delta, \beta)$ over mappings φ with the following properties:

1. Every mapping ϕ in the distribution Φ is defined by $\text{poly}(d')$ rationals; given these vectors, for every $x \in \mathbb{R}^d$, $\phi(x)$ can be computed using $\text{poly}(d')$ arithmetic operations; and, it is possible to generate the rationals defining a random sample ϕ of Φ in $\text{poly}(d')$ time.

2. If ϕ is drawn from Φ, then with probability at least $1 - \beta$, ϕ is (δ, ϵ, ℓ)-distorted on $(\cup_{i=1}^n B_p(x^i, \ell/\sqrt{\epsilon}), \mathbb{R}^d)$.

The proof of this lemma follows closely the construction from the proof of Lemma 3 in [35]. We do not include it here.

3 Clustering in the Hypercube

Observe that for any given cluster C (a subset of the data set), the best cluster center c can be computed easily. Indeed, for $i = 1, 2, \ldots, d$, $c_i = \text{majority}\{x_i; x \in C\}$.

Two clusters. Our basic algorithm is a polynomial time approximation scheme for instances in Q^d and for $k = 2$. The approximation scheme for $k > 2$ uses similar ideas in a more complicated way. The algorithms for other metrics use variations of these schemes as subroutines.

Let $X \subseteq \mathbb{Z}_2^d$ denote the input set of points. Our algorithm for $k = 2$ proceeds as follows. We guess the distance ℓ between the two centers (by enumerating over all d possible values). We then project the data points into a dimension $d' = O(\log n)$ cube. In the smaller cube, we enumerate over all $2^{2d'}$ possible locations for the projections of the optimal solution cluster centers. Each choice induces a partition of the data set into two subsets. Each subset is associated with a cluster center projection, and contains all the points whose projections are closer to this center's projection than to the other center's projection, ties broken arbitrarily. We check each possible partition in the original space, by computing the best cluster center for each subset, and summing up the distances from the points to their assigned centers. Finally, we output the best partition, over all guesses of ℓ and over all guesses of the cluster centers projections. More formally, the algorithm is given by the following pseudo-code:

HAMMING2CLUSTERING$_\epsilon(X)$

$d' \leftarrow \lambda \ln(n+2)/\epsilon$;
for $\ell = 1, 2, \ldots, d$ **do**
    Draw a random A^ℓ from $\mathcal{A}_{d,d'}(\epsilon/\ell)$;
    for all choices of $\tilde{c}^1, \tilde{c}^2 \in \mathbb{Z}_2^{d'}$ **do**
        $C_1 \leftarrow \{x \in X;\ H_{d'}(A^\ell x, \tilde{c}^1) \leq H_{d'}(A^\ell x, \tilde{c}^2)\}$;
        $C_2 \leftarrow X \setminus C_1$;
        cost $\leftarrow$ HAMMINGCOST(C_1) + HAMMINGCOST(C_2);
Output the partition (C_1, C_2) with the smallest cost.

HAMMINGCOST(C)

$c \leftarrow (\text{majority}\{x_i; x \in C\})_{i=1}^d$;
Return $\sum_{x \in C} H_d(x, c)$.

For simplicity, we left out the initialization and updating of the auxiliary variables needed to find the minimum cost and to store the solution in the main procedure. Clearly, for fixed ϵ, the running time of the algorithm is polynomial in n and in d. The following theorem states the performance guarantee for this algorithm.

Theorem 5. For every $\gamma > 0$, there exists $\lambda > 0$ such that for every $\frac{1}{4} \geq \epsilon > 0$, the above algorithm finds a solution whose value is within a factor of $1 + 4\sqrt{\epsilon}$ of the optimum, with probability at least $1 - n^{-\gamma}$.

Proof. Put $\lambda = \lambda(\gamma)$ to be the constant stipulated in Lemma 2. It is sufficient to show that one of the guesses that the algorithm uses produces a solution with value within a factor of $1 + 4\sqrt{\epsilon}$ of the optimum, with probability at least $1 - n^{-\gamma}$. To see this, consider the solutions produced by the algorithm for ℓ such that $\ell = H_d(c^1, c^2)$. By Lemma 2, with probability at least $1 - n^{-\gamma}$, the mapping $x \mapsto A^\ell x$ is $(\sqrt{\epsilon}, \epsilon, \ell)$-distorted on $X \cup \{c^1, c^2\}$. So, from now on we assume that this event happens. Of course, we don't know where the images of c^1 and c^2 are, but one of the guesses that the algorithm enumerates over is correct. So, consider the solution given by the algorithm for A^ℓ and the correct guess of the images $\tilde{c}^1 = A^\ell c^1$ and $\tilde{c}^2 = A^\ell c^2$. Denote by $\widehat{C}_1$, $\widehat{C}_2$, the clusters

computed by the algorithm, and let $\hat{c}^1$, $\hat{c}^2$ be their centers, respectively. (A point $x \in X$ is placed in $\widehat{C}_1$ iff $H_{d'}(A^\ell x, \tilde{c}^1) \leq H_{d'}(A^\ell x, \tilde{c}^2)$, and otherwise it is placed in $\widehat{C}_2$.) Using Lemma 1,

$$\begin{aligned}
&\sum_{x \in \widehat{C}_1} H_d(x, \hat{c}^1) + \sum_{x \in \widehat{C}_2} H_d(x, \hat{c}^2) \\
&\leq \sum_{x \in \widehat{C}_1} H_d(x, c^1) + \sum_{x \in \widehat{C}_2} H_d(x, c^2) \\
&\leq (1 + 4\sqrt{\epsilon}) \sum_{x \in X} \min\{H_d(x, c^1), H_d(x, c^2)\}
\end{aligned}$$

. ■

More than two clusters. We now consider partitioning the data set into k clusters, for an arbitrary (fixed) $k > 2$. The algorithm is similar to the case of $k = 2$. We enumerate over the possible distances between centers ($\binom{k}{2}$ values this time). However, for a given guess, the assignment of data points to clusters is more complicated. Recall that a *tournament* is a directed graph where every pair of distinct nodes is connected by an arc (in one of the two directions). An *apex* of a tournament is a node of maximum out degree. Every apex has the property that there is a path of length at most two from it to any other node in the tournament. The algorithm for $k > 2$ proceeds as follows. After guessing the distances ℓ_{st} between every pair s, t of cluster centers in the optimal solution, we project the data points into $\binom{k}{2}$ cubes, each of dimension $O(\log n)$. Each projection is set to check a particular pair of cluster centers. In each $O(\log n)$-dimensional cube, we enumerate over all the possible locations for the projections of the cluster centers. Given a such a choice, for every data point and for every pair of cluster centers, we decide whether the data point is closer to one center or the other according to the situation with the projected points. This induces, for every data point, a tournament among the cluster centers. We assign each data point to an apex of its tournament. The assignment of all data points induces a partition of the data set into k subsets. We check this partition in the original space, as we did for $k = 2$. Finally, we output the best partition among all the choices for inter-cluster center distances, and cluster centers projections. The following pseudo-code gives a more formal description of the algorithm:

HammingClustering$_\epsilon(X)$

$d' \leftarrow \lambda \ln(n + k)/\epsilon$;
$\forall \ell \in \{1, 2, \ldots, d\}$, draw a random A^ℓ from $\mathcal{A}_{d,d'}(\epsilon/\ell)$;
for all $(\ell_{st})_{1 \leq s < t \leq k} \in \{1, 2, \ldots, d\}^{\binom{k}{2}}$ **do**
    for all $(\tilde{c}^{ij})_{i \neq j = 1}^{k} \in (\mathbb{Z}_2^{d'})^{k(k-1)}$ **do**

$(C_1, C_2, \ldots, C_k) \leftarrow (\emptyset, \emptyset, \ldots, \emptyset)$;
for $x \in X$ **do**
    Compute a tournament T over node set $\{1, 2, \ldots, k\}$:
        for $1 \leq i < j \leq k$, ij is an edge of T
        iff $H_{d'}(A^{\ell_{ij}} x, \tilde{c}^{ij}) \leq H_{d'}(A^{\ell_{ij}} x, \tilde{c}^{ji})$,
        and otherwise ji is an edge of T;
    Find an apex i of T;
    $C_i \leftarrow C_i \cup \{x\}$;
cost $\leftarrow \sum_{i=1}^{k}$ HammingCost(C_i);
Output the partition $(C_1, C_2, \ldots, C_k)$ with the smallest cost.

Clearly, for fixed ϵ and k this algorithm runs in time polynomial in n and d. Its performance guarantee is given by the following theorem.

Theorem 6. For every $\gamma > 0$, there exists $\lambda > 0$ such that for every $\frac{1}{4} \geq \epsilon > 0$, the above algorithm finds a solution whose value is within a factor of $(1 + 4\sqrt{\epsilon})^2$ of the optimum, with probability at least $1 - n^{-\gamma}$.

Proof. Let $c^1, c^2, \ldots, c^k \in \mathbb{Z}_2^d$ denote the centers of the clusters in the optimum solution. Take λ large enough, so that with probability at least $1 - n^{-\gamma}$, for every integer $\ell \in \{1, \ldots, d\}$, the matrix A^ℓ is $(\sqrt{\epsilon}, \epsilon, \ell)$-distorted on $X \cup \{c^1, c^2, \ldots, c^k\}$.[2]

Consider the iteration of HammingClustering$_\epsilon$ where $\ell_{ij} = H_d(c^i, c^j)$, for all $1 \leq i < j \leq k$, and $\tilde{c}^{ij} = A^{\ell_{ij}} c^i$ for all $1 \leq i \neq j \leq k$. Let $x \in X$, and let c^i be the center of the cluster containing x in the optimum solution. Suppose x is clustered in C_t by the algorithm. Then, there is a path of length at most 2 from c^t to c^i in the tournament for x. Let c^j be the middle point in this path (if the path has length 0, then $i = j = t$, and if the path has length 1, then $i = j$). Applying Lemma 1 at most twice (for $A^{\ell_{tj}}$ and for $A^{\ell_{ji}}$), we get:

$$\begin{aligned} H_d(x, c^t) &\leq (1 + 4\sqrt{\epsilon}) H_d(x, c^j) \\ &\leq (1 + 4\sqrt{\epsilon})^2 H_d(x, c^i). \end{aligned}$$

The rest of the proof follows that of Theorem 5. ■

[2]The case $d \gg n$ has to be handled with care.

4 Other Metrics

Consider a metric space $\mathcal{M} = (P, d)$ and an input set of n points $X \subseteq P$. Let $B(x, \ell)$ denote the ball of radius ℓ around x in $\mathcal{M}$. We present here polynomial time approximation schemes for k-clustering for several choices of $\mathcal{M}$. To illustrate the algorithms, we first present the case $k = 2$. The generalization to arbitrary fixed k is straightforward, and we explain it afterwards.

Two clusters. The main idea of our approximation schemes is the following generic approach: Guess the distance ℓ between the cluster centers. Let $\delta = \delta(\epsilon)$ be such that $\delta \to 0$ as $\epsilon \to 0$. Map the input data set into $\mathbb{Z}_2^m$ (where $m = \text{poly}(n, d, \epsilon^{-1})$) using a mapping φ which is (δ, ϵ, ℓ)-distorted on $(\cup_{x \in X} B(x, \ell/\sqrt{\epsilon}), P)$ (with respect to Hamming distance in the target space). Now run the procedure HAMMING2CLUSTERING$_\epsilon$ on $\varphi(X)$ with the following change: Use, instead of HAMMINGCOST, a procedure OURSPACECOST that computes the cost of a cluster in $\mathcal{M}$ rather than in Q^m.

For this approach to work, three conditions are required. Firstly, the set of possible guesses for the distance between the two cluster centers has to have polynomial size. Secondly, the mapping φ must exist and must be computable in polynomial time. Thirdly, it must be possible to compute the cost of a cluster in $\mathcal{M}$ in polynomial time. We establish these conditions for a few interesting cases. Before we discuss these conditions, we analyze the performance guarantee of the above approximation scheme. Let C_1 and C_2 be the partition of X into clusters in the optimum solution, and let c^1 and c^2 be the centers of these clusters, respectively.

Theorem 7. Let ϵ be sufficiently small so that $\delta, \epsilon \leq \frac{1}{16}$. There exists $\zeta = \zeta(\epsilon) > 0$ such that $\zeta \to 0$ as $\epsilon \to 0$, and such that the following holds. For every $\gamma > 0$ there exists $\lambda > 0$ such that if $\ell \leq d(c^1, c^2) \leq 2\ell$; then, the above algorithm produces a clustering whose cost is within a factor of $1 + \zeta$ of the optimum, with probability at least $1 - n^{-\gamma}$.

Proof. Put $\lambda = \lambda(\gamma)$ to be the constant stipulated in Lemma 2. Let ℓ' be the scale for which φ is (δ, ϵ, ℓ)-distorted on $(\cup_{x \in X} B(x, \ell/\sqrt{\epsilon}), P)$. Consider the execution of the modified HAMMING2CLUSTERING on $\varphi(X) \subseteq \mathbb{Z}_2^m$. Let $\widehat{C}_1$ and $\widehat{C}_2$ be the partition of X into clusters which is computed by the algorithm in the iteration using ℓ' and the centers $\tilde{c}^1 = A^{\ell'} \varphi(c^1)$ and $\tilde{c}^2 = A^{\ell'} \varphi(c^2)$. Let $\hat{c}^1, \hat{c}^2 \in P$ be the centers of $\widehat{C}_1, \widehat{C}_2$, respectively. For $i = 1, 2$, let $A_i = \{x \in \widehat{C}_i;\ d(x, c^i) > \ell/\sqrt{\epsilon}\}$, and let $B_i = \widehat{C}_i \setminus A_i$. Now,

$$\sum_{x \in \widehat{C}_1} d(x, \hat{c}^1) + \sum_{x \in \widehat{C}_2} d(x, \hat{c}^2)$$

$$
\begin{aligned}
&\leq \sum_{x\in\widehat{C}_1} d(x,c^1) + \sum_{x\in\widehat{C}_2} d(x,c^2) \\
&= \sum_{x\in A_1} d(x,c^1) + \sum_{x\in A_2} d(x,c^2) + \sum_{x\in B_1} d(x,c^1) + \sum_{x\in B_2} d(x,c^2) \\
&\leq \sum_{x\in A_1\cup A_2} \left(1+\frac{2\sqrt{\epsilon}}{1-2\sqrt{\epsilon}}\right) \min\{d(x,c^1),d(x,c^2)\} + \sum_{x\in B_1} d(x,c^1) + \sum_{x\in B_2} d(x,c^2) \\
&\leq \sum_{x\in A_1\cup A_2} \left(1+\frac{2\sqrt{\epsilon}}{1-2\sqrt{\epsilon}}\right) \min\{d(x,c^1),d(x,c^2)\} + \\
&\quad + \sum_{x\in B_1\cup B_2} \left(1+\max\left\{\frac{2\sqrt{\epsilon}}{1-\delta}, 2(\delta+\sqrt{\epsilon}+\delta\sqrt{\epsilon})(1-\delta-\sqrt{\epsilon}-\delta\sqrt{\epsilon})\right\}\right) \min\{d(x,c^1),d(x,c^2)\},
\end{aligned}
$$

where the last inequality follows from Lemma 1, using the fact that $A^{\ell'}\circ\varphi$ is (δ',ϵ',ℓ)-distorted on $X\cup\{c^1,c^2\}$, for $\delta'=\delta+\sqrt{\epsilon}+\delta\sqrt{\epsilon}$ and $\epsilon'=\max\{(1+\delta)\epsilon,\epsilon/(1-\delta)^2\}$. ■

We now turn our attention to the three conditions required for the success of our approach. The first condition is easy to guarantee in every metric space. Indeed, to apply Theorem 7, all we need is to guess $d(c^1,c^2)$ within a factor of 2. Thus, the number of values we have to check depends only on the range of possible values. To restrict that range, we use

Lemma 8. Let $d_{\max}$ be the maximum distance between any pair of points in X. If $d(c^1,c^2)\leq\frac{\epsilon d_{\max}}{n}$, then any partition has a cost within a factor of $1+\frac{2\epsilon n}{n-2\epsilon}$ of the optimum solution.

Proof. Let $x,y\in X$ be two points at distance $d(x,y)=d_{\max}$. By the triangle inequality, $d(x,c^1)+d(y,c^1)\geq d_{\max}$. Thus, without loss of generality $d(x,c^1)\geq d_{\max}/2$. However, by the triangle inequality, $d(x,c^2)\geq(\frac{1}{2}-\frac{\epsilon}{n})d_{\max}$, so the cost of the optimum solution is at least $(\frac{1}{2}-\frac{\epsilon}{n})d_{\max}$. Consider any partition of X into two clusters A_1 and A_2 with cluster centers a^1 and a^2, respectively. Using again the triangle inequality,

$$
\begin{aligned}
&\sum_{x\in A_1} d(x,a^1) + \sum_{x\in A_2} d(x,a^2) \\
&\leq \sum_{x\in A_1} d(x,c^1) + \sum_{x\in A_2} d(x,c^2) \\
&\leq \sum_{x\in C_1} (d(x,c^1)+\epsilon d_{\max}/n) + \sum_{x\in C_2} (d(x,c^2)+\epsilon d_{\max}/n) \\
&\leq \sum_{x\in C_1} d(x,c^1) + \sum_{x\in C_2} d(x,c^2) + \epsilon d_{\max} \\
&\leq \left(1+\frac{2\epsilon n}{n-2\epsilon}\right)\left(\sum_{x\in C_1} d(x,c^1) + \sum_{x\in C_2} d(x,c^2)\right). \qquad \blacksquare
\end{aligned}
$$

As for the second condition, Lemma 4 guarantees that we can compute φ for instances in $(\mathbb{R}^d, L^1)$ and in $(\mathbb{R}^d, L^2)$. In both cases, the third condition holds as well. For the L^1 norm, implementing OURSPACECOST is easy: On input set C, compute the best center c by $c_i = \text{median}\{x_i;\ x \in C\}$, then output $\sum_{x\in C} \|x - c\|_1$. For the L^2 norm, the problem is significantly harder. Finding the best center c amounts to minimizing a convex function, and it can be approximated with arbitrary precision in polynomial time. Thus we get the following corollary of Theorem 7:

Corollary 9. There are polynomial time approximation schemes for 2-clustering in $(\mathbb{R}^d, L^1)$ and in $(\mathbb{R}^d, L^2)$. ■

Finally, we deal with the case of clustering points in $\mathbb{R}^d$ with distances measured by the square of the L^2 norm. The problem with this case is that the distance function does not induce a metric, so our analysis so far does not hold. We solve this problem by using the algorithm for L^2 distances, but using a different OURSPACECOST procedure that computes cluster costs under L^2 squared distances. Such a procedure is easy to implement. On input set C, the best center c is given by $c_i = \text{average}\{x_i;\ x \in C\}$. The procedure then returns the value $\sum_{x\in C} \|x - c\|_2^2$. We get

Theorem 10. The above algorithm is a polynomial time approximation scheme for 2-clustering in $\mathbb{R}^d$, with distances measured by the square of L^2 distance.

Proof. The proof follows closely the proof of Theorem 7. We use the same notation as in that proof. Our algorithm enumerates over a polynomial number of guesses ℓ for $\|c^1 - c^2\|_2$. By the discussion above, one of these guesses satisfies $\ell \le \|c^1 - c^2\|_2 < 2\ell$. Now, for this ℓ, the algorithm uses a mapping $\varphi : \mathbb{R}^d \to \mathbb{Z}_2^m$. Let ℓ' be the scale for which φ is (δ, ϵ, ℓ)-distorted on $(\cup_{x\in X} B(x, \ell/\sqrt{\epsilon}), P)$. Consider the execution of the modified HAMMING2CLUSTERING on $\varphi(X) \subseteq \mathbb{Z}_2^m$. Let $\widehat{C}_1$ and $\widehat{C}_2$ be the partition of X into clusters which is computed by the algorithm in the iteration using ℓ' and the centers $\tilde{c}^1 = A^{\ell'}\varphi(c^1)$ and $\tilde{c}^2 = A^{\ell'}\varphi(c^2)$. Let $\hat{c}^1, \hat{c}^2 \in P$ be the centers of $\widehat{C}_1, \widehat{C}_2$, respectively. For $i = 1, 2$, let $A_i = \{x \in \widehat{C}_i;\ \|x - c^i\|_2 > \ell/\sqrt{\epsilon}\}$, and let $B_i = \widehat{C}_i \setminus A_i$. Now,

$$
\begin{aligned}
& \sum_{x\in\widehat{C}_1} \|x - \hat{c}^1\|_2^2 + \sum_{x\in\widehat{C}_2} \|x - \hat{c}^2\|_2^2 \\
\le\ & \sum_{x\in\widehat{C}_1} \|x - c^1\|_2^2 + \sum_{x\in\widehat{C}_2} \|x - c^2\|_2^2 \\
=\ & \sum_{x\in A_1} \|x - c^1\|_2^2 + \sum_{x\in A_2} \|x - c^2\|_2^2 + \sum_{x\in B_1} \|x - c^1\|_2^2 + \sum_{x\in B_2} \|x - c^2\|_2^2 \\
\le\ & \sum_{x\in A_1\cup A_2} \left(1 + \frac{2\sqrt{\epsilon}}{1 - 2\sqrt{\epsilon}}\right)^2 \min\{\|x - c^1\|_2^2, \|x - c^2\|_2^2)\} + \sum_{x\in B_1} \|x - c^1\|_2^2 + \sum_{x\in B_2} \|x - c^2\|_2^2
\end{aligned}
$$

$$\leq \sum_{x \in A_1 \cup A_2} \left(1 + \frac{2\sqrt{\epsilon}}{1-2\sqrt{\epsilon}}\right)^2 \min\{\|x-c^1\|_2^2, \|x-c^2\|_2^2)\} +$$
$$+ \sum_{x \in B_1 \cup B_2} \left(1 + \max\left\{\frac{2\sqrt{\epsilon}}{1-\delta}, 2(\delta+\sqrt{\epsilon}+\delta\sqrt{\epsilon})(1-\delta-\sqrt{\epsilon}-\delta\sqrt{\epsilon})\right\}\right)^2 \min\{\|x-c^1\|_2^2, \|x-c^2\|_2^2\}.$$

■

More than two clusters. A similar approach works for more than two clusters: Guess the $\binom{k}{2}$ distances between centers. For a matrix of guesses ℓ, use mappings ϕ_{st} of the data set into $\mathbb{Z}_2^m$, which are $(\delta, \epsilon, \ell_{st})$-distorted on $(\cup_{x \in X} B(x, \ell_{st}/\sqrt{\epsilon}), P)$, for all $1 \leq s < t \leq k$. As in HammingClustering$_\epsilon$, for each data point $x \in X$ compute a tournament over the centers by guessing, for every pair of centers c^s, c^t, the images of c^s, c^t under ϕ_{st}, and then checking to which image $\phi_{st}(x)$ is closer. Assign x to an apex of its tournament. Finally, compute the cost of every solution using OurSpaceCost, and output the best solution.

Clearly, the discussion regarding $k = 2$ shows that the required mappings ϕ_{st} exist and are computable in polynomial time in all the settings considered there. Moreover, OurSpaceCost can be implemented efficiently in all those settings. Thus, the only issue that requires further discussion is the number of guesses for inter-center distances that are needed. As in the $k = 2$ case, we only need to guess the distances within a factor of 2, so the question is the bounds on the range of possible values. The following lemma resolves this issue.

Lemma 11. If distances between cluster centers are restricted to the range

$$\bigcup_{x,y \in X} \left[\frac{\epsilon d(x,y)}{k^2 n}, \frac{k^2 d(x,y)}{\epsilon}\right],$$

then the best solution with this restriction is worse than the optimum solution with no restriction by a factor of at most $1 + \frac{4\epsilon k^2}{k^2 - 2\epsilon} + o(\epsilon)$.

Proof. Consider any two clusters C_1, C_2 with centers c^1, c^2 of the optimum solution. Let $d_{\max}$ be the maximum distance between two points in $C_1 \cup C_2$. If $d(c^1, c^2) > k^2 d_{\max}/\epsilon$, then assigning all points in $C_1 \cup C_2$ to a single center increases the cost by at most a factor of $1 + \frac{4\epsilon}{k^2 - 2\epsilon}$. To see that, assume without loss of generality that c^1 is closer to the set $C_1 \cup C_2$ than c^2. Thus, for any point $x \in C_1 \cup C_2$, $d(x, c^1) \leq \frac{1}{2} d(c^1, c^2) + d_{\max}$ and $d(x, c^2) \geq \frac{1}{2} d(c^1, c^2) - d_{\max}$. Thus, assigning a point x to c^1 instead of c^2 could increase its distance to its cluster center by at most $2d_{\max} \leq \frac{4\epsilon}{k^2 - 2\epsilon} d(x, c^2)$.

On the other hand, if $d(c^1, c^2) < \epsilon d_{\max}/k^2 n$, then the cost of both clusters is at least $\left(\frac{1}{2} - \frac{\epsilon}{k^2 n}\right) d_{\max}$. Assigning all the points in $C_1 \cup C_2$ (there are at most n such points) to c^1 increases the cost by at most $n\epsilon d_{\max}/k^2 n = \epsilon d_{\max}/k^2$, so the total cost increases by a factor of at most $1 + \frac{2\epsilon n}{k^2 n - 2\epsilon} \leq 1 + \frac{4\epsilon}{k^2 - 2\epsilon}$.

By repeating the reassignment of points to centers for at most $\binom{k}{2}$ distances, we increase the total cost by at most a factor of $\left(1+\frac{4\epsilon}{k^2-2\epsilon}\right)^{k^2} \leq 1+\frac{4\epsilon k^2}{k^2-2\epsilon}+o(\epsilon)$. ■

References

[1] P.K. Agarwal and C.M. Procopiuc. Exact and approximation algorithms for clustering. In *Proc. SODA '98.*

[2] P.K. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Computing Surveys*, 30(4):412–458, 1998.

[3] N. Alon and J. Spencer. *The Probabilistic Method.* Wiley, 1992.

[4] N. Alon and B. Sudakov. On two segmentation problems. *Journal of Algorithms*, 33:173–184, 1999.

[5] S. Arora, D. Karger, and M. Karpinski. Polynomial-time approximation schemes for dense instances of NP-hard problems. In *Proc. 27th STOC*, pp. 284-293, 1995.

[6] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for Euclidean k-medians and related problems. In *Proc. STOC '98.*

[7] A. Borodin, R. Ostrovsky, and Y. Rabani. Subquadratic approximation algorithms for clustering problems in high dimensional spaces. In *Proc. STOC '99.*

[8] A. Broder, S. Glassman, M. Manasse, and G. Zweig. Syntactic clustering of the Web. In *Proceedings of the Sixth International World Wide Web Conference*, pp. 391-404, 1997.

[9] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595, 1995.

[10] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. Chapter 8 in D. Hochbaum, Ed. *Approximation Algorithms for Hard Problems*. PWS Publishing, 1996.

[11] M. Charikar, S. Guha, D.B. Shmoys, and É. Tardos. A constant factor approximation algorithm for the k-median problem. In *Proc. STOC '99.*

[12] M. Charikar and S. Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *Proc. FOCS '99.*

[13] D.R. Cutting, D.R. Karger, and J.O. Pedersen. Constant interaction-time scatter-gather browsing of very large document collections. In *Proc. SIGIR '93.*

[14] D.R. Cutting, D.R. Karger, J.O. Pedersen, and J.W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. SIGIR '92*, pp. 318–329.

[15] S. Dasgupta. Learning mixtures of Gaussians. In *Proc. FOCS '99.*

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

[17] W.F. de la Vega and C. Kenyon. A randomized approximation scheme for metric MAX CUT. In *Proc. FOCS '98.*

[18] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *Proc. SODA '99.*

[19] S.T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments and Computers*, 23(2):229–236, 1991.

[20] S.T. Dumais, G.W. Furnas, T.K. Landauer and S. Deerwester. Using latent semantic analysis to improve information retrieval. In *Proc. of CHI '88*, pp. 281–285.

[21] D. Eppstein. Fast hierarchical clustering and other applications of dynamic closest pair. In *Proc. of 9th SODA*, 1998.

[22] P. Frankl and H. Maehara. The Johnson-Lindenstrauss lemma and the sphericity of some graphs. *J. of Combinatorial Theory B*, 44:355–362, 1988.

[23] A. Frieze and R. Kannan. The regularity lemma and approximation schemes for dense problems. In *Proc. 37th FOCS*, pp. 12–20, 1996.

[24] A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. In *Proc. of FOCS '98*, pp. 370–378.

[25] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. In *Proc. SODA '98.*

[26] P. Indyk. A sublinear time approximation scheme for clustering in metric spaces. In *Proc. FOCS '99.*

[27] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. of 30th STOC*, pp. 604–613, 1998.

[28] K. Jain and V.V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *Proc. FOCS '99*.

[29] W.B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[30] R. Kannan and V. Vinay. The Manjara Meta-Search Engine. `http://cluster.cs.yale.edu/about.html`

[31] R.M. Karp. The genomics revolution and its challenges for algorithmic research. *Bulletin of the EATCS*, 71:151–159, June 2000.

[32] J. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. In *Proc. of 29th STOC*, pp. 599–608, 1997.

[33] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. of 9th SODA*, 1998.

[34] J. Kleinberg, C. Papadimitriou, and P. Raghavan. Segmentation problems. In *Proc. STOC '98*.

[35] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *SIAM J. Comput.*, to appear. Preliminary version appeared in *Proc. STOC '98*.

[36] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[37] E. Rasmussen. Clustering algorithms. In W.B. Frakes and R. Baeza-Yates, eds. *Information Retrieval*. Prentice Hall, 1992.

[38] J. O'Rourke and G. Toussaint. Pattern recognition. In J. Goodman and J. O'Rourke, eds. *Handbook of Discrete and Computational Geometry*. CRS press, 1997.

[39] L.J. Schulman. Clustering for edge-cost minimization. To appear in *Proc. STOC 2000*.

[40] O. Zamir, O. Etzioni, O. Madani, and R.M. Karp. Fast and intuitive clustering of web documents. In *Proc. KDD '97*.