Clustering Affine Subspaces: Algorithms and Hardness

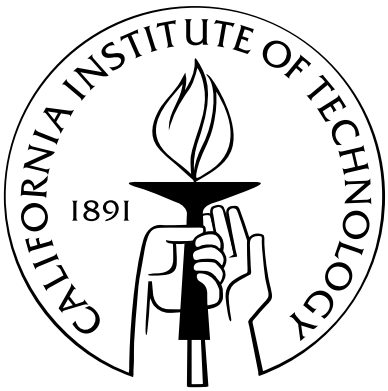
Thesis by

Euiwoong Lee

In Partial Fulfillment of the Requirements

for the Degree of

Master of Science



California Institute of Technology

Pasadena, California

2012

(Submitted)

Acknowledgements

The author is indebted to Professor Leonard J. Schulman, who supervised the thesis, for helpful criticism and advice.

Abstract

We study a generalization of the famous k-center problem where each object is an affine subspace of dimension Δ, and give either the first or significantly improved algorithms and hardness results for many combinations of parameters. This generalization from points ($\Delta = 0$) is motivated by the analysis of incomplete data, a pervasive challenge in statistics: incomplete data objects in $\mathbb{R}^d$ can be modeled as affine subspaces. We give three algorithmic results for different values of k, under the assumption that all subspaces are axis-parallel, the main case of interest because of the correspondence to missing entries in data tables.

1) $k = 1$: Two polynomial time approximation schemes which runs in $poly(\Delta, 1/\epsilon)nd$.

2) $k = 2$: $O(\Delta^{1/4})$-approximation algorithm which runs in $poly(n, d, \Delta)$

3) General k: Polynomial time approximation scheme which runs in $2^{O(\Delta k \log k(1+1/\epsilon^2))}nd$

We also prove nearly matching hardness results; in both the general (not necessarily axis-parallel) case (for $k \geq 2$) and in the axis-parallel case (for $k \geq 3$), the running time of an approximation algorithm with any approximation ratio cannot be polynomial in even one of k and Δ, unless P = NP. Furthermore, assuming that the 3-SAT problem cannot be solved subexponentially, the dependence on both k and Δ must be exponential in the general case (in the axis-parallel case, only the dependence on k drops to $2^{\Omega(\sqrt{k})}$). The simplicity of the first and the third algorithm suggests that they might be actually used in statistical applications. The second algorithm, which demonstrates a theoretical gap between the axis-parallel and general case for $k = 2$, displays a strong connection between geometric clustering and classical coloring problems on graphs and hypergraphs, via a new Helly-type theorem.

Contents

Chapter 1

Introduction

Clustering is one of the most important problems in computational theory. Due to its wide range of applications, it has been investigated for decades and continues to be actively studied not only in theoretical computer science, but in other disciplines such as statistics, data mining and machine learning. Generally speaking, the goal of clustering is to partition a given set of n data objects into k groups so that the objects within each group are similar. One way to represent data objects and measure similarity is to regard each data object with d features as a point in the Euclidean space $\mathbb{R}^d$, where each coordinate corresponds to each feature; two objects are similar when the Euclidean distance between them is small. Based on this observation, the famous k-means clustering (minimizing the sum of the squared distance from each point to the nearest center), k-median clustering (minimizing the sum of the distances), and k-center clustering (minimizing the maximum distance) problems have been suggested and studied both theoretically and practically.

However, in reality, data objects often do not come fully equipped with a mapping into Euclidean space. A simple and ubiquitous example can be found in a questionnaire where a few questions are left blank. Likewise, in a longitudinal medical study, a patient's missed appointment creates missing real-valued entries in their record.

Assume that the answers to these unanswered questions, or unmeasured values, actually exist (but are not known to us). Then a "true" data point exists, and lies on the axis-parallel affine subspace fixed by the answers or measurements which we *do* possess. No more information, however, can be retrieved from the incomplete data object.

This is a huge and pervasive problem in statistical practice, and there are many methods to deal with these incomplete data in the statistical literature [1]. Listwise deletion deletes any incomplete data object, and pairwise deletion only considers available entries in each feature to compute statistical data such as the average and the variance. The former might throw away too much information, while the latter cannot be directly applied to clustering, since it only gives a way to retrieve some statistical information about the whole set but does not specify how to represent each data object to cluster. More sophisticated methods are those filling in the missing entries using

heuristics. They include substituting with the sample mean and replacing a missing element using a learning algorithm or criterion (EM, max likelihood). While these methods work better than the simple deletion methods in practice [1], little is known about their theoretical performance, except in the very special and almost-never–satisfied case that the data is "missing at random"—with no correlation with the underlying missing entries. It is unusual that the experimenter even has any good probabilistic model for the "missingness" of entries, and the absence of such a model precludes guarantees of the type one is accustomed to for statistical procedures in the theoretical statistics literature.

As an alternative, Gao et al. [13] first suggested a new approach rooted in the geometry of a data set. A data object that is lacking information about one or more features corresponds to an affine subspace, also known as a flat, in $\mathbb{R}^d$, whose dimension is the number of missing features. Since the distance between a flat and a point (we still model each center as a point) is well-defined, the classical clustering problems such as k-means, k-median, and k-center can be defined on a set of flats; we simply need to find k centers to minimize the objective function, based on the distance between each flat to its nearest center. This approach has the strong theoretical guarantee that if we find the optimal clustering of a set of flats with respect to some objective, we automatically compute the set of points, one in each flat, such that it minimizes the objective over all such sets of points. In terms of clustering incomplete data, this method computes the best guess about incomplete entries (i.e., guess about a true point in each flat) that gives the most-clusterable complete data among all guesses. Therefore, if the set of data objects is guaranteed to be well-clustered, this geometric approach yields the correct clustering as well as the correct guesses for missing entries.

We choose to extend the k-center problem to this setting, which minimizes the maximum distance from any flat to its nearest center. Among many clustering problems, the k-center problem has the most classic and simplest theoretical guarantees; there is a simple 2-approximation algorithm [17], and it is NP-hard to approximate with an approximation ratio smaller than 1.822 [11]. However, the only geometric property that enables a simple 2-approximation algorithm, the triangle inequality, does not hold at all; even though two points are within some fixed distance from one flat, the two points can be arbitrarily far apart. This fact changes the classical k-center problem dramatically so that we need new algorithms and hardness results, even when there is only one cluster ($k = 1$). This paper starts by giving efficient algorithms for $k = 1$ and moves on to general k where we give an approximation scheme and the matching hardness results, closing with the difference between the case where all flats are axis-parallel and the general case.

1.1 Related Work

Different clustering problems have been studied extensively. The k-median problem has been often studied in a finite metric space, where the problem can be solved exactly in polynomial time if k is fixed. For general k, there is a known $(1 + 2/e)$-hardness of approximation result [21] and substantial work on approximation algorithms [18, 7, 2, 21, 9], with the best guarantee a $3 + \epsilon$ approximation. The k-means problem has been often studied in the Euclidean space, and even 2-means problem was shown to be NP-hard [8]. Instead, many PTAS's have been suggested for the Euclidean k-means and k-median problem, with the best running time polynomial in n and d but exponential in k[25, 26, 6, 9, 19, 23, 10]. Recently, the focus has been on the *well-clusterable* or *stable* instances and approximation schemes with better running times have been suggested for those instances[27, 4, 3, 22].

The k-center problem has relatively simple history where a simple 2-approximation algorithm and an almost matching hardness result were provided for general k [17, 11]. As in the Euclidean k-means and k-median, there is a PTAS which runs in time exponential in k [6]. However, while the 1-mean problem in the Euclidean space and the 1-median problem in a finite metric space can be solved trivially, the 1-center problem in the Euclidean space, which tries to find the smallest ball containing all data points, is nontrivial. There are several approximation schemes for this 1-center problem, also known as the Minimum Enclosing Ball problem [5, 29, 28], with the best running time $O(\frac{nd}{\epsilon})$ to find an enclosing ball of radius at most $(1 + \epsilon)$ times the optimal radius.

In [13], Gao et al. first suggested the problem of clustering flats as a means to cope with incomplete data, and studied the 1-center problem. They proved the existence of a $\epsilon-$*core set* of size $O(\frac{\Delta^4}{\epsilon})$; there is a subset of $O(\frac{\Delta^4}{\epsilon})$ flats of which the optimal radius is at least $\frac{1}{1+\epsilon}$ of the optimal radius of the entire set. Their main result is based on an *Intrinsic-dimension Helly theorem*, which says that there is a 1-core set of size $\Delta + 2$. However, their algorithm to find such a core set requires time exponential in Δ, so they suggested two alternative algorithms for the problem. The first algorithm using convex programming requires time $\tilde{O}(\sqrt{n}(d^3 + d^2n)poly(\Delta))$ to get a constant additive approximation, and the second algorithm using LP-type programming requires expected time $O(n(poly(\Delta)d + 2^{poly(\Delta)}))$ to get a constant multiplicative approximation. In the subsequent paper [14], the same authors studied an actual clustering problem with $k = 2, 3$ in the restricted case $\Delta = 1$, and developed 2-approximation algorithms whose running time is quasi-linear in n and d. To the best of our knowledge, all cases [$\Delta = 1$ and $k \geq 4$] and [$\Delta \geq 2$ and $k \geq 2$] were completely open (even for axis-parallel flats). For the combinations of small k and Δ already studied, it was unclear whether there were faster approximation schemes.

1.2 Our Results

We present the algorithms and hardness results for clustering flats for many possible combinations of k and Δ, where each of them either is the first result or significantly improves the previous results for the given values for k and Δ. Our algorithms and hardness results are summarized in Table 1.1 and 1.2 respectively. They can be classified into three categories.

1) Improved algorithms for $k = 1$ and the first algorithm for general k: In Section 3, we revisit the case $k = 1$ where all flats are axis-parallel and suggest two PTAS's; both are simple to describe but their running times are linear in n and d, which are even better than the fastest known algorithm, which uses a convex programming solver. We also partially explain why the problem gets harder as Δ increases. The key lemma to many 1-center algorithms, which shows an upper bound on the distance between any point and the optimal center in terms of the radius of the minimum intersecting ball centered at that point, gets relaxed as Δ increases. Extending one of these algorithms, in Section 5, we present a PTAS for fixed k and Δ for the axis-parallel case. The running time to get a $(1+\epsilon)$-approximation is $2^{O(\Delta k \log k(1+\frac{1}{\epsilon^2}))}nd$, which is tight in the sense that it is exponential in both k and Δ but linear in both n and d.

2) The first and almost matching lower bounds on approximation ratio and running time: In Section 4, we show that actual k-center clustering($k \geq 2$) is greatly harder than finding the minimum intersecting ball. In fact, we show that the running time of any approximation algorithm has to depend exponentially on both k and Δ; for fixed $k \geq 2$ there is no polynomial (in n, d, Δ) time algorithm that guarantees any approximation ratio, and for fixed $\Delta \geq 1$ there is no polynomial (in n, d, k) time approximation algorithm either. In the axis-parallel case, the same result holds for $k \geq 3$ and $\Delta \geq 3$, meaning that the axis-parallel case is not much easier than the general case when k and Δ become large. Furthermore, assuming that the 3-SAT problem cannot be solved in subexponential time, our reductions also show that there cannot be any approximation algorithm for the general case whose running time is subexponential in either k or Δ; our algorithm is almost tight in each parameter.

3) Axis-parallel vs General, Connection to Graph Coloring, and New Helly-type theorem Finally, in Section 6, we study axis-parallel 2-center, where our inapproximability ratio drops from ∞ in the general case to $\frac{2}{\sqrt{3}}$. The fact that axis-parallel 2-center is greatly easier than even slightly more general problems shows the relationship between clustering flats in the Euclidean space and the fundamental coloring problems in graphs and hypergraphs. We give a polynomial time approximation algorithm for the axis-parallel 2-center problem using an algorithm for the GRAPH 2-COLORING problem, the only version of the coloring problems that can be solved efficiently. As a tool, we prove another Helly-type theorem for a set of axis-parallel flats; when all pairwise distances are at most r, then there is a ball of radius $O(d^{\frac{1}{4}}r)$ that intersects every flat. We also give

an example where the optimal radius is $\Omega(d^{\frac{1}{4}}r)$.

k \ Δ	1	2...
1	$(R^*+\delta, O(\sqrt{n}(d^3+d^2n)poly(\Delta)\log(n/\delta)))$ [13] $((\mathbf{1+\epsilon})\mathbf{R}^*, \mathbf{O}(\frac{\mathbf{nd\Delta}}{\boldsymbol{\epsilon}^\mathbf{2}}\mathbf{log}\frac{\boldsymbol{\Delta}}{\boldsymbol{\epsilon}}))$	
2	$((1+\epsilon)R^*, O(nd\log 1/\epsilon + n\log n\log 1/\epsilon))$ [14]	$(\mathbf{O(\Delta^{1/4})R^*, O(dn^2logn)})$
3	$((1+\epsilon)R^*, O(nd\log 1/\epsilon + \frac{n\log^2 n\log 1/\epsilon}{\epsilon}))$ [14]	$((\mathbf{1+\epsilon})\mathbf{R}^*, \mathbf{2^{O(\Delta(1+1/\epsilon^2))}nd})$
4...	$((\mathbf{1+\epsilon})\mathbf{R}^*, \mathbf{2^{O(klogk(1+1/\epsilon^2))}nd})$	$((\mathbf{1+\epsilon})\mathbf{R}^*, \mathbf{2^{O(\Delta klogk(1+1/\epsilon^2))}nd})$

Table 1.1: Algorithms for different k and Δ. Each pair (x, y) indicates that it takes time y until the objective is less than x, when R^* denotes the optimum. Prior work is indicated by plain text and a citation, and applies to general flats; results of this paper are indicated by boldface, and apply to axis-parallel flats. Algorithms in column 2... work for any value of Δ, and algorithms in row 4... work for any value of k.

k \ Δ	1, 2	3...
2	1 **(PTAS)**	∞ / $\frac{\mathbf{2}}{\sqrt{\mathbf{3}}}$
3...	∞ / 1.822 [11]	∞ / ∞

Table 1.2: Lower bounds on the multiplicative approximation ratio of polynomial algorithms. Prior work is indicated by plain text and a citation; results of this paper are indicated by boldface. In each cell, the top-right corner represents the general case and the bottom-left corner represents the axis-parallel case. Results in column 3... hold for any value of Δ, and results in row 3... hold for any value of k.

Chapter 2

Preliminaries

A Δ-flat in $\mathbb{R}^d (d > \Delta)$ is a Δ-dimensional affine space. Let $\mathcal{L} = \{l_1, ..., l_n\}$ be the set of flats that we want to cluster. For simplicity, we consider the situation where all of them are of dimension Δ. All our algorithms and analyses work even when $\mathcal{L}$ contains flats of different dimensions and Δ is taken to be the greatest dimension of any flat.

For $c \in \mathbb{R}^d$ and $r \in \mathbb{R}^+$, let $B_{c,r}$ be the closed ball of radius r centered at c. The ball $B_{c,r}$ *intersects* $\mathcal{L}$ if it intersects each flat in $\mathcal{L}$. Let $R^*(\mathcal{L})$ be the minimum radius of any intersecting ball of $\mathcal{L}$. A minimum intersecting ball is B_{c,R^*} for some $c \in \mathbb{R}^d$ that intersects $\mathcal{L}$. Unlike the situation where $\mathcal{L}$ consists of points, there can be more than one minimum intersecting ball. Let $C^*(\mathcal{L})$ be the set of points such that $c \in C^*(\mathcal{L})$ if and only if c is the center of a minimum intersecting ball of $\mathcal{L}$. When it is clear from the context, we do not use $\mathcal{L}$ in these notations.

Let $d(p, q)$ be the Euclidean distance between two points p and q, $d(p, l) = \min_{q \in l} d(p, q)$ be the distance between a point p and a set of points l, and $R(c, \mathcal{L}) = \max_{l_i \in \mathcal{L}} d(c, l_i)$ be the minimum radius of any intersecting ball centered at c. Note that $R^* \leq R(c)$ for all $c \in \mathbb{R}^d$. For a flat l_i and its translation l_i' that contains the origin(i.e., l_i' is a subspace of $\mathbb{R}^d$), let $d_{l_i}(p, q)$ be the length of the vector $p - q$ projected to l_i', and $d_{l_i^\perp}(p, q)$ be the length of the vector $p - q$ projected to the orthogonal complement of l_i'. By the Pythagorean theorem, $(d(p, q))^2 = (d_{l_i}(p, q))^2 + (d_{l_i^\perp}(p, q))^2$.

Sometimes we consider the situation where all flats are axis-parallel. In this case, each flat l_i is represented as $l_i = \left\{(x_1, ..., x_d) | (x_{j_1}, ..., x_{j_\Delta}) \in \mathbb{R}^\Delta\right\}$ where each $x_u (u \notin \{j_1, ..., j_\Delta\})$ is fixed. We sometimes write l_i as a d-dimensional vector, writing $\sim$ in the jth coordinate when it is not fixed. For example, $l_j = (\sim, 5, \sim, 3, 1) = \left\{(x_1, 5, x_3, 3, 1) | (x_1, x_3) \in \mathbb{R}^2\right\}$ represents a 2-flat that is parallel to the first and third unit vector. Call jth coordinate trivial when there is no flat that has the fixed jth coordinate. We can assume that no coordinate is trivial since otherwise simply removing this coordinate from all flats will decrease Δ and d by 1 while not affecting the clustering cost for any k. Let m_j and M_j be the minimum and the maximum of the jth coordinate over all flats that have the fixed jth coordinate, respectively. Since the diameter of the minimum intersecting ball must be at least the largest difference in the jth coordinate for any j, $R^*(\mathcal{L}) \geq \max_j \frac{M_j - m_j}{2}$. Furthermore,

when the jth coordinate of c is outside of $[m_j, M_j]$ (say $c_j < m_j$), changing c_j to m_j does not increase $d(c, l_i)$ for any l_i, which means that there is an optimal center $c^* \in C^*$ in the hypercube $[m_1, M_1] \times ... \times [m_d, M_d]$. Therefore, any point c in this hypercube satisfies $R(c, \mathcal{L}) \leq 2\sqrt{d}R^*$, and $d(c, C^*) \leq 2\sqrt{d}R^*$.

Chapter 3

Two algorithms for 1-center for axis-parallel flats

We give two approximation schemes for the case $k = 1$, which is finding the smallest ball that intersects every flat. The first algorithm resembles Lloyd's method for the k-means problem [24]; in each iteration we find the closest point to the current center in each flat, and move the current center to minimize the maximum distance to these points. However, unlike Lloyd's method for $k > 1$, the convergence time of this iterative method to the global optimum can be shown to be fast, as we shown in Theorem 3.1. First, we take the initial center to be any point in $l_i \cap ([m_1, M_1] \times ... \times [m_d, M_d])$ for any i. For any optimal center c^* in the hypercube, $d_{l_i^\perp}(c, c^*) \leq d_{l_i^\perp}(c, l_i) + d_{l_i^\perp}(l_i, c^*) \leq 0 + R^* \leq R^*$ and $d^2_{l_i}(c, c^*) \leq \Delta \cdot \max_j (M_j - m_j)^2 \leq \Delta \cdot 4(R^*)^2$. Therefore, $(d(c, c^*))^2 \leq O(\Delta(R^*)^2)$ and $R(c) \leq O(\Delta(R^*)^2)$.

Algorithm 1 Lloyd-Type Algorithm

Given a set of Δ-flats $\mathcal{L} = \{l_1, ..., l_n\}$ and the initial center $c \in l_1 \cap ([m_1, M_1] \times ... \times [m_d, M_d])$

1: For $i = 1, ..., n$, find $p_i \in l_i$ which is closer to c than any other point in l_i (i.e. p_i is the projection of c to l_i).
2: Find an approximate minimum enclosing ball for $p_1, ..., p_n$ using an existing algorithm (see Section 1.1). Update c to be the center of the minimum enclosing ball.
3: Iterate 1 and 2 for the precomputed number of iterations.

The convergence of Algorithm 1 depends on how fast $R(c, \mathcal{L})$ is improved in each iteration. Given c and its projections $p_1, ..., p_n$ to $l_1, ..., l_n$ respectively, we show that there exists c' such that $R(c', \{p_1, ..., p_n\})$ is significantly less than $R(c, \{p_1, ..., p_n\}) = R(c, \mathcal{L})$. Since in each step an (approximate) minimum enclosing ball of $\{p_1, ..., p_n\}$ is found, we are guaranteed that our updated center reduces $R(c, \mathcal{L})$ significantly, leading to the convergence of the algorithm. This c' is found from the line segment joining the current center c and one of the optimal centers $c^* \in C^*$.

Theorem 3.1. *If all flats are axis-parallel, Algorithm 1 satisfies $R(1 - \epsilon_0) < R^*$ in $O(\Delta \log \Delta + \frac{\Delta}{\epsilon_0^2})$ iterations.*

Proof. We find c' on the line segment joining the current center c and one of the optimal centers such that $R(c', \{p_1, ..., p_n\})$ is significantly less than $R(c, \{p_1, ..., p_n\}) = R(c, \mathcal{L})$. Let $c^* \in C^*$ be the optimal center which is the closest to c. Let $c' = c + \frac{s}{\|c^* - c\|}(c^* - c)$ for some s that will be determined later. In other words, we consider the point obtained by moving c by s towards c^*. If c' is significantly better than c for $\{p_1, ..., p_n\}$, so will the new center computed by the minimum enclosing ball algorithm, and it implies that $R(c)$ is significantly improved.

Fix one iteration between Line 1 and Line 2 of Algorithm 1 and let $R = R(c)$ be the current radius. Suppose $R^* = \beta R$ for some $\beta \in (0, 1)$. We classify each flat into one of three different categories. The first category consists of l_i's such that $d(c, p_i) \leq R^*$. Since we moved the center by s, $d(c', p_i) \leq R^* + s$ by using the simple triangle inequality. The second category consists of l_i's such that $R^* < d(c, p_i) \leq \alpha R$ for some constant $\alpha \in (\beta, 1)$ that will be determined later. Since $d(c^*, l_i) \leq R^* < d(c, p_i)$, the angle $\angle p_i c c^* = \angle p_i c c'$ cannot be greater than $\pi/2$. Therefore, $d(c', p_i) \leq \sqrt{(\alpha R)^2 + s^2}$. The third category, which consists of l_i's such that $\alpha R < d(c, p_i) \leq R$, is the only category where each member's distance is guaranteed to be decreased by choosing the new center, and therefore needs the most involved analysis. Figure 3.1 shows the three different categories.

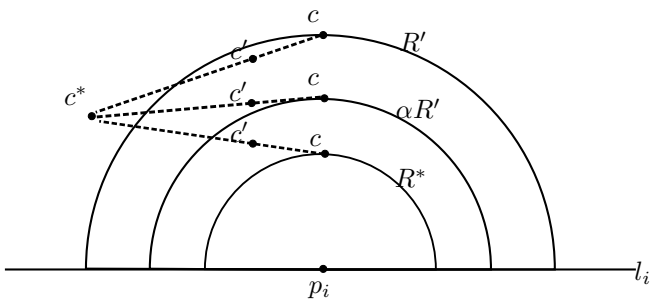


Figure 3.1: 3 possible categories depending on $d(c, p_i)$ in $d = 2, \Delta = 1$

Fix one l_i in the third category. Assume without loss of generality that it is parallel to the first Δ coordinates. In other words, $l_i = (\sim, ..., \sim, x_{\Delta+1}, ..., x_d) = \left\{(x_1, ..., x_d) | (x_1, ..., x_\Delta) \in \mathbb{R}^\Delta\right\}$ where $x_{\Delta+1}, ..., x_d$ are fixed. To show that $d(c', p_i)$ is significantly less than $d(c, p_i)$, we show an upper bound of $d_{l_i}(c, c^*)$ and a lower bound of $d_{l_i^\perp}(c, c^*)$. This means that the vector $(c^* - c)$ has the significant component orthogonal to l_i. As shown in Figure 3.2, by choosing s appropriately, we can make $d(c', p_i)$ significantly less than $d(c, p_i)$.

Lemma 3.2. $d_{l_i}(c, c^*) \leq R\sqrt{\Delta}$

Proof. Let $c = (c_1, ..., c_d), c^* = (c_1^*, ..., c_d^*)$, and assume without loss of generality that $c_j > c_j^*$ for all j. For each jth coordinate($j = 1, ..., \Delta$), there exists a flat in $\mathcal{L}$ that has the fixed jth coordinate smaller than c_j^*, since otherwise we can slightly increase c_j^* to reduce $d(c, c^*)$ while

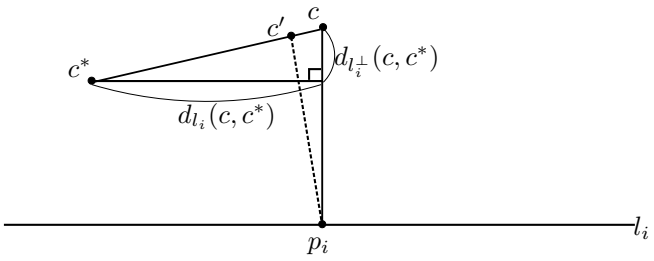


Figure 3.2: Larger $d_{l_i^\perp}(c, c^*)$ guarantees smaller $d(c', p_i)$.

not increasing $R(c^*)$, contradicting the fact that c^* is the optimal center closest to c. Therefore, $R \geq \max_{1 \leq j \leq \Delta}(c_j - c_j^*) \geq \frac{1}{\sqrt{\Delta}}\sqrt{(c_1 - c_1^*)^2 + ... + (c_\Delta - c_\Delta^*)^2} = \frac{1}{\sqrt{\Delta}} d_{l_i}(c, c^*)$.

□

Lemma 3.3. $d_{l_i^\perp}(c, c^*) \geq d(c, l_i) - R^* \geq \alpha R - R^* = (\alpha - \beta)R$.

Proof. Since $d(p, l_i) = d_{l_i^\perp}(p, l_i) = d_{l_i^\perp}(p, q)$ for any $p \in \mathbb{R}^d$ and its projection q on l_i, $d(c, l_i) = d_{l_i^\perp}(c, p_i)$ and $R^* \geq d(c^*, l_i) = d_{l_i^\perp}(c^*, p_i)$. Therefore, the triangle inequality $d_{l_i^\perp}(c, c^*) \geq d_{l_i^\perp}(c, p_i) - d_{l_i^\perp}(p_i, c^*)$ implies the first inequality. The last inequality just follows from the assumption made on the third category. □

Lemma 3.4. *Let* $s_{l_i^\perp} = d_{l_i^\perp}(c, c')$. $(d_{l_i^\perp}(c', p_i))^2 \leq R^2 - 2Rs_{l_i^\perp}\sqrt{1 - (\frac{\beta}{\alpha})^2} + (s_{l_i^\perp})^2$.

Proof. Since this lemma deals only with $d_{l_i^\perp}$, we restrict ourselves to the orthogonal complement of l_i', where l_i' is the translation of l_i that is a subspace. Figure 3.3 shows the situation projected to $(l_i')^\perp$. In this space, $l_i = p_i$ is a single point. Let θ be the angle $\angle c'cl_i = \angle c^*cl_i$. By the law of cosines, $(d_{l_i^\perp}(c', p_i))^2 \leq R^2 - 2Rs_{l_i^\perp}\cos\theta + (s_{l_i^\perp})^2$. Since $d_{l_i^\perp}(c^*, p_i) < d_{l_i^\perp}(c, p_i)$, we know that $\theta < \pi/2$. By the law of sines, $\frac{\sin\theta}{d_{l_i^\perp}(c^*, p_i)} = \frac{\sin\angle cc^*p_i}{d_{l_i^\perp}(c, p_i)} \leq \frac{1}{\alpha R}$, so $\theta \leq \arcsin\frac{R^*}{\alpha R} = \arcsin\frac{\beta}{\alpha}$. Therefore, $\cos\theta \geq \sqrt{1 - (\frac{\beta}{\alpha})^2}$.

□

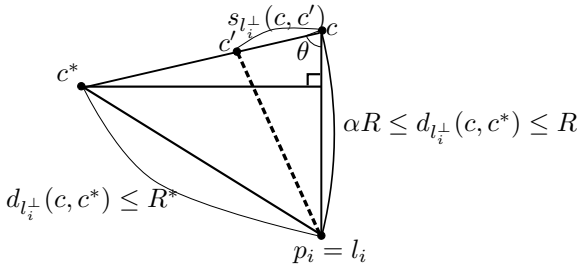


Figure 3.3: The whole space projected to $(l_i')^\perp$

Finally, we are ready to combine the results on d_{l_i} and $d_{l_i^\perp}$ to compute how we can get close to p_i by moving along $c^* - c$. By Lemma 3.2 we have $d_{l_i}(c, c^*) \leq R\sqrt{\Delta}$, and by Lemma 3.3 we have

$d_{l_i^\perp}(c, c^*) \geq (\alpha - \beta)R$. Therefore, if we make $c' = c + \frac{s}{\|c^* - c\|}(c^* - c)$, $(d_{l_i}(c, c'))^2 \leq s^2 \frac{\Delta}{\Delta + (\alpha-\beta)^2}$ and $d_{l_i^\perp}(c, c')^2 \geq s^2 \frac{(\alpha-\beta)^2}{\Delta+(\alpha-\beta)^2}$. Take s to be $R\sqrt{\frac{(\alpha-\beta)^2}{\Delta+(\alpha-\beta)^2}}\sqrt{1 - (\frac{\beta}{\alpha})^2}$.

$$
\begin{aligned}
(d(c', p_i))^2 &= (d_{l_i}(c', p_i))^2 + (d_{l_i^\perp}(c', p_i))^2 \\
&\leq s^2 \frac{\Delta}{\Delta + (\alpha-\beta)^2} + R^2 - 2Rs_{l_i^\perp}\sqrt{1 - (\frac{\beta}{\alpha})^2} + (s_{l_i^\perp})^2 \\
&= s^2 \frac{\Delta}{\Delta + (\alpha-\beta)^2} + R^2 - 2Rs\sqrt{\frac{(\alpha-\beta)^2}{\Delta + (\alpha-\beta)^2}}\sqrt{1 - (\frac{\beta}{\alpha})^2} + s^2 \frac{(\alpha-\beta)^2}{\Delta + (\alpha-\beta)^2} \\
&= R^2 - 2Rs\sqrt{\frac{(\alpha-\beta)^2}{\Delta + (\alpha-\beta)^2}}\sqrt{1 - (\frac{\beta}{\alpha})^2} + s^2 \\
&= R^2 - s^2
\end{aligned}
$$

Combining all three categories, $(d(c', p_i))^2 \leq \max((R^* + s)^2, (\alpha R)^2 + s^2, R^2 - s^2)$. $R^2 - s^2 \geq (\alpha R)^2 + s^2$ if and only if $s^2 \leq \frac{(1-\alpha^2)}{2}R^2$, and $R^2 - s^2 \geq (R^* + s)^2$ if and only if $s \leq \frac{-\beta + \sqrt{2-\beta^2}}{2}R$. $\alpha = 1/2$ and $\beta \leq 1/4$ makes $\sqrt{\frac{1}{16\Delta+1}}(\frac{3}{4})R \leq s \leq \sqrt{\frac{1}{4\Delta+1}}R$ and satisfies these two inequalities, meaning that the maximum distance to each flat is still attained in a flat in the third category. We can apply the above analysis for the third category to argue that $R^2(c)$ is decreased by at least $(1 - \Omega(\frac{1}{\Delta}))R^2$ in each iteration until $R < 4R^*$. From the inequality $(1 - \frac{c}{\Delta})^\Delta \leq e^{-c}$ for any $0 < c < \Delta$, R^2 decreases exponentially in every $O(\Delta)$ iteration. Since we can take the initial c such that $R(c) = O(\sqrt{\Delta}R^*)$, we need only $O(\Delta \log \Delta)$ iterations until $R < 4R^*$.

This analysis also shows that $R(c)$ converges to R^*, and to get $\frac{1}{1-\epsilon_0}$-approximation we need $O(\frac{\Delta}{\epsilon_0^2})$ iterations. For each iteration after $R < 4R^*$, choose ϵ be such that $\frac{R^*}{R} = \beta = 1 - 2\epsilon$ and let $\alpha = 1 - \epsilon$, making $s^2 = \frac{\epsilon^3(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2}R^2$. $R < 4R^*$ implies that $\epsilon < \frac{3}{8}$. As above, we need to check that the maximum distance to each flat is attained in a flat in the third category. $\frac{(1-\alpha^2)}{2}$ becomes $\frac{\epsilon(2-\epsilon)}{2}$, and

$$
\begin{aligned}
(\frac{\epsilon^3(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2})/(\frac{\epsilon(2-\epsilon)}{2}) &= \frac{2\epsilon^3(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2\epsilon(2-\epsilon)} \\
&= \frac{2\epsilon^2(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2(2-\epsilon)} \\
&\leq \frac{2\epsilon^2}{(\Delta+\epsilon^2)(1-\epsilon)^2} \\
&\leq 2(\frac{\epsilon}{(1-\epsilon)})^2 \\
&\leq 2(\frac{3}{5})^2 \\
&\leq 1
\end{aligned}
$$

shows that the maximum distance in the second category cannot exceed the maximum distance in the third category. For the first category, we have $\frac{-\beta+\sqrt{2-\beta^2}}{2} = \frac{2\epsilon-1+\sqrt{1+4\epsilon-4\epsilon^2}}{2}R \geq \epsilon R$.

$$\begin{aligned}(\frac{\epsilon^3(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2})/(\epsilon^2) &= \frac{\epsilon(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2} \\ &\leq \frac{\epsilon(2-3\epsilon)}{(1-\epsilon)^2} \\ &\leq 1\end{aligned}$$

since $\frac{2\epsilon-3\epsilon^2}{(1-\epsilon)^2} < 1$ for all $0 < \epsilon < 3/8$. Therefore, the maximum distance in the third category again dominates, and we can apply the same argument to show that $R^2(c)$ is decreased by $s^2 \geq \frac{\epsilon^3(2-3\epsilon)}{(\Delta+\epsilon^2)(1-\epsilon)^2}R^2 = \Omega(\frac{\epsilon^3}{\Delta})R^2$.

Therefore, as long as $R > \frac{1}{1-\epsilon_0}R^*$, R^2 is decreased by $\Omega(\frac{\epsilon_0^3}{\Delta})$. If we start from $R < 4R^*$, we need $O(\frac{\Delta}{\epsilon_0^3})$ iterations until $R > \frac{1}{1-\epsilon_0}R^*$. Since R and ϵ decrease together and ϵ in the early iterations is much bigger than ϵ_0, using the idea of Gao et al. [13], we can prove a better upper bound on the number of iterations. Assume $\epsilon_0 = 1/2^m$. For each $i = 1, ..., m$, β can be reduced from $(1 - 1/2^{i-1})$ to $(1 - 1/2^i)$ in just $O(\frac{1/2^i}{(1/2^i)^3}\Delta) = O(2^{2i}\Delta)$ iterations. Summing over $i = 1, ..., m$, the last term is greater than the sum of the other terms, so the total running time is $O(2^{2m}\Delta) = O(\frac{\Delta}{\epsilon_0^2})$. Combining the first part of iterations that make $R < 4R^*$, the number of iterations until $R(1 - \epsilon_0) < R^*$ is $O(\Delta \log \Delta + \frac{\Delta}{\epsilon_0^2})$. □

Since Algorithm 1 uses an algorithm for finding the minimum enclosing ball for points as a subroutine, its overall time complexity depends on the algorithm we use. Currently, the best algorithm for the Minimum Enclosing Ball problem runs in time $O(\frac{nd}{\epsilon})$ to get a $(1+\epsilon)$-approximation [29, 28]. Since we aim to reduce R^2 by $\Omega(\frac{1}{\Delta})$ in the first phase to have $R < 4R^*$ and $\Omega(\frac{\epsilon_0^3}{\Delta})$ in the second phase to have $R < \frac{R^*}{1-\epsilon}$, we need a $(1 + \Omega(\frac{1}{\Delta}))$-approximation in the first phase and $(1 + \Omega(\frac{\epsilon_0^3}{\Delta}))$-approximation in the second phase. Therefore, the total running time is $O(nd(\Delta^2 \log \Delta + \frac{\Delta^2}{\epsilon_0^5}))$. Even though the running time is better than that of the algorithm which uses a convex programming solver in terms of n, d, and Δ, it has the relatively bad dependence on ϵ. The next algorithm, inspired by the work of Panigrahy [28], shows that finding a minimum intersecting ball for flats can be done nearly as fast as finding the minimum enclosing ball for points except the natural penalty depending on Δ. This algorithm starts from guessing the optimal radius, which can be done easily by using binary search. With the ball of radius $(1 + \epsilon)R^*$, we find a flat that is not covered by the ball, move the ball until it hits the flat. Its convergence is guaranteed by the fact that $d(c, c^*)$ will be decreased in each iteration.

Theorem 3.5. *If all flats are axis-parallel and R^* is given, Algorithm 2 computes an intersecting ball with radius $R < R^*(1 + \epsilon)$ in time $O(\frac{nd\Delta}{\epsilon^2})$.*

Algorithm 2 Moving Center Algorithm

Given a set of Δ-flats $\mathcal{L} = \{l_1, ..., l_n\}$, the initial center $c \in l_1 \cap [m_1, M_1] \times ... \times [m_d, M_d]$, and the optimal radius $R^* = R^*(\mathcal{L})$,

1: Find the farthest flat l_i from the current center c. Find $p_i \in l_i$ such that $d(c, l_i) = d(c, p_i)$ (i.e. p_i is the projection of c to l_i).
2: If $d(c, l_i) < (1 + \epsilon)R^*$, we found a $(1 + \epsilon)$-approximate minimum intersecting ball. Stop.
3: Move c towards p_i until B_{c,R^*} contains p_i (can compute in closed form).
4: Iterate 1, 2, and 3 for the precomputed number of iterations.

Proof. Let l_i be the flat chosen in the ith iteration. Let c be the center before we move it towards l_i, and c' be the center after the move. Since c' is on the line segment joining c and its projection p_i to l_i and every distance from a point l_i is defined in $l_i^\perp$, we can restrict our attention to the orthogonal complement of l_i'. In this space, $l_i = p_i$ becomes a single point. $d(c^*, l_i) = d_{l_i^\perp}(c^*, l_i) \leq R^*$, $d(c, l_i) = d_{l_i^\perp}(c, l_i) > (1 + \epsilon)R^*$, $d(c', l_i) = d_{l_i^\perp}(c', l_i) = R^*$

Since $d_{l_i^\perp}(c^*, l_i) \leq d_{l_i^\perp}(c, l_i)$, $\angle l_i c' c^*$ is acute and $\angle c c' c^*$ is obtuse as shown in Figure 3.4. Therefore, $(d_{l_i^\perp}(c^*, c'))^2 \leq (d_{l_i^\perp}(c^*, c))^2 - (d_{l_i^\perp}(c, c'))^2 \leq (d_{l_i^\perp}(c^*, c))^2 - \epsilon^2$. Therefore, in each iteration, $(d_{l_i^\perp}(c^*, c))^2$ is decreased by ϵ^2, and so is $(d(c^*, c))^2$. Since we take the initial center c such that $d(c, c^*) \leq O(\sqrt{\Delta})R^*$, so we need only $O(\frac{\Delta}{\epsilon^2})$ iterations. Each iteration only consists of computing the distance and the projection to each flat from the current center, which takes time $O(nd)$. The total running time is $O(\frac{nd\Delta}{\epsilon^2})$.

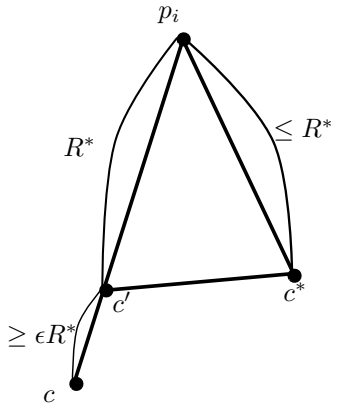


Figure 3.4: The center of the ball gets closer to the optimal position in each iteration

□

Note that our initial center also satisfies $R(c) \leq O(\sqrt{\Delta}R^*)$. Therefore, to guess the optimal radius with error $< \epsilon$, we need at most $O(\log \frac{\Delta}{\epsilon})$ times of guessing. Therefore, the overall running time is $O(\frac{nd\Delta}{\epsilon^2} \log \frac{\Delta}{\epsilon})$.

Why is the running time still worse than the original Minimum Enclosing Ball problem, and why does it depend on Δ? Earlier we mentioned that the triangle inequality does not hold when $\Delta \geq 1$. Algorithmically, the lack of the triangle inequality prevented us from using a simple but powerful

lemma first proved in [16]; when c and c^* are the current and the optimal center respectively, we can find a point p such that $d(c^*, p) = R^*$ and $d(c, p)^2 \geq (R^*)^2 + d(c, c^*)^2$, which implies $d(c, c^*)^2 \leq R^2 - (R^*)^2$. Many algorithms for the Minimum Enclosing Ball problem [6, 28] used this lemma to improve their running time. Also in our algorithms, this lemma would guarantee a tighter bound of the distance between the current center and the optimal center, giving a better running time. However, the next lemma shows that the bound of this key lemma gets gradually relaxed as Δ increases. When Δ is big, $d(c, c^*)$ can be nearly as big as R^*, which does not improve our convergence analyses.

Lemma 3.6. *Let c be the current center, $c^* \in C^*$ be the closest optimal center to c, and $R = R(c, \mathcal{L})$ be the radius of the minimum intersecting ball centered at c. There is an instance where $(d(c, c^*))^2 \geq (R^2 - (R^*)^2)^{\frac{1}{2^\Delta}} (R^*)^{(2 - \frac{2}{2^\Delta})}$.*

Proof. Our counterexample has $n = d = \Delta + 2$, and consists of n flats of different dimensions. Note that this can be easily extended to the case where all flats have the same dimension Δ, by adding at most Δ coordinates and giving either the same fixed value or a free variable to each added coordinate.

$\mathcal{L} = \{l_1, l_2, p_1, ..., p_\Delta\}$. l_1 and l_2 are of dimension Δ and each p_i is of dimension less than Δ. Let $l_1 = (-I, 0, \sim, ..., \sim), l_2 = (+I, 0, \sim, ..., \sim)$ and $p_i = (0, ..., 0, I, 0, \sim, ..., \sim)$ where $I > 0$ appears in the $i + 1$th coordinate. In other words, p_i has the first i coordinates 0, $i + 1$th coordinate I, $i + 2$th coordinate 0, and the remaining coordinates $\sim$.

Claim 3.7. *The origin is the center of unique the minimum intersecting ball, i.e. $C^* = \{O\}$.*

Proof. Since the first coordinate of l_1 and l_2 differ by $2I$, $R^* \geq I$. It is obvious that $B_{O,I}$ intersects all flats, since each flat has only one nonzero and nonfree coordinate of which the value is I. We argue that there is no other center c that achieves I.

By l_1 and l_2, it is obvious that to achieve the radius I, the first coordinate of c must be zero. We consider $p_1, ..., p_\Delta$ in this order. Inductively, when we look at p_i and the first i coordinates are fixed to 0, it means that the distance from c to each of $l_1, l_2, p_1, ..., p_{i-1}$ in the first i coordinates is I. Since the value of $(i + 1)$th coordinate of each of these flats is 0 or $\sim$, c must have the the $i + 1$ coordinate 0. Again, the distance from c to p_i in the first $i + 1$ coordinates is I, so the induction is complete until all coordinates of c must be zero.

□

Let $c' = (0, x^{2^\Delta}, x^{2^{\Delta-1}}, ..., x^2, x)I$.

$$\begin{aligned}
(d(c', l_1))^2 &= (1 + x^{2^{\Delta+1}})I^2 \\
(d(c', p_1))^2 &= ((1 - x^{2^\Delta})^2 + x^{2^\Delta})I^2 = (1 + x^{2^{\Delta+1}} - x^{2^\Delta})I^2 \\
(d(c', p_2))^2 &= (x^{2^{\Delta+1}} + (1 - x^{2^{\Delta-1}})^2 + x^{2^{\Delta-1}})I^2 = (1 + x^{2^{\Delta+1}} + x^{2^\Delta} - x^{2^{\Delta-1}})I^2 \\
&... \\
(d(c', p_i))^2 &= (x^{2^{\Delta+1}} + ... + x^{2^{\Delta-i+3}} + (1 - x^{2^{\Delta-i+1}})^2 + x^{2^{\Delta-i+1}})I^2 = (1 + x^{2^{\Delta+1}} + ... + x^{2^{\Delta-i+2}} - x^{2^{\Delta-i+1}})I^2
\end{aligned}$$

$1 + x^{2^{\Delta+1}} + ... + x^{2^{\Delta-i+2}} - x^{2^{\Delta-i+1}} \leq 1$ for small x, so $(R(c'))^2 = (d(c', l_1))^2 = (1 + x^{2^{\Delta+1}})I^2$. $(R(c'))^2 - (R^*)^2 = x^{2^{\Delta+1}} I^2$, but

$$\begin{aligned}
(d(c^*, c'))^2 &\geq x^2 I^2 \\
&\geq (x^{2^{\Delta+1}})^{\frac{1}{2^\Delta}} I^{\frac{2}{2^\Delta}} I^{2-\frac{2}{2^\Delta}} \\
&= ((R(c'))^2 - (R^*)^2)^{\frac{1}{2^\Delta}} (R^*)^{2-\frac{2}{2^\Delta}}
\end{aligned}$$

□

Chapter 4

Clustering Hardness

A k-clustering $\mathcal{C} = (\mathcal{L}_1, ..., \mathcal{L}_n, c_1, ..., c_k)$ is a partition of $\mathcal{L}$ into disjoint subsets (clusters) $\mathcal{L}_1, ..., \mathcal{L}_k$ with centers $c_1, ..., c_k$. Let $R(\mathcal{C}) = \max_i R(c_i, \mathcal{L}_i)$ be the maximum radius of any cluster of $\mathcal{C}$, which we also call the radius of the clustering $\mathcal{C}$. Let $R^*(k, \mathcal{L})$ be the optimal radius of any k-clustering. Again, we do not use $\mathcal{L}$ or $\mathcal{C}$ in these notations when it is clear from the context.

While we have a FPTAS for the 1-center problem, that is not the case when we have a real clustering problem with $k \geq 2$. The fundamental concept of clustering - partitioning objects into k groups so that the objects in each group are similar - has been considered in even the earliest problems in computational theory when each object corresponds to a vertex in a graph and there is an edge when two objects are similar or related; the VERTEX COVER and DOMINATING SET problem ask to find k *centers* that cover their relationships (incident edges) or directly similar objects (neighbors), respectively. k-COLORING, which is finding a partition of the vertex set into k disjoint sets such that the subgraph induced by each set is a coclique (here, the existence of an edge means that the two vertices are different), looks more like a version of the clustering problem. Even more boldly, some variants of the 3-SAT problem can be said to look for a partition of a set of variables into 2 disjoint sets (those assigned true or false) to satisfy the given constraints, if flipping true and false does not change the satisfiability (i.e., clusters are symmetric).

These many fundamental graph-theoretic or logic problems that are related to the concept of clustering, and their hardness results have not been reflected very much in the typical clustering problems studied in the Euclidean space. One of the reasons is that it is hard to find a right embedding of a given graph to the Euclidean space such that the relationships are preserved. When objects correspond to Δ-flats, however, such embeddings can be found more easily, enabling us to prove that clustering flats is much harder than clustering points. Some interesting graph-theoretic properties are preserved in the reduction to our problem, as the distinction of easy 2-COLORING and hard 3-COLORING is reflected in the strict better approximability with 2 centers than with 3 centers in the axis-parallel case. We start by reducing the k-COLORING problem to our problem, which shows that k-clustering is hard even for fixed $k \geq 3$ and the axis-parallel restriction.

Theorem 4.1. *For fixed $k \geq 3$ and any I, there is no algorithm that computes a I-approximate k-clustering and runs in time polynomial of n, d, Δ unless $P = NP$.*

Proof. We reduce the GRAPH k-COLORING problem($k \geq 3$) to our problem. Given $G = (V, E)$ with $V = \{v_1, ..., v_{|V|}\}$ and $E = \{(v_{1,1}, v_{1,2}), ..., (v_{|E|,1}, v_{|E|,2})\}$, we construct the set of $n = |V|$ flats $\mathcal{L} = \{l_1, ..., l_n\}$ in $d = |E|$-dimensional coordinates. Each flat l_i corresponds to each vertex v_i and each jth coordinate corresponds to each edge $(v_{j,1}, v_{j,2})$. For each $1 \leq i \leq n$ and $1 \leq j \leq m$, $(l_i)_j$ is decided as the following:

- $v_i \notin (v_{j,1}, v_{j,2})$: $(l_i)_j = \sim$, i.e. s_i does not have a fixed value in the jth coordinate.
- $v_i = v_{j,1}$: $(l_i)_j = -1$
- $v_i = v_{j,2}$: $(l_i)_j = +1$

G is k-colorable if and only if there is a k-clustering where each cluster does not have two flats that have the fixed values -1 and $+1$ in the same coordinate, which ensures $R^*(k, \mathcal{L}) = 0$. Therefore, G is k-colorable if and only if there is a k-clustering with $R^*(k, \mathcal{L}) = 0$. If there is an algorithm that computes a I-approximate k-clustering and runs in time polynomial of n, d, Δ, we can solve the GRAPH k-COLORING problem, implying $P = NP$.

□

Note that all the flats used in the above theorem are axis-parallel, so both axis-parallel and general case are hard when $k \geq 3$. Since 2-COLORING is in P, a harder version of coloring(HYPERGRAPH 2-COLORING) is used to show that 2-clustering in the general (not necessarily axis-parallel) case is still hard.

Theorem 4.2. *There is no algorithm that computes a I-approximate 2-clustering and runs in time polynomial of n, d, Δ for any I unless $P = NP$.*

Proof. We reduce the SET SPLITTING problem, also known as the HYPERGRAPH 2-COLORING problem, to our problem. The SET SPLITTING problem, given a collection S of subsets of a finite set U, decides whether there is a partition of U into two subsets U_1 and U_2 such that no subset in S is entirely contained in either U_1 or U_2. This problem remains NP-complete even if all $s \in S$ have $|s| \leq 3$. Therefore, we assume that $|s| = 3$ for all $s \in S$ without losing NP-completeness.

Let $U = \{u_1, ..., u_n\}$ be the given finite set, and $S = \{s_1, ..., s_m\}$ be the collection of subsets. For each j, $s_j = \{u_{j,1}, u_{j,2}, u_{j,3}\} \subseteq S$. Given such an instance, we construct the set of n flats $\mathcal{L} = \{l_1, ..., l_n\}$ in $d = 2m$-dimensional coordinates. Think $(2j-1)$th and $(2j)$th coordinates are associated with s_j. For each $1 \leq i \leq n$ and $1 \leq j \leq m$, $(l_i)_{2j-1}$ and $(l_i)_{2j}$ are decided as the following. Let $A = (0, 1), B = (-\frac{\sqrt{3}}{2}, -\frac{1}{2}), C = (\frac{\sqrt{3}}{2}, -\frac{1}{2})$ be the vertices of the triangle that contains 0 in the middle.

- $u_i \notin s_j$: $(l_i)_{2j-1} = (l_i)_{2j} = \sim$, i.e. s_i does not have a fixed value in these coordinates.
- $u_i = u_{j,1}$: $\{(l_i)_{2j-1}, (l_i)_{2j}\} \subseteq \mathbb{R}^2$ is the line passing A and B.
- $u_i = u_{j,2}$: $\{(l_i)_{2j-1}, (l_i)_{2j}\} \subseteq \mathbb{R}^2$ is the line passing A and C.
- $u_i = u_{j,3}$: $\{(l_i)_{2j-1}, (l_i)_{2j}\} \subseteq \mathbb{R}^2$ is the line passing B and C.

	...	$(2i-1, 2i)$	...	$(2j-1, 2j)$	...
l_1	...	/	...	$\sim$	...
l_2	...	\	...	/	...
l_3	...	--	...	\	...
l_4	...	$\sim$	...	--	...

Table 4.1: An example where $S_i = \{u_1, u_2, u_3\}$ and $S_j = \{u_2, u_3, u_4\}$. Three lines in the same two columns form a regular triangle around the origin.

Let (C_1, C_2) be a 2-clustering of $\mathcal{L}$. If $l_{j,1}, l_{j,2}, l_{j,3}$ are in the same cluster for any j, the minimum intersecting ball for that cluster must cover the triangle formed in $(2j-1)$th and $(2j)$th coordinates, forcing $R^*(k, \mathcal{L}) \geq 1/2$. If $l_{j,1}, l_{j,2}, l_{j,3}$ are split for every j, each cluster has at most 2 of them and we can place the center on the intersection of those two lines, so $R^* = 0$. Therefore, $R^*(k, \mathcal{L}) = 0$ if and only if the given instance of the SET SPLITTING problem has a satisfactory partitioning. If there is an algorithm that computes a I-approximate 2-clustering and runs in time polynomial of n, d, Δ, we can solve the SET SPLITTING problem with $|s| = 3$ for all $s \in S$, implying $P = NP$.

□

This construction is not axis-parallel. By replacing three edges of a triangle with three vertices with some additional changes similar to the NP-hardness of 2-means[8], we can make each flat used in the reduction axis-parallel. However, in this case the inapproximability ratio reduces from infinity to a constant less than 2. In Section 6, we show a $O(\Delta^{1/4})$-approximation algorithm for $k = 2$, indicating the fundamental difference between the axis-parallel case and the general case in 2-clustering.

Theorem 4.3. *When all flats are restricted to be axis-parallel, there is no algorithm that computes a $\frac{2}{\sqrt{3}}$-approximate 2-clustering and runs in time polynomial of n, d, Δ unless $P = NP$.*

Proof. We reduce the NOT ALL EQUAL 3-SAT problem, which is quite similar to the SET SPLITTING PROBLEM introduced above, to our problem. Given a set of variables $x_1, ..., x_n$ and a set of clauses $C = \{C_1, ..., C_m\}$ where each C_j consists of three literals $x_{j,1}, x_{j,2}, x_{j,3}$, negated or unnegated, the problem asks to find the truth assignment to each x_i such that no clause has three true literals or three false literals. It is also NP-hard to find an exact solution for the NOT ALL EQUAL 3-SAT problem.

Given an instance of the NOT ALL EQUAL 3-SAT problem, we construct the set of $6m$ flats $\mathcal{L} = \left\{l_{i,j}, l'_{i,j} | x_i \in C_j\right\}$ in $d = (n+2m)$-dimensional coordinates. The number of flats is $6m$ since we generate two flats, $l_{i,j}$ for the literal and $l'_{i,j}$ for its negation, per each appearance of a literal. The purpose of the first n coordinates is to make sure that all $l_{i,j}$'s for fixed i are in the same cluster and all $l'_{i,j}$'s in the other, and the purpose of remaining $2m$ coordinates is to penalize the clause with the three literals with the same value. For the first n coordinates, $(l_{i,j})_i = 0$, $(l'_{i,j})_i = M$ for a large M, and $(l_{i,j})_{i'} = (l'_{i,j})_{i'} = \sim$ when $i \neq i'$. Therefore, $l_{i,j}$ and $l'_{i,j'}$ cannot be in the same cluster even if $j \neq j'$. Since we have only two clusters, for fixed i, all $l_{i,j}$'s have to gather in the one cluster and all $l'_{i,j}$'s have to gather in the other.

The $(n+2j-1)$th and $(n+2j)$th coordinates are associated with C_j. For each $1 \leq i \leq n$ and $1 \leq j, j' \leq m$ such that $x_i \in C_j$ (i.e. $l_{i,j} \in \mathcal{L}$), $(l_{i,j})_{2j'-1}$, $(l_{i,j})_{2j'}$, $(l'_{i,j})_{2j'-1}$ and $(l'_{i,j})_{2j'}$ are decided as the following:

- $j \neq j'$: $(l_{i,j})_{2j-1} = (l_{i,j})_{2j} = (l'_{i,j})_{2j} = (l'_{i,j})_{2j} = \sim$, i.e. $l_{i,j}$ and $l'_{i,j}$ does not have a fixed value in these coordinates.
- $x_i = x_{j,1}$: $(l_{i,j})_{2j-1} = 0, (l_{i,j})_{2j} = 1$. $(l'_{i,j})_{2j-1} = -\frac{\sqrt{3}}{2}, (l'_{i,j})_{2j} = -\frac{1}{2}$. Swap the role of $l_{i,j}$ and $l'_{i,j}$ if $x_{j,1}$ is negated.
- $x_i = x_{j,2}$: $(l_{i,j})_{2j-1} = -\frac{\sqrt{3}}{2}, (l_{i,j})_{2j} = -\frac{1}{2}$. $(l'_{i,j})_{2j-1} = \frac{\sqrt{3}}{2}, (l'_{i,j})_{2j} = -\frac{1}{2}$. Swap the role of $l_{i,j}$ and $l'_{i,j}$ if $x_{j,2}$ is negated.
- $x_i = x_{j,3}$: $(l_{i,j})_{2j-1} = \frac{\sqrt{3}}{2}, (l_{i,j})_{2j} = -\frac{1}{2}$. $(l'_{i,j})_{2j-1} = 0, (l'_{i,j})_{2j} = 1$. Swap the role of $l_{i,j}$ and $l'_{i,j}$ if $x_{j,3}$ is negated.

	1	2	3	...	$(2j-1, 2j)$	...
$l_{1,j}$	0	$\sim$	$\sim$	...	$\uparrow$	...
$l'_{1,j}$	M	$\sim$	$\sim$	...	$\rightarrow$	...
$l_{2,j}$	$\sim$	0	$\sim$	...	$\leftarrow$	...
$l'_{2,j}$	$\sim$	M	$\sim$	...	$\rightarrow$	...
$l_{3,j}$	$\sim$	$\sim$	0	...	$\leftarrow$	...
$l'_{3,j}$	$\sim$	$\sim$	M	...	$\uparrow$	...

Table 4.2: An example where $C_j = (x_1, \neg x_2, x_3)$. $(0,1), (-\frac{\sqrt{3}}{2}, -\frac{1}{2}), (\frac{\sqrt{3}}{2}, -\frac{1}{2})$ are represented as up, left, right arrow vectors, respectively.

The above table shows the example of a clause C_j which consists of $x_{j,1} = x_1, \neg x_{j,2} = \neg x_2, x_{j,3} = x_3$. Since $l_{i,j}$ and $l_{i,j'}$ must be separated, there are at most $2^3 = 8$ clusterings (if we do consider the order of the clusters), and we cannot have the three same points in one cluster. Furthermore, there are exactly two cases where we can have three different points in one cluster, namely $l_{1,j}, l'_{2,j}, l_{3,j}$ in one cluster and $l'_{1,j}, l_{2,j}, l'_{3,j}$ in the other (again, when we do not consider the order of the clusters,

it becomes one case). They are exactly the two assignments that make all the literals true or false at the same time. By symmetry, this argument holds for any clause with arbitrary negated or unnegated literals.

When all $l_{i,j}$'s gather in the one cluster and all $l'_{i,j}$'s gather in the other for fixed i, the first n coordinates do not affect the radius of the clustering at all. For each two coordinates corresponding to C_j, there are only 6 flats with fixed values, 3 for each cluster. And these flats do not have fixed values for any other coordinate corresponding to $C_{j'} \neq C_j$. Therefore, the $(n+2j-1)$th and $(n+2j)$th coordinates of the center are decided by these coordinates of the 3 flats (either three points or two points) only, and the distances of these flats to the center are also decided by only those two coordinates. It implies that the radius of a clustering is 1 if and only if there is a clause for which each cluster corresponds to the assignment resulting in three literals with the same value. Otherwise, the radius of clustering is $\frac{\sqrt{3}}{2}$.

Therefore, there exists a satisfying assignment to an instance of the NOT ALL EQUAL 3-SAT problem if and only if the optimal radius is $\frac{\sqrt{3}}{2}$. If there is an algorithm that computes a $\frac{2}{\sqrt{3}}$-approximate 2-clustering and runs in time polynomial of n, d, Δ, we can solve the NOT ALL EQUAL 3-SAT problem, implying $P = NP$.

□

The three theorems above show that it is impossible to find a good approximation algorithm that runs in time polynomial in n, d, Δ, even for fixed k. We also study the case where Δ is fixed. Similarly to what we have proved, it is impossible to find any approximation algorithm that runs in time polynomial in n, d, k, even for fixed Δ. The reduction is from the VERTEX-COVER problem.

Theorem 4.4. *For fixed $\Delta \geq 1$, there is no algorithm that computes a I-approximate k-clustering and runs in time polynomial of n, d, k for any I unless $P = NP$.*

Proof. We reduce the famous VERTEX COVER problem to our problem with $\Delta = 1$. Given $G = (V, E)$ with $V = \{v_1, ..., v_{|V|}\}$ and $E = \{(v_{1,1}, v_{1,2}), ..., (v_{|E|,1}, v_{|E|,2})\}$, we construct the regular $d = (|V|-1)$-simplex whose $|V|$ vertices corresponds to each vertex in V. $\mathcal{L}$ consists of $n = |E|$ lines where each line corresponds to each edge; l_i is the line passing $v_{i,1}$ and $v_{i,2}$.

If there is a vertex cover of size k, using the vertices in the vertex cover as centers ensures $R^*(k, \mathcal{L}) = 0$, since each line contains at least one center. Conversely, if there is a k-clustering with $R^*(k, \mathcal{L}) = 0$, we can move each center to some vertex while maintaining $R^* = 0$ since there is no intersection of any two lines except the vertices of the simplex. Therefore, there is a vertex cover of size k if and only if there is a k-clustering with $R^*(k, \mathcal{L}) = 0$.

If there is an algorithm that computes a I-approximate k-clustering and runs in time polynomial of n, d, k for any I, we can solve the VERTEX COVER problem, implying $P = NP$.

□

Again, the flats used in the reduction are not axis-parallel. By allowing a slightly larger dimension and more involved analysis, we can use the same idea to extend the proof to the axis-parallel case.

Theorem 4.5. *When all flats are restricted to be axis-parallel, for fixed $\Delta \geq 3$, there is no algorithm that computes a I-approximate k-clustering and runs in time polynomial of n, d, k for any I unless $P = NP$.*

Proof. We reduce the 3-REGULAR VERTEX COVER problem to our problem. First, we prove that the problem remains NP-hard even though all graphs are restricted to have no cycle of length ≤ 4. For any given graph $G = (V, E)$, construct the graph G' with $|V| + 2|E|$ vertices and $3|E|$ edges by splitting each edge of G into three edges and adding two vertices of degree two in each junction. If G has a vertex cover of size k, G' has a vertex cover of size $k + |E|$ since if an edge of G is covered, at least one edge of G' corresponding to the original edge is covered, so we need only one vertex to cover the remaining two edges of G' corresponding to the original edge.

For the other direction, suppose G' has a vertex cover of size $k+|E|$ and this vertex cover contains k' vertices of G. Since this vertex cover has to cover each middle edge (the middle of three pieces of the original edge) that does not contain any vertex of G (there are $|E|$ of them), $k' \leq k$. Also, if the two endpoints of any middle edge are in the vertex cover, moving one of them to the nearest vertex of G still ensures that all edges of G' are covered (it is okay even if the nearest vertex was already in the cover). This may increase k', but still $k' \leq k$. Now we claim that these k' vertices of G form a vertex cover of G. If there is an uncovered edge of G by these vertices, the only way to cover the three edges of G' is to include two new vertices into the vertex cover of G', but we already removed these cases. Therefore, G has a vertex cover of size k if and only if G' has a vertex cover of size $k + |E|$, completing the first reduction.

Given $G = (V, E)$ with $V = \{v_1, ..., v_{|V|}\}$ and $E = \{e_1, ..., e_{|E|}\}$, we construct the set of $n = |E|$ flats $\mathcal{L} = \{l_1, ..., l_n\}$ in $d = |V|$-dimensional coordinates. Think jth coordinate is associated with v_j. For each $1 \leq i \leq n$ and $1 \leq j \leq m$, $(l_i)_j$ is decided as the following:

- v_j is an endpoint of e_i: $(l_i)_j = 1$.
- v_j is a neighbor of an endpoint of e_i: $(l_i)_j = \sim$.
- Otherwise: $(l_i)_j = 0$.

Note that in a graph of which the maximum degree is ≤ 3, one edge can have at most 4 vertices (excluding its endpoints) that are adjacent to its vertices. However, using the reduction above to split each edge into three, we are sure that no adjacent vertices are both of degree 3, so each edge can have at most 3 such vertices.

For fixed j, if $c \in \mathbb{R}^d$ is such that $c_j = 1$ and $c_{j'} = 0$ for $j' \neq j$, c intersects all flats corresponding to the edges incident on v_j. Therefore, if there is a vertex cover of size k, there is a k-clustering

$\mathcal{C}$ with $R(\mathcal{C}) = 0$. If there is no vertex cover of size k, for any partition of the flats(edges) into k clusters, there must be a cluster whose edges are not covered by a single vertex. There are two cases.

- There exists $e_i = (v_x, v_y), e_{i'} = (v_z, v_w)$ such that x, y, z, w are different:
  Note that $(l_i)_x = (l_i)_y = 1$. However, it is impossible that $(l_{i'})_x = \sim, (l_{i'})_y = \sim$, because this implies both v_x and v_y are adjacent to at least one of v_z and v_w, which results in a cycle of length 3 or 4. There must be at least one coordinate in which l_i is 1 and $l_{i'}$ is 0, implying that $R^*(k, \mathcal{L}) > 0$.

- Each pair of edges shares an endpoint:
  Take any edge (v_x, v_z) from this cluster. Since this cluster is not covered by either v_x or v_z, there must be an edge $e_i = (v_x, v_y)$ and $e_{i'} = (v_z, v_w)$ for some $y \neq z$ and $w \neq x$. $y = w$ means a cycle of length 3, so $y \neq w$, implying that x, y, z, w are all different again.

Therefore, G admits a vertex cover of size k if and only if $R^*(k, \mathcal{L}) = 0$. If there is an algorithm that computes a I-approximate k-clustering and runs in time polynomial of n, d, k for any I, we can solve the 3-REGULAR VERTEX COVER problem, implying $P = NP$. □

Note that our reductions do not increase the size of each instance much. Therefore, if we assume the exponential time hypothesis formalized in [20], which implies that the 3-SAT problem cannot be solved in $2^{o(n)}$, our reductions also imply that we cannot expect an algorithm whose running time is subexponential in k or Δ, with the only exception being the axis-parallel case for fixed Δ.

Theorem 4.6. *Assuming the exponential time hypothesis, for fixed $k \geq 3$, there is no algorithm that computes I-approximate k-clustering and runs in time $2^{o(\Delta)}poly(n, d)$, both for the general and the axis-parallel case for any I. For fixed $\Delta \geq 1$, there is no such algorithm for the general case which runs in time $2^{o(k)}poly(n, d)$. In the axis-parallel case, for fixed $\Delta \geq 3$, there is no such algorithm which runs in time $2^{o(\sqrt{k})}poly(n, d)$.*

Proof. Note that if there is an approximation algorithm with any factor I, the problems that have been used in our reductions will be solved because all of our reductions mapped the accepting instances to the clustering instances with $R^* = 0$, and the rejecting instances to those with $R^* > 0$.

Assuming the exponential time hypothesis, the 3-SAT problem, the GRAPH 3-COLORING problem, and the VERTEX COVER problem (with k as a part of input, not a parameter) cannot be solved in time $2^{o(n+m)}$, where n is the number of vertices (variables) and m is the number of edges (clauses) [12]. Our reduction from the GRAPH 3-COLORING in Theorem 4.1, where we used $d = |E|$ dimensions, shows that, for fixed k, we cannot obtain an algorithm whose running time is subexponential in Δ. This result holds both in the general case and in the axis-parallel case since the range of the reduction consists of axis-parallel instances.

For fixed Δ, our reduction in Theorem 4.4 preserves k in the original VERTEX-COVER problem, so we cannot have a subexponential (in k) algorithm in the general case. In the axis-parallel case, the reduction from the 3-SAT problem to the 3-REGULAR VERTEX COVER problem increases the size $(n + m)$ of the problem at most quadratically [15], and the reduction to the 3-REGULAR VERTEX COVER without any cycle of length ≤ 4 in Theorem 4.5 increases the size of the graph linearly$(n' = n + 2m, m' = 3m)$. Therefore, an algorithm for the axis-parallel case which runs in time $2^{o(\sqrt{k})}poly(n, d)$ solves the 3-REGULAR VERTEX COVER problem in time $2^{o(\sqrt{n+m})}$ and the 3-SAT problem in time $2^{o(n+m)}$, contradicting the exponential time hypothesis. $\square$

Chapter 5

Clustering Algorithms

In the previous section, we showed that there cannot be any polynomial time approximation algorithm even when one of k or Δ is fixed. Therefore, the natural next step is to find an approximation algorithm which is fast for small k and Δ. Using Algorithm 2 (Moving Center) for computing the minimum intersecting ball, we can obtain a PTAS whose running time is linear in n and d for fixed k and Δ. Note that algorithm 2 starts with an initial center whose distance to the closest optimal center is bounded by $O(\sqrt{\Delta}R^*)$. However, in clustering, it is impossible, because R^* can be arbitrarily small compared to $\max_j (M_j - m_j)$; given two points, the radius of the minimum enclosing ball is at least half of the distance between these points, but the radius of the optimal clustering becomes zero when we allow two clusters. Thus, we need to study how Algorithm 2 behaves if it starts with an arbitrary center. Fortunately, with some modification on the algorithm, this does not affect the running time greatly. Since the difference between the optimal center and the current center in the jth coordinate is bounded by R^* once the jth coordinate is *considered* (the center has move to a flat with a fixed value for the jth coordinate), and we have only Δ unconsidered coordinates after the first move, we need only $\Delta+1$ additional steps to move to a reasonable position. Figure 5.1 shows a simple example in which an arbitrary initial center moves to the desired position in $2 = \Delta + 1$ steps.

Theorem 5.1. *If all flats are axis-parallel and R^* is given, Algorithm 2 with an arbitrary initial center c computes an intersecting ball with radius $R < R^*(1+\epsilon)$ in time $O(nd\Delta(\frac{1}{\epsilon^2}+1))$.*

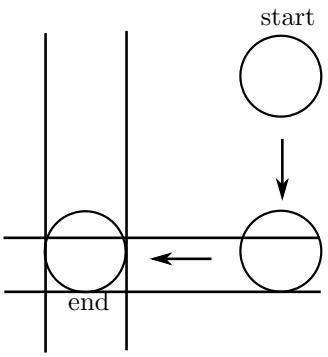


Figure 5.1: A simple example in which an arbitrary initial center moves to the desired position

Proof. Let $a_i \subseteq [d]$ be the set of the indices of the coordinates in which l_i has fixed values. $|a_i| = d - \Delta$ for each i. Since we assumed that there is no coordinate in which every flat does not have a fixed value, the union of a_i over all $1 \leq i \leq n$ is $[d]$. Let $c^* \in C^*$ be an arbitrary optimal center. Without loss of generality, let l_i be the flat chosen in the ith iteration, c_i be the current center after the ith iteration, and $b_i = \cup_{j=1}^{i} a_i$. In other words, b_i is the set of coordinates in which the current center has moved at least once. Let $d_{a_i}(p,q) = d_{l_i^\perp}(p,q) = \sqrt{\sum_{j \in a_i}(p_j - q_j)^2}$. d_{b_i} is also defined similarly. We classify each ith iteration into one of two categories depending on whether $a_i \subseteq b_{i-1}$ or not. If ith iteration belongs to the former, the analysis of Algorithm 2, applied to the case where we only consider the coordinates in $b_{i-1} = b_i$, shows that $(d_{b_i}(c_i, c^*))^2$ is decreased by at least $\Omega(\frac{R^*}{\epsilon^2})$ from $(d_{b_{i-1}}(c_{i-1}, c^*))^2$. Otherwise, the same analysis cannot be applied since $b_{i-1} \neq b_i$, but the fact that $(d_{a_i}(c_i, c^*)) \leq 2R^*$ shows that

$$\begin{aligned}
(d_{b_i}(c_i, c^*))^2 &= \sum_{j \in a_i} ((c_i)_j - (c^*)_j)^2 + \sum_{j \in b_{i-1} - a_i} ((c_i)_j - (c^*)_j)^2 \\
&\leq \sum_{j \in a_i} ((c_i)_j - (c^*)_j)^2 + \sum_{j \in b_{i-1} - a_i} ((c_{i-1})_j - (c^*)_j)^2 \\
&\leq 4(R^*)^2 + (d_{b_{i-1}}(c_{i-1}, c^*))^2
\end{aligned}$$

Therefore, $(d_{b_i}(c_i, c^*))^2$ is increased by at most $O((R^*)^2)$ from $(d_{b_{i-1}}(c_{i-1}, c^*))^2$. The first iteration makes $|b_1| = |a_1| = d - \Delta$, so there are at most $\Delta + 1$ iterations of the second category and $(d_{b_i}(c_i, c^*))^2$ is increased at most by $O(\Delta(R^*)^2)$ totally during the algorithm. Since each iteration of the first category decreases $(d_{b_i}(c_i, c^*))^2$ by $\Omega(\frac{R^*}{\epsilon^2})$, and there are at most $O(\frac{\Delta}{\epsilon^2})$ iterations of the first category. Therefore the total number of iterations is $O(\Delta(\frac{1}{\epsilon^2} + 1))$.

□

Combined with the technique used in [6], where we pick a far point from the current centers and *guess* the cluster to which it belongs, the above theorem enables us to find a $(1+\epsilon)$-approximate clustering when we know the optimal radius $R^*(k, \mathcal{L})$.

Theorem 5.2. *If all flats are axis-parallel and R^* is given, a $(1+\epsilon)$-approximate k-clustering can be found in time $2^{O(\Delta k \log k(1+\frac{1}{\epsilon^2}))} nd$.*

Proof. Let $\mathcal{C}^* = ({\mathcal{L}^*}_1, ..., {\mathcal{L}^*}_k, c_1^*, ..., c_k^*)$ be an optimal clustering. We start from the centers $c_1, ..., c_k$ at arbitrary positions.

The algorithm picks l_i that maximizes the minimum distance to each $c_1, ..., c_k$. If $d(l_i, c_j) \leq (1+\epsilon)R^*$ for every j then $\cup_{j \in [k]} B(c_j, (1+\epsilon)R^*)$ intersects every flat in $\mathcal{L}$ and we are done. Otherwise, exhaustively guess the jth cluster such that $d(c_j, l_i) > (1+\epsilon)R^*$ and move c_j to l_i as in Algorithm 2. If we guess correctly, $d(l_i, c_j) \leq (1+\epsilon)R^*$ for each $l_i \in {\mathcal{L}^*}_j$ in $O(\Delta(1 + \frac{1}{\epsilon^2}))$ iterations where jth cluster is chosen. Therefore, we need at most $O(k\Delta(1+\frac{1}{\epsilon^2}))$ iterations total. Each iteration involves

guessing between at most k clusters and can be implemented in time $O(nd)$. Therefore, the total running time is $2^{O(\Delta k \log k(1+\frac{1}{\epsilon^2}))}nd$. □

As in the 1-center problem, guessing the optimal radius R^* can be done by using binary search. Finding a meaningful upper bound on R^* uses the main algorithm with the radius 0.

Theorem 5.3. *If all flats are axis-parallel, running the above clustering algorithm with the guessed optimal radius zero will find the clustering with the radius at most $O(\sqrt{\Delta}R^*)$ in time $2^{O(\Delta k \log k)}nd$.*

Proof. As above, let l_i be the flat chosen in the ith iteration, $a_i \subseteq [d]$ be the set of the indices of the coordinates in which l_i has the fixed value. We also define $c_{i,j}$ to be the center of the jth cluster after the ith iteration, and $b_{i,j}$ to be the union of a_k's $(1 \leq k \leq i)$ that belong to the jth cluster. For any pair of (i, j) such that $b_{i,j} \neq \emptyset$ (i.e. at least one flat is classified to the jth cluster), let $l_k(k \leq i)$ be the last flat to which the center of the jth cluster has moved. As the above proofs, $d_{a_k}(c_{i,j}, c_j^*) \leq R^*$ and for each coordinate q in $b_i - a_k$ (at most Δ of them), $|(c_{i,j})_q - (c_j^*)_q| \leq R^*$. Therefore, $(d_{b_{i,j}}(c_{i,j}, c_j^*))^2 \leq \Delta(R^*)^2$.

In the ith iteration, let j be the cluster that l_i belongs to in the optimal clustering. If $a_i \subseteq b_{i-1,j}$, $d(l_i, c_{i-1,j}) = d_{b_{i-1,j}}(l_i, c_{i-1,j}) \leq d_{b_{i-1,j}}(l_i, c_j^*) + d_{b_{i-1,j}}(c_j^*, c_{i-1,j}) \leq O(\sqrt{\Delta}R^*)$. Since l_i is chosen as the flat that maximizes the minimum distance to any current center, $d(l_i, c_{i-1,j}) \leq O(\sqrt{\Delta}R^*)$ means that the distance from any flat to its nearest center is $O(\sqrt{\Delta}R^*)$. Otherwise, the size of $b_{i,j} = b_{i-1,j} \cup a_i$ grows strictly. Since the first classification makes $|b_{i,j}| = d - \Delta$, there are only $\Delta + 1$ iterations for one cluster to grow $b_{i,j}$ strictly. Therefore, in $O(k\Delta)$ iterations, we are guaranteed to have a situation where the radius of the current clustering is $O(\sqrt{\Delta}R^*)$. Each iteration involves guessing the correct cluster that l_i belongs to, so the total running time is $2^{O(\Delta k \log k)}nd$. □

Therefore, we need $O(\frac{\Delta}{\epsilon})$ trials of binary search to guess the optimal radius with error $< \epsilon$. Guessing the optimal radius and trying for different values of the optimal radius does not increase the asymptotic complexity of the main algorithm, so the total running time is $2^{O(\Delta k \log k(1+\frac{1}{\epsilon^2}))}nd$. Conversely, we have shown that it is impossible to have any approximation algorithm whose running time is polynomial in even one of k or Δ. The linear dependence on n and d suggests that this algorithm can be practical for small values of k and Δ, which usually are much less than n and d.

Chapter 6

2-Clustering of axis-parallel flats

Section 4 shows that for fixed $k \geq 2$ it is impossible to have any approximation algorithm in the general case. When all flats are restricted to be axis-parallel, this threshold is increased by 1. If $k = 2$, the lower bound of the approximation ratio for the axis-parallel case shown in Theorem 4.3 is $\frac{2}{\sqrt{3}}$, which is quite different from the other cases. This leads to the natural conjecture that there is a polynomial time(in n, d, Δ) approximation algorithm for the axis-parallel 2-center problem. One reason that makes this conjecture more plausible is the relationship between the problems we reduced to prove the hardness results of our problem. The hardness of 2-clustering in the general case is shown by the reduction from the HYPERGRAPH 2-COLORING problem, and the hardness of 3-clustering in the axis-parallel case is shown by the reduction from the GRAPH 3-COLORING problem. In these reductions, there is a natural correspondence between the number of colors/the number of centers, hypergraphs/non-axis parallel flats, and regular graphs/axis-parallel flats. Therefore, that GRAPH 2-COLORING is in P motivates us to find an approximation algorithm for this special axis-parallel 2-clustering problem, whose existence will show the fundamental difference between the axis-parallel case and the general case in terms of approximability.

In fact, we can find a polynomial time approximation algorithm for the axis-parallel 2-clustering problem by solving a problem similar to GRAPH-2-COLORING as a subroutine. To do this, we need another Helly-type theorem which holds specifically for axis-parallel flats. It extends the relatively trivial fact that pairwise nonempty intersection of some set of axis-parallel flats implies all of them intersect. Similarly, if any pairwise distance of some set of flats is small, there is a small intersecting ball.

Theorem 6.1. *Let $\mathcal{L} = \{l_1, ..., l_n\}$ be the set of axis-parallel flats, where each l_i is of dimension $\Delta_i < d$. If $d(l_i, l_j) \leq r$ for every $1 \leq i, j \leq n$, $R^*(\mathcal{L}) < 2d^{\frac{1}{4}}r$. In other words, if any pair of* slabs $\left\{l_i + B_{O,\frac{r}{2}}\right\}$ *intersect, then all slabs $\left\{l_i + B_{O,d^{\frac{1}{4}}r}\right\}$ intersect where $+$ denotes the Minkowski sum.*

Proof. As usual, without loss of generality assume that every jth coordinate is nontrivial, since we can discard trivial coordinates without affecting any distance. m_j and M_j, the smallest and the

largest fixed value for the jth coordinate, are defined for each j. Let $r' = \max_j (M_j - m_j)$ be the longest edge of the hypercube $[m_1, M_1] \times ... \times [m_d, M_d]$. It is clear that $r \geq r'$.

Let $\Delta = \min_i \Delta_i$. Assume without loss of generality $\dim(l_1) = \Delta$ and l_1 has the fixed value in $1, ..., (d-\Delta)$th coordinate. If $\Delta \geq d - p$ for some p, $c = (\frac{m_1+M_1}{2}, ..., \frac{m_d+M_d}{2})$ makes $d(c, l_i) \leq \frac{\sqrt{d-\dim(l_i)}r'}{2} \leq \frac{\sqrt{p}r}{2}$ for each i, so $R^*(\mathcal{L}) \leq \frac{\sqrt{p}r}{2}$.

If $\Delta < d - p$, fix $(c)_j = (l_1)_j$ for $j = 1, ..., d - \Delta$. $(d_{[d-\Delta]}(c, l_i))^2 \leq (d(l_1, l_i))^2 \leq r^2$ for all i. Since we have fixed the first $d - \Delta$ coordinates and $d_{[d]-[d-\Delta]}(l_i, l_j) \leq d(l_i, l_j) \leq r$, we can apply the same argument recursively where the dimension of the entire space is $\Delta < d - p$. Therefore, $(R^*(\mathcal{L}))^2 \leq r^2 + (R^*(\mathcal{L}'))^2$ where $\mathcal{L}' = \{l'_2, ..., l'_n\}$ and $l'_i \in \mathbb{R}^\Delta$ is the projection of l_i to $l_1 (1 < i \leq n)$.

Let $p = \left\lceil \sqrt{d} \right\rceil$ and prove the theorem by the induction on the underlying dimension d. When $d = 1$ it holds trivially. When the theorem is true for every $1, ..., d-1$, $R^*(\mathcal{L}) \leq \frac{\sqrt{p}r}{2} \leq 2d^{\frac{1}{4}}r$ or $(R^*(\mathcal{L}))^2 \leq r^2 + (R^*(\mathcal{L}'))^2$ where $\mathcal{L}'$ is the set of flats in the Euclidean space of dimension $d - p \leq d - \sqrt{d}$ with the smaller pairwise distance. By the induction hypothesis,

$$\begin{aligned}(R^*(\mathcal{L}))^2 &\leq r^2 + 4r^2\sqrt{d - \sqrt{d}} \\ &= (1 + 4\sqrt{d - \sqrt{d}})r^2 \\ &\leq 4r^2\sqrt{d}\end{aligned}$$

since $(1 + 4\sqrt{d - \sqrt{d}})^2 = (1 + 8\sqrt{d - \sqrt{d}} + 16(d - \sqrt{d})) \leq 16d$. Therefore, the induction is complete and $R^*(\mathcal{L}) < 2d^{\frac{1}{4}}r$.

□

The next theorem shows that this Helly-type theorem for axis-parallel flats is tight in terms of the radius of an intersecting ball.

Theorem 6.2. *There is a set of axis-parallel flats $\mathcal{L} = \{l_1, ..., l_n\}$ in $\mathbb{R}^d$ such that $d(l_i, l_j) \leq r$ for every $1 \leq i, j \leq n$, and $R^*(\mathcal{L}) = \Omega(d^{\frac{1}{4}})r$.*

Proof. Let $d = (n-1)n$ and partition the coordinates into n blocks each of which consists of $(n-1)$ coordinates. For $1 \leq i \neq j \leq n$, define

$$p(i, j) = \begin{cases} j, & \text{if } j < i \\ j - 1, & \text{if } j > i \end{cases}$$

For $1 \leq i \leq n$ and $1 \leq j \leq n(n-1)$, let $q = \left\lfloor \frac{j-1}{n-1} \right\rfloor + 1$ indicate the block of the coordinates to which j belongs. Then $(l_i)_j$ is decided by

$$(l_i)_j = \begin{cases} 0, & \text{if } i = q \\ 1, & \text{i} \neq q \text{ and } j - (q-1)(n-1) = p(q,i) \\ \sim, & \text{otherwise} \end{cases}$$

In other words, (l_i) has all 0's in the ith block and exactly one 1 in each of the other blocks, but the position of 1 in each block is different for each flat. Obviously, each pairwise distance is $r = \sqrt{2}$, since the only two coordinates where l_i and l_j are fixed and differ are the position in the ith block where l_j has 1 and the position in the jth block where l_i has 1.

Let $c = (c_1, ..., c_d) \in \mathbb{R}^d$ be an arbitrary center. For each coordinate j, there is exactly one flat that has 1 in this coordinate and exactly one flat that has 0 in this coordinate. Therefore, $\sum_{1 \leq i \leq n}((l_i)_j - (c_j))^2 \geq \frac{1}{2}$ with equality if and only if $c_j = \frac{1}{2}$. Summing over all coordinates, $\sum_{1 \leq i \leq n}(d(c, l_i))^2 \geq d/2 = \frac{n(n-1)}{2}$ with equality if $c = (\frac{1}{2}, ..., \frac{1}{2})$. Therefore, $\max_{1 \leq i \leq n}(d(c, l_i))^2 \geq \frac{n-1}{2}$ so $R^*(\mathcal{L}) \geq \sqrt{\frac{n-1}{2}} = \frac{\sqrt{n-1}}{2} r = \Omega(d^{\frac{1}{4}})r$.

□

Equipped with this new Helly-type Theorem 6.1, we give a polynomial time approximation algorithm for the axis-parallel 2-clustering problem. Note that $d(l_i, l_j) \leq 2R^*$ when l_i and l_j belong to the same cluster in the optimal clustering. Therefore, we can construct a graph based on the pairwise distances, use a 2-COLORING algorithm to get two clusters; Within each cluster, Theorem 6.1, applied to the flats projected to one of them, ensures that they can be covered by a ball of radius $O(\Delta^{1/4})R^*$.

Theorem 6.3. *If all flats are axis-parallel, an $O(\Delta^{\frac{1}{4}})$-approximate 2-clustering can be found in time $O(dn^2 \log n)$.*

Proof. Note that $d(l_i, l_j) \leq 2R^*$ when l_i and l_j belong to the same cluster in the optimal clustering. Let r be the maximum $d(l_i, l_j)$ that is not larger than $2R^*$. Construct a graph $G = (\mathcal{L}, E)$ where $(l_i, l_j) \in E$ if and only if $d(l_i, l_j) > r$. Then G is 2-colorable since there is one corresponding to the optimal clustering. Find a 2-coloring; each color represents a cluster so that $d(l_i, l_j) \leq r$ for any i, j in the same cluster.

Without loss of generality, let $l_1, ..., l_{n'}$ belong to the same cluster. Let $l'_{1,j}$ be l_j projected into l_1. For any $1 < j, k \leq n'$, $d_{l_1}(l'_{1,j}, l'_{1,k}) = d_{l_1}(l_j, l_k) \leq d(l_j, l_k) \leq 2R^*$. By Theorem 6.1 applied to l_1 (of dimension Δ), there exists $c \in l_1$ such that $d_{l_1}(c, l_j) = d_{l_1}(c, l'_{1,j}) \leq O(\Delta^{\frac{1}{4}})R^*$ for any $1 \leq j \leq n'$. Since $d_{l_1^\perp}(c, l_j) = d_{l_1^\perp}(l_1, l_j) \leq 2R^*$, so $d(c, l_j) \leq O(\Delta^{\frac{1}{4}})R^*$ for any $1 \leq j \leq n'$. This analysis also works for the other cluster, so we found an $O(\Delta^{\frac{1}{4}})$-approximate 2-clustering.

Finding r can be done by binary search in the set of all pairwise distances; it can be done in $O(\log n)$. For each r, the most expensive step is to construct G, which takes $O(n^2 d)$. Therefore, the

total running time is $O(dn^2 \log n)$.

□

Chapter 7

Conclusions and Future Work

We have presented the algorithms and hardness results for clustering affine subspaces. Our main algorithm for general k and Δ and hardness results almost match in the sense that we proved that the exponential dependence on k and Δ is inevitable and suggested an algorithm which runs in time exponential in k and Δ but linear in n and d. Furthermore, assuming the exponential time hypothesis, the exponent linear in Δ (for fixed k) and quasi-linear in k (for fixed Δ) is almost tight because we cannot expect a subexponential(for any of k and Δ) algorithm in the general case; the axis-parallel case only relaxes the dependence on k from $2^{o(k)}$ to $2^{o(\sqrt{k})}$. Therefore, other than trying to find a subexponential (in k) algorithm for the axis-parallel case, one immediate improvement of the running time will be to reduce the current quasi-multilinear exponent $2^{O(\Delta k \log k)}$ to a quasi-linear one like $2^{O(\Delta + k \log k)}$. If this improvement is achieved, it will be nearly the theoretically fastest algorithm. Conversely, there might be another reduction that, by considering general k and Δ simultaneously, proves it is impossible to have an algorithm which runs in time $2^{o(\Delta k)}$; in this case, our algorithm will nearly match the lower bound. Since our algorithms are all deterministic and do not assume instances are easily clusterable, it is also tempting to try to exploit the power of randomness or to assume *well-clusterability* or *stability* introduced in[27, 4, 3, 22] on the instances to improve the running time.

All of our algorithms can be applied in the general case and proved to converge to an optimal clustering; it is the theoretical upper bound on the number of iterations that can be proved only in the axis-parallel case. The key downside of not having axis-parallel flats is that it is hard to find an initial center close to the optimal center; given initial centers within distance $\sqrt{\Delta}R^*$ from the corresponding optimal centers, our iterative algorithms in Section 3 and 5 will converge with the same upper bound on the number of iterations proved in this paper. Such initial centers were found in the big axis-parallel hypercube containing any intersection of flats, or by using a few steps of the main iterative algorithms where the distance is bounded in each coordinate, but these methods cannot be used when flats are not axis-parallel. It would be meaningful to devise new techniques (e.g. scaling the space) so that our methods work for the general case.

Bibliography

[1] P. D. Allison. Missing data. *Sage Publications*, 2002.

[2] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 21–29, New York, NY, USA, 2001. ACM.

[3] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Stability yields a ptas for k-median and k-means clustering. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 309–318, Washington, DC, USA, 2010. IEEE Computer Society.

[4] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 1068–1077, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[5] Mihai Bădoiu and Kenneth L. Clarkson. Optimal core-sets for balls. *Comput. Geom. Theory Appl.*, 40(1):14–22, May 2008.

[6] Mihai Bădoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 250–257, New York, NY, USA, 2002. ACM.

[7] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem (extended abstract). In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, STOC '99, pages 1–10, New York, NY, USA, 1999. ACM.

[8] Sanjoy Dagupta. The hardness of k-means clustering. In *Technical report CS-2007-0890*, University of California at San Diego, 2008.

[9] W. Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation schemes for clustering problems. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, STOC '03, pages 50–58, New York, NY, USA, 2003. ACM.

[10] M. Effros and L. J. Schulman. Deterministic clustering with data nets. In *ECCC*, pages TR04–050, 2004.

[11] Tomás Feder and Daniel Greene. Optimal algorithms for approximate clustering. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 434–444, New York, NY, USA, 1988. ACM.

[12] J. Flum and M. Grohe. Parameterized complexity theory. *Springer*, 2006.

[13] Jie Gao, Michael Langberg, and Leonard Schulman. Analysis of incomplete data and an intrinsic-dimension helly theorem. *Discrete and Computational Geometry*, 40:537–560, 2008. 10.1007/s00454-008-9107-5.

[14] Jie Gao, Michael Langberg, and Leonard J. Schulman. Clustering lines in high-dimensional space: Classification of incomplete data. *ACM Trans. Algorithms*, 7(1):8:1–8:26, December 2010.

[15] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified np-complete graph problems. *Theoretical Computer Science*, 1(3):237 – 267, 1976.

[16] Ashish Goel, Piotr Indyk, and Kasturi Varadarajan. Reductions among high dimensional proximity problems. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, SODA '01, pages 769–778, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[17] Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38(0):293 – 306, 1985.

[18] Sudipto Guha and Samir Khuller. Greedy strikes back: improved facility location algorithms. In *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, SODA '98, pages 649–657, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[19] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 291–300, New York, NY, USA, 2004. ACM.

[20] R. Impagliazzo and R. Paturi. Complexity of k-sat. In *Computational Complexity, 1999. Proceedings. Fourteenth Annual IEEE Conference on*, pages 237 –240, 1999.

[21] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 731–740, New York, NY, USA, 2002. ACM.

[22] A. Kumar and R. Kannan. Clustering with spectral norm and the k-means algorithm. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 299 –308, oct. 2010.

[23] A. Kumar, Y. Sabharwal, and S. Sen. A simple linear time $(1+\epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 454 – 462, oct. 2004.

[24] S. Lloyd. Least squares quantization in PCM. *Information Theory, IEEE Transactions on*, 28(2):129 – 137, mar 1982.

[25] J. Matoušek. On approximate geometric k -clustering. *Discrete and Computational Geometry*, 24:61–84, 2000.

[26] R. Ostrovsky and Y. Rabani. Polynomial time approximation schemes for geometric k-clustering. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 349 –358, 2000.

[27] Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. In *Foundations of Computer Science, 2006. FOCS '06. 47th Annual IEEE Symposium on*, pages 165 –176, oct. 2006.

[28] Rina Panigrahy. Minimum enclosing polytope in high dimensions. *CoRR*, cs.CG/0407020, 2004.

[29] E. Alper Yildirim. Two algorithms for the minimum enclosing ball problem. *SIAM J. on Optimization*, 19(3):1368–1391, November 2008.