Approximate Clustering via Core-Sets

Mihai Bădoiu[*] Sariel Har-Peled[†] Piotr Indyk[‡]

22/02/2002 17:40

Abstract

In this paper, we show that for several clustering problems one can extract a small set of points, so that using those *core-sets* enable us to perform approximate clustering efficiently. The surprising property of those core-sets is that their size is independent of the dimension.

Using those, we present a $(1+\varepsilon)$-approximation algorithms for the k-center clustering and k-median clustering problems in Euclidean space. The running time of the new algorithms has linear or near linear dependency on the number of points and the dimension, and exponential dependency on $1/\varepsilon$ and k. As such, our results are a substantial improvement over what was previously known.

We also present some other clustering results including $(1+\varepsilon)$-approximate 1-cylinder clustering, and k-center clustering with outliers.

1 Introduction

Clustering is one of the central problems in computer-science. It is related to unsupervised learning, classification, databases, spatial range-searching, data-mining, etc. As such, it received a lot of attention in computer-science in the last twenty years. There is a large literature on this topic with numerous variants, see [DHS01, BE97].

In this paper, we present several results on clustering of a set of points P in $\mathbb{R}^d$, where the dimension d is large. All our results rely on a new technique that extract a small subset of points that "represents" this point-set ε-well as far as specific clustering problems are concerned. The surprising property of those sets is that their size is *independent* of the dimension. The existence of such core-sets for various approximation problems was known before, but their size depended polynomially or exponentially on the dimension [MOP01, ADPR00, Har01, HV01, IT00].

Using this new technique, we present the following results in this paper:

[*]MIT Laboratory for Computer Science; 545 Technology Square, NE43-371; Cambridge, Massachusetts 02139-3594; mihai@theory.lcs.mit.edu

[†]Department of Computer Science, DCL 2111; University of Illinois; 1304 West Springfield Ave.; Urbana, IL 61801; USA; http://www.uiuc.edu/~sariel/; sariel@uiuc.edu

[‡]MIT Laboratory for Computer Science; 545 Technology Square, NE43-373; Cambridge, Massachusetts 02139-3594; indyk@theory.lcs.mit.edu

- In Section 2, we show that one can extract a core-set of size $O(1/\varepsilon^2)$, so that the minimum enclosing ball of the sample is an $(1+\varepsilon)$-approximation to the minimum enclosing ball of a set of points $P \subset \mathbb{R}^d$. Using this core-set, we present a $2^{O((k \log k)/\varepsilon^2)} \cdot dn$ time algorithm that computes an $(1+\varepsilon)$-approximate k-center clustering of P, i.e., finds a set of k points in $\mathbb{R}^d$ such that the maximum distance of points in P to their closest center is minimized. No algorithm for this problem has been previously known, although (as pointed out in [Ind01]) by using the techniques of [OR00] one could achieve a much slower algorithm with running time $n^{O(k^2/\varepsilon^2)}$. For lower dimension, Agarwal and Procopiuc [AP98] showed a $O(n \log k) + (k/\varepsilon)^{O(dk^{1-1/d})}$ time algorithm for this problem.

  For $k = 1$, the core-set technique yields an algorithm with running time $O\left(dn/\varepsilon^2 + (1/\varepsilon)^{O(1)}\right)$. This significantly improves previously known bounds (obtained via ellipsoid algorithm) of $O(d^3 n \log(1/\varepsilon))$ [GLS88].

- In Section 3, we show that by using random sampling, one can find an $O(1/\varepsilon^{O(1)})$-size set of points R, such that the flat spanned by those points contains a point ε-close to the 1-median of the point-set. The only previous result of this type [IT00] used a sample of size linearly dependent on the dimension. Using the sampling technique, we present a

$$2^{(k/\varepsilon)^{O(1)}} d^{O(1)} n \log^{O(k)} n$$

  expected time algorithm that computes a $(1+\varepsilon)$-approximation to the optimal k-median for P (i.e., finds k points-medians in $\mathbb{R}^d$, such that the sum of the distances from all points P to their closest medians is minimized). Previously, the fastest known algorithm with polynomial dependence on d was due to Ostrovsky and Rabani [OR00] and it ran in $n^{(k+1/\varepsilon)^{O(1)}}$ time. For relevant results, see [ARR98].

- In Section 5, we present an $(1+\varepsilon)$-approximation algorithm for the problem of computing the minimum radius cylinder that covers P. The running time of the new algorithm is $n^{O(\log(1/\varepsilon)/\varepsilon^2)}$. The fastest algorithm previously known run in $O(n + 1/\varepsilon^{O(d)})$ time [HV01], which is exponential in the dimension. The algorithm combines the use of a core-set for 1-center, dimensionality reduction (to reduce the search space, in a manner similar to [OR00]) and convex programming.

- Section 4 we present an efficient algorithm for solving k-center problem with *outliers*.

Concluding remarks are given in Section 6.

2 k-center clustering

In this section, we present an efficient approximation algorithm for the k-center problem.

Definition 2.1 For a point-set P in $\mathbb{R}^d$, let $r_{cen}(P,k)$ denote the radius of the k-center clustering of P. Here one wishes to find the k centers (i.e., points), so that the maximum distance of a point to a center is minimized. This distance the radius of the clustering.

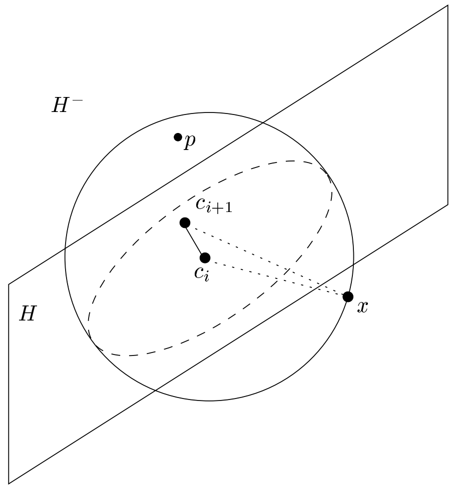

Figure 1: If the two centers c_{i+1}, c_i are far, then the radius of the min-enclosing ball must grow.

We start from restating the following lemma, proved originally in [GIV01]. For completeness, we give the proof here.

Lemma 2.2 *Let $B = \mathrm{Ball}(c_B, r)$ be a minimum enclosing ball of a point-set $P \subseteq \mathbb{R}^d$, then any closed half-space that contains the center of B, must also contain at least a point from P that is at distance r from the center of B.*

Proof: Suppose there exists a closed half-space H that contains the center of B and does not contain any point of P of distance r from the center c_B. Since H is closed, there exist an $\varepsilon > 0$ such that the minimum distance between the points of $P \setminus H$ and H is $> \varepsilon$. Also, fix ε such that the distance between any of the points in $P \bigcap H$ and c_B is at most $r - \varepsilon$. This means we can translate the ball B in the direction perpendicular to H by $\varepsilon/2$. After we translate B, none of the points of P will lie exactly on the boundary of the translated ball, which means we can shrink the ball radius by δ and we have found a smaller ball that contains our point-set. A contradiction. ■

Now we proceed with the core-set result.

Lemma 2.3 *There exist a subset of points $S \subseteq P$, $|S| = O(1/\varepsilon^2)$, such that the radius of the minimum enclosing ball of S is at least $1/(1+\varepsilon)$ the minimum enclosing ball of P.*

Proof: Start with an arbitrary point $x \in P$ and let y be the furthest point in P away from x. Clearly, $||x - y|| \geq \Delta/2$, where Δ denotes the diameter of P.

Set $S_0 = \{x, y\}$. In the following, we maintain a set S_i of points and their minimum enclosing ball $B_i = \mathrm{Ball}(c_i, r_i)$ of S_i, where c_i, r_i denotes the center and radius of B_i, respectively. Clearly, $r_0 \geq \Delta/4$.

There are two possibilities:

- If there is no point $p \in P$, such that $||p - c_i|| \geq (1+\varepsilon)r_i$, then we are done, as the current enclosing ball B_i is a $(1+\varepsilon)$-approximation.

- There exist a point $p \in P$, such that $||p - c_i|| \geq (1+\varepsilon)r_i$

  In this case we set $S_{i+1} = S_i \cup \{p\}$.

Claim 2.4 $r_{i+1} \geq \left(1 + \frac{\varepsilon^2}{16}\right) r_i$.

Proof: If $\|c_i - c_{i+1}\| < (\varepsilon/2) r_i$, then by the triangle inequality, we have

$$\|p - c_{i+1}\| \quad \geq \quad \|p - c_i\| - \|c_i - c_{i+1}\| \geq (1+\varepsilon) r_i - \frac{\varepsilon}{2} r_i = \left(1 + \frac{\varepsilon}{2}\right) r_i.$$

Otherwise, if $\|c_i - c_{i+1}\| \geq (\varepsilon/2) r_i$ then let H be the $(d-1)$-dimensional hyperplane that passes through c_i and is orthogonal to $c_i c_{i+1}$. Let H^- be the open half-space having p inside it. See Figure 1.
Using Lemma 2.2 we know that there exist a point x at distance r_i from c_i that is not in H^-. Therefore

$$r_{i+1} \geq \|c_{i+1} - x\| \geq \sqrt{r_i^2 + \frac{\varepsilon^2}{4} r_i^2} \geq \left(1 + \frac{\varepsilon^2}{16}\right) r_i,$$

as $0 < \varepsilon < 1$. ■

Since $r_0 \geq \Delta/4$, and at each step we increase the radius of our solution by at least $(\Delta/4)\varepsilon^2/16 = \Delta\varepsilon^2/64$, it follows that we cannot encounter this case more than $64/\varepsilon^2$ times, as Δ is an upper bound of the radius of the minimum enclosing ball of P. ■

Theorem 2.5 *For any point-set $P \subset \mathbb{R}^d$ and $1 > \varepsilon > 0$, there is a subset $S \subset P$, $|S| = O(1/\varepsilon^2)$, such that if o is the 1-center for S, then o is a $(1+\varepsilon)$-approximate 1-center for P. The set S can be found in time $O(dn/\varepsilon^2 + (1/\varepsilon)^{10} \log(1/\varepsilon))$.*

Proof: The algorithm follows the proof of Lemma 2.3. This requires computing $M = O(1/\varepsilon^2)$ times $(1+\varepsilon)$-approximate enclosing ball of at most $N = O(1/\varepsilon^2)$ points in $D = O(1/\varepsilon^2)$ dimensions. This can be done in $O(MD^3 N \log(1/\varepsilon)) = O(1/\varepsilon^{10} \log(1/\varepsilon))$ time, using convex programming techniques [GLS88]. This also requires scanning the points M times, which takes $O(nd/\varepsilon^2)$ time overall. ■

Theorem 2.6 *For any point-set $P \subset \mathbb{R}^d$ and $1 > \varepsilon > 0$, a $(1+\varepsilon)$-approximate 2-center for P can be found in $2^{O(1/\varepsilon^2)} dn$ time.*

Proof: We start from two empty sets of points S_1, S_2. At each stage let B_1, B_2 denote the smallest enclosing ball for S_1 and S_2. In the i-th iteration, we pick the point p_i furthest away from B_1 and B_2. To decide whether to put p_i in S_1 or in S_2, we read a bit from a guessing oracle. Clearly, by Theorem 2.5, after $O(1/\varepsilon^2)$ iterations we would be done, assuming our guessing oracle, classified the required points correctly. Thus, the running time of this algorithm is $O(dn/\varepsilon^2 + (1/\varepsilon)^{10})$.

To remove the guessing oracle, we exhaustively enumerate all possible guesses. This would require running this algorithm $2^{O(1/\varepsilon^2)}$ times, once for each guess sequence. Overall, resulting in $dn 2^{O(1/\varepsilon^2)}$ running time. ■

Theorem 2.7 *For any point-set $P \subset \mathbb{R}^d$ and $1 > \varepsilon > 0$, a $(1+\varepsilon)$-approximate k-center for P can be found in $2^{O((k \log k)/\varepsilon^2)} dn$ time.*

Proof: Follows by a straightforward extension of the algorithm of Theorem 2.6, where each guess now is a number between 1 and k, and we have to generate $O(k/\varepsilon^2)$ guesses. ■

3 k-median clustering

In this section, we present an efficient approximation algorithm for the k-median problem.

Definition 3.1 For a set X and a point y in $\mathbb{R}^d$, let $\mathrm{dist}(X,p) = \min_{x\in X} \|x-p\|$. For a set P of n points in $\mathbb{R}^d$, let

$$\mathrm{med}_{\mathrm{opt}}(P,k) = \min_{K\subseteq\mathbb{R}^d, |K|=k} \sum_{p\in P} \mathrm{dist}(K,p)$$

denote the optimal price of the k-median problem. Let

$$\mathrm{AvgMed}(P,k) = \mathrm{med}_{\mathrm{opt}}(P,k)/|P|$$

denote the average radius of the k-median clustering.

For any sets $A,B \subset P$, we use the notation

$$\mathrm{cost}(A,B) = \sum_{a\in A, b\in B} \|a-b\|.$$

If $A = \{a\}$, we write $\mathrm{cost}(a,\cdot)$ instead of $\mathrm{cost}(\{a\},\cdot)$; similarly for b. Moreover, we define $\mathrm{cost}(x \vee y, A) = \sum_{a\in A} \min(\|a-x\|, \|a-y\|)$.

For a set of points $X \subseteq \mathbb{R}^d$, let $\mathrm{span}(X)$ denote the affine subspace spanned by the points of X. We refer to $\mathrm{span}(X)$ as the *flat* spanned by X.

Theorem 3.2 *Let P be a point-set in $\mathbb{R}^d$, $1 > \varepsilon > 0$, and let X be a random sample of $O(1/\varepsilon^3 \log 1/\varepsilon)$ points from P. Then with constant probability, the following two events happen: (i) The flat* $\mathrm{span}(X)$ *contains a $(1+\varepsilon)$-approximate* 1*-median for P, and (ii) X contains a point in distance* $\leq 2\,\mathrm{AvgMed}(P,1)$ *from the center of the optimal solution.*

Proof: Let $\mathrm{med}_{\mathrm{opt}} = \mathrm{med}_{\mathrm{opt}}(P,1)$ be the price of the optimal 1-median, $R = \mathrm{AvgMed}(P,1)$, $\beta = \varepsilon/16$, and $s_1,\ldots,s_u$ be our random sample. In the following, we are going to partition the random sample into rounds: A round continues till we sample a point that has some required property. The first round continues till we encounter a point s_i, such that $\|s_i - c_{opt}\| \leq 2R$, where c_{opt} is the center of the optimal 1-median. By the Markov inequality, for any sample s_i we have $\|s_i - c_{opt}\| \leq 2R$, with probability $\geq 1/2$, as $E[\|s_i - c_{opt}\|] = R$.

Let's assume that s_i is a sample that just terminated a round, and we start a new sampling round, and F_i is the flat spanned by the first i points in our sample: $s_1,\ldots,s_i$. Observe that if $\mathrm{dist}(F_i, c_{opt}) \leq \varepsilon R$, then we are done, as the point $\mathrm{proj}(c_{opt}, F_i)$ is the required approximation, where $\mathrm{proj}(c_{opt}, F_i)$ denote the projection of c_{opt} into F_i.

Note, that the distance from c_{opt} to F_i is monotone decreasing. That is $d_{i+1} = \mathrm{dist}(F_{i+1}, c_{opt}) \leq d_i = \mathrm{dist}(F_i, c_{opt})$. We would next argue that either after taking enough sample points, d_i is small enough so that we can stop, or otherwise almost all the points of P lie very close to our spanned subspace, and we can use it to find our solution.

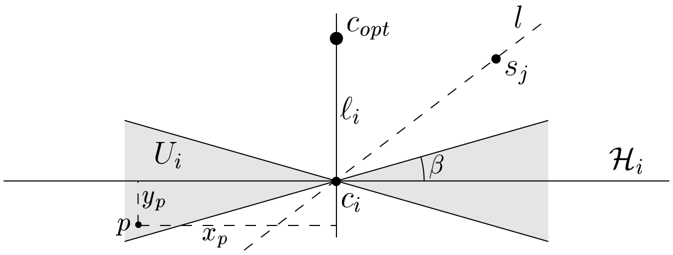

Figure 2: A round terminates as soon as we pick a point outside U_i.

Indeed, let $c_i = \mathrm{proj}(c_{opt}, F_i)$, and let

$$U_i = \left\{ x \,\middle|\, x \in \mathbb{R}^d \text{ s.t. } \pi/2 - \beta \le \angle c_{opt} c_i x \le \pi/2 + \beta \right\},$$

be the complement of the cone of angle $\pi - \beta$ emanating from c_i, and having $c_i c_{opt}$ as its axis. See Figure 2. Let $\mathcal{H}_i$ be the $(d-1)$-dimensional hyperplane passing through c_i and perpendicular to $c_i c_{opt}$. For a point $p \in P$, let x_p be the distance of p to the line ℓ_i passing through c_{opt} and c_i, and let y_p be the distance between p and $\mathcal{H}_i$.

If $p \in U_i$, then

$$y_p \le x_p \tan\beta \le x_p \frac{\sin\beta}{\cos\beta} \le 4\beta x_p \le \frac{\varepsilon x_p}{4} \le \frac{\varepsilon}{4} \left\| p - c_{opt} \right\|,$$

as $\beta < 1/16$. In particular,

$$\| p - c_i \| \le x_p + y_p \le (1 + \varepsilon/4) \left\| p - c_{opt} \right\|.$$

Namely, if we move our center from c_{opt} to c_i, the error generated by points inside U_i is smaller than $\mathrm{med}_{\mathrm{opt}} \varepsilon/4$.

Thus, if the number of points in $Q_i = P \setminus U_i$ is smaller than $n\varepsilon/4$, then we are done, as the maximum error encountered for a point of Q_i when moving the center from c_{opt} to c_i is at most $2R$.

Thus, it must be that $|Q_i| \ge n\varepsilon/4$. We now perform a round of random sampling till we pick a point that is in Q_i. Let $s_j \in Q_i$ be this sample point, where $j > i$. Clearly, the line l connecting c_i to s_j must belong to F_j, as $c_i \in F_i \subset F_j$. Now, the angle between l and $\ell_i = \mathrm{line}(c_i, c_{opt})$ is smaller than $\pi/2 - \beta$. Namely,

$$\begin{aligned} \mathrm{dist}(F_j, c_{opt}) &\le \mathrm{dist}(l, c_{opt}) \le \left\| c_{opt} - c_i \right\| \sin(\pi/2 - \beta) = \left\| c_{opt} - c_i \right\| \cos(\beta) \\ &\le (1 - \beta^2/4) \left\| c_{opt} - c_i \right\| \\ &= (1 - \beta^2/4)\, \mathrm{dist}(F_i, c_{opt}). \end{aligned}$$

Thus, after each round, the distance between F_i and c_{opt} shrinks by a factor of $(1 - \beta^2/4)$. Namely, either we are close to the optimal center, or alternatively, we make reasonable progress in each round.

In the first round, we picked a point s_u such that $\left\| s_u - c_{opt} \right\| \le 2R$. Either during our sampling, we had $\mathrm{dist}(F_i, c_{opt}) \le \varepsilon R$, or alternatively, we had reduced in each round the distance between our sample flat and c_{opt} by a factor of $(1 - \beta^2/4)$. On the other hand, once this distance drops below

εR, we stop, as we had found a point that belongs to F_i and provide a $(1+\varepsilon)$-approximate solution. Furthermore, as long as $|Q_i| \geq \varepsilon n/4$, the probability of success is at least $\varepsilon/4$. Thus, the expected number of samples in a round till we pick a point of Q_i (and thus terminating the i-th round) is $\lceil 4/\varepsilon \rceil$. The number of rounds we need is

$$M = \left\lceil \log_{1-\beta^2/4} \frac{\varepsilon}{2} \right\rceil = \left\lceil \frac{\log(\varepsilon/2)}{\log(1-\beta^2/4)} \right\rceil = O\left(\frac{1}{\varepsilon^2} \log \frac{2}{\varepsilon}\right).$$

Let X be the random variable which is the number of random samples till get M successes. Clearly, $E[X] = O(1/\varepsilon^3 \log 1/\varepsilon)$. It follows, by the Markov inequality, that with constant probability, if we sample $O(1/\varepsilon^3 \log(1/\varepsilon))$ points, then those points span a subspace that contains a $(1+\varepsilon)$-approximate 1-median center. ■

We are next interested in solving the k-median problem for a set of points in $\mathbb{R}^d$. We first normalize the point-set.

Lemma 3.3 *Given a point-set P in $\mathbb{R}^d$, and a parameters k, ε, one can can scale-up space and compute a point-set P', such that: (i) The distance between any two points in P' is at least one. (ii) The optimal k-median cost of the modified data set is at most n^b for $b = O(1)$, where $n = |P|$. (iii) The costs of any k-median solutions in both (old and modified) data sets are the same up to a factor of $(1+\varepsilon/5)$. This can be done in $O(nkd)$ time.*

Proof: Observe that by using Gonzalez [Gon85] 2-approximation algorithm to the k-center clustering, we can compute in $O(nkd)$ time a value L (the radius of the approximate k-center clustering), such that $L/2 \leq \mathrm{med}_{\mathrm{opt}}(P,k) \leq nL$.

We cover space by a grid of size $L\varepsilon/(5nd)$, and snap the points of P to this grid. After scaling, this is the required point-set. ■

From this point on, we assume that the given point-set is normalized.

Theorem 3.4 *Let P be a normalized set of n points in $\mathbb{R}^d$, $1 > \varepsilon > 0$, and let R be a random sample of $O(1/\varepsilon^3 \log 1/\varepsilon)$ points from P. Then one can compute, in $O\left(d 2^{O(1/\varepsilon^4)} \log n\right)$ time, a point-set $S(P,R)$ of cardinality $O\left(2^{O(1/\varepsilon^4)} \log n\right)$, such that with constant probability (over the choice of R), there is a point $q \in S(P,R)$ such that $cost(q,P) \leq (1+\varepsilon)\mathrm{med}_{\mathrm{opt}}(P,1)$.*

Proof: Let's assume that we had found a t such that $t/2 \leq \mathrm{AvgMed}(P,1) \leq t$. Clearly, we can find such a t by checking all possible values of $t = 2^i$, for $i = 0, \ldots, O(\log n)$, as P is a normalized point-set (see Lemma 3.3).

Next, by Theorem 3.2, we know that with constant probability, there is a point of R with distance $\leq 2\,\mathrm{AvgMed}(P,1) \leq 2t$ from the optimal 1-median center c_{opt} of P. Let $H = \mathrm{span}(R)$ denote the affine subspace spanned by R.

Note, that by Theorem 3.2, with constant probability, the flat H contains a point x such that $\mathrm{cost}(x,P) \leq (1+\varepsilon/4)\mathrm{med}_{\mathrm{opt}}(P,1)$. We next use exponential grids to find a point on H close to the (unknown) x.

For each point of $p \in R$, we construct a grid $G_p(t)$ of side length $\varepsilon t/(10|R|) = O(t\varepsilon^4 \log(1/\varepsilon))$ centered at p on H, and let $B(p,3t)$ be a ball of radius $2t$ centered at p. Finally, let $S'(p,t) = G_p(t) \cap B(p,3t)$. Clearly, if $t/2 \leq \mathrm{AvgMed}(P,1) \leq t$, and $\left\| p - c_{opt} \right\| \leq 2t$, then there is a point $q \in S'(p,t)$ such that $\mathrm{cost}(q,P) \leq (1+\varepsilon)\mathrm{med}_{\mathrm{opt}}(P,1)$.

Let $S(P,R) = \bigcup_{i=0}^{O(\log n)} \bigcup_{p \in R} S'(p, 2^i)$. Clearly, $S(P,R)$ is the required point-set, and furthermore,

$$
\begin{aligned}
|S(P,R)| &= O\left((\log n)\, |R| \left(\frac{1}{\varepsilon^4} \log \frac{1}{\varepsilon} \right)^{O(|R|)} \right) \\
&= O\left(2^{O(1/\varepsilon^3 \log^2 (1/\varepsilon))} \log n \right) = O\left(2^{O(1/\varepsilon^4)} \log n \right).
\end{aligned}
$$

■

Theorem 3.5 *For any point-set $P \subset \mathbb{R}^d$ and $0 < \varepsilon < 1$, a $(1+\varepsilon)$-approximate 2-median for P can be found in*

$$O(2^{(1/\varepsilon)^{O(1)}} d^{O(1)} n \log^{O(1)} n)$$

expected time, the results are correct with high-probability.

Proof: In the following, we assume that the solution is irreducible, i.e., removing a median creates a solution with cost at least $1 + \Omega(\varepsilon)$ times the optimal. Otherwise, we can focus on solving the 1-median instead.

Let c_1, c_2 be the optimal centers and P_1, P_2 be the optimal clusters. Without loss of generality we assume that $|P_1| \geq |P_2|$. The algorithm proceeds by considering whether P_2 is large or small when compared with the size of P_1. In both cases the algorithm return an approximate solution with constant probability. By exploring both cases in parallel and repeating the computation several times we can achieve arbitrarily large probability of success.

Case 1: $|P_1| \geq |P_2| \geq |P_1|\varepsilon$. In this case we sample a random set of points R of cardinality $O(1/\varepsilon^4 \log 1/\varepsilon)$. We now exhaustively check all possible partitions of R into $R_1 = P_1 \cap R$ and $R_2 = P_2 \cap R$ (there are $O(2^{O(1/\varepsilon^4 \log 1/\varepsilon)})$ such possibilities). For the right such partition, R_i is a random sample of points in P_i of cardinality $\Omega(1/\varepsilon^3 \log 1/\varepsilon)$ (since $E[|R \cap P_i|] = \Omega(1/\varepsilon^3 \log 1/\varepsilon)$). By Theorem 3.4, we can generate point-sets S_1, S_2 that with constant probability contains $c'_1 \in S_1, c'_2 \in S_2$, such that $\mathrm{cost}(c'_1 \vee c'_2, P) \leq (1+\varepsilon)\mathrm{med}_{\mathrm{opt}}(P, 2)$. Checking each such pair c'_1, c'_2 takes $O(nd)$ time, and we have $O(|S_1||S_2|)$ pairs. Thus the total running time is $O\left(nd 2^{O(1/\varepsilon^4 \log 1/\varepsilon)} \log^2 n\right)$.

Case 2: $|P_1|\varepsilon > |P_2|$. In this case we proceed as follows. First, we sample a set R of $\lambda = O(1/\varepsilon^3 \log 1/\varepsilon)$ points from P_1. This can be done just by sampling λ points from P, since with probability $2^{-O(1/\varepsilon^3 \log 1/\varepsilon)}$ such a sample contains only points from P_1; we can repeat the whole algorithm several times to obtain constant probability of success. Next, using Theorem 3.4, we generate a set C_1 of candidates to be center points of the cluster P_1. In the following, we check all possible centers $c'_1 \in C_1$. With constant probability, there exists $c'_1 \in C_1$ such that $\mathrm{cost}(c'_1, P_1) \leq (1+\varepsilon/3)\mathrm{cost}(c_1, P_1)$.

Let (P'_1, P'_2) denote optimal 2-median clustering induced by median c'_1 (as above), and let c'_2 denote the corresponding center of P'_2. We need to find c''_2 such that $\mathrm{cost}(c'_1 \vee c''_2, P) \leq (1 + \varepsilon/3)\mathrm{cost}(c'_1 \vee c'_2, P) \leq (1+\varepsilon)\mathrm{med}_{\mathrm{opt}}(P, 2)$. In order to do that, we first remove some elements from P_1, in order to facilitate random sampling from P_2.

First, observe that $\mathrm{cost}(c'_1, P'_2) \leq |P'_2| \cdot \|c'_2 - c'_1\| + \mathrm{cost}(c'_2, P'_2)$ and therefore we can focus on the case when $|P'_2| \cdot \|c'_2 - c'_1\|$ is greater than $O(\varepsilon) \cdot \mathrm{cost}(c'_1 \vee c'_2, P)$, since otherwise $c'_2 = c'_1$ would be a good enough solution.

We exhaustively search, for the right values of two parameters t, $\mathcal{U}$, such that $t/2 \leq \|c_1' - c_2'\| \leq t$ and $\mathcal{U}/2 \leq \mathrm{med}_{\mathrm{opt}}(P,2) \leq \mathcal{U}$. Since P is normalized this would require checking $O(\log^2 n)$ possible values for t and $\mathcal{U}$. If $t > 4\mathcal{U}$, then $t > 4\mathrm{med}_{\mathrm{opt}}(P,2)$ and for any $p,q \in P_i$ we have $\|p-q\| \leq \mathcal{U}$. Moreover, for any $p \in P_1, q \in P_2$ we have $\|p-q\| \geq \|c_1'-c_2'\| - \|c_1'-p\| - \|c_1'-q\| > 2\mathcal{U}$. Thus, take all the points in distance $\leq 2\mathcal{U}$ from c_1' to be in P_1', and take all the other points to be in P_2'. The problem is thus solved, as we partitioned the points into the correct clusters, and can compute an approximated 1-median for each one of them directly.

Otherwise, $t \leq 4\mathcal{U}$ and let $S = \left\{ p \;\middle|\; \|p - c_1'\| \leq t/4 \right\}$. Clearly, $S \subset P_1'$. Moreover, we claim that $|P_2'| \geq \varepsilon |P_1' \setminus S|$, since otherwise we would have

$$|P_2'| \left\|c_2' - c_1'\right\| \leq \varepsilon |P_1' \setminus S| \left\|c_2' - c_1'\right\|$$

and

$$\mathrm{cost}(c_1', P_1' \setminus S) \geq \frac{t}{4} |P_1' \setminus S| \geq \frac{\|c_2' - c_1'\|}{8} |P_1' \setminus S|.$$

Thus, $|P_2'| \, \|c_2' - c_1'\| \leq 8\varepsilon \mathrm{cost}(c_1', P_1' - S)$ and thus $\mathrm{cost}(c_1', P) \leq (1+8\varepsilon)\mathrm{cost}(c_1' \vee c_2', P)$. This implies that we can solve the problem in this case by solving the 1-median problem on P, and thus contradicting our assumption.

Thus, $|P_2'| \geq \varepsilon |P_1' \setminus S|$. We create $P' = P \setminus S = P_1'' \cup P_2'$, where $P_1'' = P_1' \setminus S$. Although P_1'' might now be considerably smaller than P_2'', and as such case 1 does not apply directly, we can overcome this by adding enough copies of c_1' into P_1'', so that it would be of similar size to P_2'.

To carry that out, we again perform an exhaustive enumeration of the possible cardinality of P_2' (up to a factor of 2). This would require checking $O(\log n)$ possibilities. Let $\mathcal{V}$ be the guess for the cardinality of P_2', such that $\mathcal{V} \leq |P_2'| \leq 2\mathcal{V}$.

We add $\mathcal{V}$ copies of c_1' to P_1''. We can now apply the algorithm for the case when the cardinalities of both clusters are comparable, as long as we ensure than the algorithm reports c_1' as one of the medians. To this end, it is not difficult to see that by adding copies of c_1' to P_1'' we also ensured that for any 2 medians x and y, replacing at least one of them by c_1' yields better solution. Therefore, without loss of generality we can assume that the algorithm described above, when applied to P', reports c_1' as one of the medians. The complexity of the algorithm is as stated. ■

Theorem 3.6 *For any point-set $P \subset \mathbb{R}^d$, $\varepsilon < 1$, and a parameter k, a $(1+\varepsilon)$-approximate k-median for P can be found in*

$$2^{(k/\varepsilon)^{O(1)}} d^{O(1)} n \log^{O(k)} n$$

expected time, the results are correct with high-probability.

Proof: We describe the extension of the algorithm of Theorem 3.5 for $k > 2$. The general ideas used in the algorithm are as in the case $k = 2$, but are applied in a recursive fashion.

Let the optimal clusters be $C_1 \ldots C_k$, and let their centers be $c_1 \ldots c_k$. Assume $|C_1| \geq \ldots \geq |C_k|$. We can assume that we know $|C_i|$ up to any constant factor (by exhaustive enumeration of $O(\log n)^k$ possibilities). In the following, for simplicity of exposition we will assume we know $|C_i|$'s *exactly*.

The algorithm proceeds by performing at most k *reductions*. Each reduction reduces the number of remaining clusters to compute (by finding an approximate cluster center for one of them, or by observing that one of the clusters is unnecessary to consider) as well as the total number of points to consider. Moreover, each reduction will induce an increase in the total clustering cost by a factor $(1+O(\varepsilon)/k)$. Thus, the cost of the final clustering will be at most $(1+O(\varepsilon))$ times larger than the optimal cost.

The algorithm proceeds by considering the following cases.

Case 1: $|C_{i+1}| \geq \varepsilon^b/k^b |C_i|$, for a certain constant $b > 1$. In this case we know that $|C_i| \geq n\left(\varepsilon^b/k^b\right)^k$ for all $i = 1 \ldots k$. Thus, we can solve the problem by sampling $\left(\frac{k^b}{\varepsilon^b}\right)^k k^{O(1)}/\varepsilon^{O(1)}$ points, then guessing (by enumeration) which points belong to which clusters, and applying the 1-median sampling theorem for the sample of points of each cluster.

Case 2: $|C_{i+1}| < (\varepsilon^b/k^b)|C_i|$, for some i. Assume that i is the smallest index with this property. Note that $\sum_j > i|C_j| \leq 2\varepsilon^b/k^{b-1}n$.

In this case, we choose a random sample of size at most $n/\sum_{j>i}|C_j|$ and not greater than the one in Case 1. Observe that with constant probability the sample contains only points in $C_1 \cup \ldots \cup C_i$, and moreover it contains enough of them so that $(1+\varepsilon/k)$-approximate medians for the first i clusters can be computed as in Case 1. We will keep these i medians till the end of the computation, and compare the cost of our solution to the cost of the best clustering with the same first i medians. We will denote the optimal solution medians (again) by $c_1 \ldots c_k$, and the induced clusters by $C_1 \ldots C_k$.

Since we know $c_1 \ldots c_i$, it remains to find the remaining cluster centers. Let AvgMed $=$ AvgMed$(C_1 \cup \ldots \cup C_i, i)$. If the distance from any c_j, $j > i$ to the nearest $c_1, \ldots, c_i$ is less than AvgMed, then we ignore (i.e., not compute) such median c_j. Note that the additional cost incurred in this way is at most AvgMed$\cdot 2\varepsilon n/k$, which is at most $O(\varepsilon/k)$ times the optimal cost. Thus, we reduced the number of clusters to at most $k-1$, and we apply the clustering procedure again, this time for the smaller number of clusters.

Assume now that the distance from all c_j, $j > i$ to the nearest $c_1, \ldots, c_i$ is at least AvgMed. In this case we apply the "shrinking" approach, similar to the argument used in Theorem 3.5, to show that we can shrink $C_1, \ldots, C_i$ to make their sizes comparable to C_{i+1}. In particular, let t be the distance between c_{i+1} and its closest points in $c_1 \ldots c_i$. We can estimate t up to any constant factor by guessing one out of $O(\log n)$ approximations. As before, for simplicity, assume t is the exact distance.

We know that all points within distance $< t/2$ to $c_1, \ldots, c_i$ belong to clusters $C_1, \ldots, C_i$. Therefore, we can remove them from the set to cluster. Note that this does not change the medians $c_1 \ldots c_k$. However, the number of points with distance $> t/2$ from $c_1, \ldots, c_k$ is comparable to the size of C_{i+1}, or otherwise c_{i+1} is redundant. Specifically, assume that the number of such points is m. Then the cost of the clustering is at least $mt/2$. If we removed c_{i+1}, the cost would increase by at most $t|C_{i+1}|$. Thus, if $m > 2k/\varepsilon|C_{i+1}|$, then we could reduce the number of clusters by 1 and proceed recursively. If this is not the case, it means that $m < 2k/\varepsilon|C_{i+1}|$. At the same time we can enforce $|C_j| \geq |C_{i+1}|$ for $j \leq i$ by adding copies of c_j (as for the case $k = 2$). We can now apply the algorithm recursively to find the remaining medians. Observe that this particular recursive step can be applied at most k times, since after the step the minimum value of i s.t. $|C_{i+1}| \ll |C_i|$ is increased by 1.

This is yields a recursive algorithm, that gets rid one one cluster in each recursive call. Thus, the algorithm has the running time stated. Note, that we need to rerun this algorithm $O\left(2^{O(k)} \log n\right)$

times to get high probability of correctness. ∎

4 k-center clustering with outliers

In this section, we extend our algorithm from Section 2 to handle outliers.

Definition 4.1 For a point-set P in $\mathbb{R}^d$, let $r_{cen}(P,k,\alpha)$ denote the minimal radius clustering with outliers; namely, we allow to throw out $\alpha|P|$ outliers. Computing this value is the (k,α)-center problem. Formally,

$$r_{cen}(P,k,\alpha) = \min_{S\subseteq P, |S|\geq(1-\alpha)|P|} r_{cen}(S,k).$$

The problem of computing k-center with outliers is interesting, as the standard k-center clustering is very sensitive to outliers.

Theorem 4.2 *For any point-set $P \subset \mathbb{R}^d$, parameters $1 > \varepsilon, \alpha > 0, \mu > 0$, a random sample R of $O(1/(\varepsilon\mu))$ points from P spans a flat containing a $(1+\varepsilon, 1, \mu+\alpha)$-approximate solution for the 1-center with α-outliers for P. Namely, there is a point $p \in \mathrm{span}(R)$, such that the ball of radius $(1+\varepsilon)r_{cen}(P,1,\alpha)$ centered at p, contains at least $(1-\alpha-\mu)$ points of P.*

Furthermore, we can compute such a cluster in $O(f(\varepsilon,\mu)nd)$ time, where $f(\varepsilon,\mu) = \exp\left(O\left(\frac{1}{\varepsilon\mu}\log^2\frac{1}{\varepsilon\mu}\right)\right)$

Proof: Let c_{opt} denote the center of the optimal solution, $r_{opt} = r_{cen}(P,1,\alpha)$ denote its radius, and $B_{\mathrm{opt}} = \mathrm{Ball}(c_{opt}, r_{opt})$. Let $s_1,\ldots,s_i$ be our random sample, $F_i = \mathrm{span}(s_1,\ldots,s_i)$, and $c_i = \mathrm{proj}(c_{opt}, F_i)$, and we set $\beta = \sqrt{\varepsilon}$, and

$$U_i = \left\{ x \,\middle|\, \pi/2 - \beta \leq \angle c_{opt} c_i x \leq \pi/2 + \beta \right\},$$

be the complement of the cone of angle $\pi - \beta$ emanating from c_i, and having $c_i c_{opt}$ as its axis.

Let $P_i = U_i \cap P \cap B_{\mathrm{opt}}$. For any point $p \in P_i$, we have

$$\|p - c_i\| \leq \sqrt{x_p^2 + y_p^2} \leq r_{opt}\sqrt{1+4\beta^2} = r_{opt}(1+O(\varepsilon)).$$

Namely, as far as the points of P_i are concerned, c_i is a good enough solution.

Let $Q_i = P \setminus P_i$. As long as $|Q_i| \geq (\alpha+\mu)|P|$, we have a probability of $1/\mu$ to sample a random point that is in Q_i and is not an outlier. Arguing as in the proof of Theorem 3.2, in such a case, the distance of the current flat to c_{opt} shrank down by a factor of $(1-\beta^2/4)$. Thus, as in the proof of Theorem 3.2, we perform the random sampling in rounds. In our case, we need $O((1/\varepsilon)\log(1/\varepsilon))$ rounds, and each round requires in expectation $1/\mu$ random samples. Thus, overall, if we sample $M = O((1/(\varepsilon\mu))\log(1/\varepsilon))$ points, we know that with constant probability, $\mathrm{span}(s_1,\ldots,s_M)$ contains a point in distance εr_{opt} from c_{opt}.

As for the algorithm, we observe that using random sampling, we can approximate $r_{opt}(P,1,\alpha)$ up to a factor of two in $o(dn)$ time. Once we have this approximation, we can generate a set of candidates, as done in Theorem 3.4. This would result in

$$f(\varepsilon,\mu) = O\left(\left(\frac{10|R|}{\varepsilon}\right)^{|R|} \cdot |R|\right) = 2^{O((1/(\varepsilon\mu))\log^2(1/(\varepsilon\mu)))}$$

candidates. For each candidate we have to spend $O(nd)$ time on checking its quality. Overall, we get $f(\varepsilon,\mu)nd$ running time. ■

Remark 4.3 Theorem 4.2 demonstrates the robustness of the sampling approach we use. Although we might sample outliers into the sample set R, this does not matter, as in our argumentation we concentrate only on the points that are sampled correctly. Essentially, $\mathrm{dist}(F_i, c_{opt})$ is a monotone decreasing function, and the impact of outliers, is only on the size of the sample we need to take.

Theorem 4.4 *For any point-set $P \subset \mathbb{R}^d$, parameters $1 > \varepsilon, \alpha > 0, \mu > 0, k > 0$, one can compute a $(k, \alpha+\mu)$-center clustering with radius smaller than $(1+\varepsilon)r_{cen}(P,k,\alpha)$ in $2^{(k/\varepsilon\mu)^{O(1)}}nd$ time. The result is correct with constant probability.*

Proof: Let P' be the set of points of P covered by the optimal k-center clustering $C_1,\ldots,C_k$, with αn outliers. Clearly, if any of the C_is contain less than $(\mu/k)n$ points of P', we can just skip it altogether.

To apply Theorem 4.2, we need to sample $O(k/(\varepsilon\mu))$ points from each of those clusters (we apply it with μ/k for the allowed fraction of outliers). Each such cluster, has size at least $(\mu/k)n$, which implies that a sample of size $O(k^2/(\mu^2\varepsilon))$ would contain enough points from C_i. To improve the probabilities, we would sample $O(k^3/(\mu^2\varepsilon))$ points. Let R be this random sample. With constant probability (by the Markov inequality), $|R \cap C_i| = \Omega(k/(\varepsilon\mu))$, for $i = 1,\ldots,k$. We exhaustively enumerate for each point of R to which cluster it belongs. For such partition we apply the algorithm of Theorem 4.2. The overall running time is $2^{O((k/(\varepsilon\mu)^{O(1)}))}nd$. ■

5 The 1-cylinder problem

Let P be a set of n points in $\mathbb{R}^d$, we are interested in finding a line which minimizes the maximum distance from the line to the points. More specifically, we are interesting in finding a $(1+\varepsilon)$ approximation to the problem in polynomial time.

In the following, let ℓ_{opt} denote the axis-line of the optimal cylinder, $r_{opt} = \mathrm{radius}(P, \ell_{opt}) = \max_{p\in P} \left\| \ell_{opt} - p \right\|$ denote its radius, and H_{opt} denote the hyperplane perpendicular to ℓ_{opt} that passes through the origin.

In this section, we prove the following theorem:

Theorem 5.1 *Given a set of n points in $\mathbb{R}^d$, and a parameter $\varepsilon > 0$, one can compute, in $n^{O(\log(1/\varepsilon)/\varepsilon^2)}$ a line l, such that* $\mathrm{radius}(P,l) \le (1+\varepsilon)r_{opt}$*, where r_{opt} is the radius of the minimal 1-cylinder that contains P.*

5.1 The Algorithm

First, observe that one can compute a 2-approximation to the 1-cylinder problem by finding the furthest away 2 points $p, q \in P$ (i.e., the diameter) and taking the minimum cylinder having pq as its axis that contains P. It is easy to verify that the radius R of this cylinder is at most $2r_{opt}$, where r_{opt} is the radius of the smallest cylinder. Computing R takes $O(n^2 d)$ time.

Let l be the center line of an optimal solution with cost r_{opt}. We will assume that the value of r_{opt} is known to us (up to a factor very close to 1), as we can enumerate all potential values of r_{opt} in the range $R/2 \dots R$. The high-level idea of the algorithm is to compute a $(1+\varepsilon)$ distortion embedding of the n points from $\mathbb{R}^d$ into $\mathbb{R}^{\log n/\varepsilon^2}$, "guess" the solution to the problem there and then to retrieve the solution in the original space. However, a simple application of this method is known to not work for *continuous* clustering problems (and 1-cylinder in particular), due to the following problems:

- The optimal solution found in the low-dimensional space does not have to correspond to any solution in the original space.
- Even if this was true, it is not clear how to retrieve the high-dimensional solution from the low-dimensional one.

To overcome this difficulties, we proceed as follows:

1. In the first step, we find a point h lying on the ℓ_{opt}. To be more precise, we "guess" it (by enumerating polynomially many candidates) instead of finding it; moreover, the point does not lie on the line by only sufficiently close to it.
2. We remove from P all the points within distance $(1+\varepsilon)r_{opt}$ from h. Since our final solution line passes through h, it is sufficient to find a solution for the smaller set P.
3. We embed the whole space into a low-dimensional space, such that with high probability for all points $p \in P$, the angles $\angle \overrightarrow{ph}\ell_{opt}$ are approximately preserved.

   As we discuss later, such an embedding A is guaranteed by Johnson-Lindenstrauss lemma.
4. We guess (approximately) the low-dimensional image Al of the line l (again, by exploring polynomially many possibilities). By the properties of the embedding A, we can detect which points in P lie on the positive side of the $(d-1)$-dimensional hyperplane H passing through h and orthogonal to l. We modify the set P by replacing each point p from the negative side of the hyperplane by its reflection around h. Note that this operation does not increase the solution cost by too much.
5. It is now sufficient to find an optimal half-infinite *ray* beginning at point h, to minimize the distance from P to the ray. Moreover, we know that the points are on the same side of a $(d-1)$-dimensional hyperplane that passes through h. This problem can be solved using convex programming tools.

Thus, we use the low-dimensional image of l to discover the "structure" of the optimal solution, namely which points lie on which side of the hyperplane H. Knowing this fact allows us to reduce our (non-convex) problem to a convex one.

5.2 Detailed description of computing the approximate 1-cylinder

In the following, we elaborate on each step in computing the min-radius cylinder containing a point-set.

5.2.1 Step 1: finding a point near the line

In order to find the point h, we will need to embed additional "helper" points. For each subset S of P, $M = |S| = O(1/\varepsilon^2)$, we compute an ε-net on the interior of the convex body spanned by them.

Definition 5.2 Let U be any set of points in $\mathbb{R}^d$, and $\varepsilon > 0$ be a parameter. We say that a subset V of points is an ε-*net* for U if for any line ℓ that intersects $\mathcal{CH}(U)$, there is a $v \in V$ such that $\mathrm{dist}(v, \ell) \leq \frac{\varepsilon}{2} r_{opt}$.

Lemma 5.3 *Let U be a set of points in $\mathbb{R}^d$ and $\varepsilon > 0$ be a parameter. We can compute an ε-net $A(U)$ for U in $(|U|^{2.5}/\varepsilon)^{O(|U|)}$ time. The cardinality of $A(U)$ is $(|U|^{2.5}/\varepsilon)^{O(|U|)}$.*

Proof: Let H the $(M-1)$-dimensional affine subspace spanned by U (notice that $M \leq |U|$), and let $\mathcal{E} \subseteq H$ be an ellipsoid such that $\mathcal{E}/(M+1)^2 \subseteq \mathcal{CH}(U) \subseteq \mathcal{E}$, where $\mathcal{E}/(M+1)^2$ is the scaling down of $\mathcal{E}$ around its center by a factor of $1/(M+1)^2$. Such an ellipsoid exists (a stronger version of this statement is known as John theorem), and can be computed in polynomial time in $|U|$ [GLS88, Section 4.6]. Let $\mathcal{B}$ be the minimum bounding box of $\mathcal{E}$ which is parallel to the main axises of $\mathcal{E}$. We claim, that $\mathcal{B}/\sqrt{M}$ is contained inside $\mathcal{E}$. Indeed, there exists a linear transformation $\mathcal{T}$ that maps $\mathcal{E}$ to a unit ball $\mathcal{S}$. The point $\mathbf{q} = (1/\sqrt{M}, 1/\sqrt{M}, \ldots, 1/\sqrt{M})$ lies on the boundary of this sphere. Clear, $\mathcal{T}^{-1}(\mathbf{q})$ is a corner of $\mathcal{B}/\sqrt{M}$, and is on the boundary of $\mathcal{E}$. In particular,

$$\begin{aligned}\Delta(\mathcal{B}) &= \sqrt{M}\Delta(\mathcal{B}/\sqrt{M}) \leq \sqrt{M}\Delta(\mathcal{E}) \\ &\leq \sqrt{M}(M+1)^2\Delta(\mathcal{E}/(M+1)^2) \leq \sqrt{M}(M+1)^2\Delta(U).\end{aligned}$$

For any line ℓ, the same arguments works for the projection of those entities in the hyperplane perpendicular to ℓ. Let $\mathcal{TP}_\ell$ denote this projection, we have:

$$\begin{aligned}\Delta(\mathcal{TP}_\ell(\mathcal{B})) &\leq \sqrt{M}(M+1)^2\Delta(\mathcal{TP}_\ell(U)) \\ &\leq 2\sqrt{M}(M+1)^2\,\mathrm{dist}(U,\ell).\end{aligned}$$

Next, we partition $\mathcal{B}$ into a grid, where each grid cell is a translated copy of $\mathcal{B}_\varepsilon = (\varepsilon/2)\mathcal{B}/(2\sqrt{M}(M+1)^2)$. This grid has $V = (M^{2.5}/\varepsilon)^{O(M)}$ vertices, and let $A(U)$ denote this set of vertices.

Let ℓ be any flat intersecting $\mathcal{CH}(U)$. We claim that one of the points in $A(U)$ is in distance $\leq \frac{\varepsilon}{2}\,\mathrm{dist}(U,\ell)$ from ℓ. Indeed, let z be any point in $\mathcal{CH}(U) \cap \ell$. Let $\mathcal{B}''_\varepsilon$ be the grid cell containing z, and let v be one of its vertices. Clearly,

$$\begin{aligned}\mathrm{dist}(v,\ell) &\leq \|\mathcal{TP}_\ell(v)\mathcal{TP}_\ell(z)\| \leq \Delta(\mathcal{TP}_\ell(\mathcal{B}''_\varepsilon)) \\ &= \frac{\varepsilon}{2}\cdot\frac{1}{2\sqrt{M}(M+1)^2}\Delta(\mathcal{TP}_\ell(\mathcal{B})) \leq \frac{\varepsilon}{2}\,\mathrm{dist}(U,\ell),\end{aligned}$$

which establishes our claim. ■

Let $G(S) = A(S)$ as defined by Lemma 5.3. Clearly, $|G(S)| = (|S|^{2.5}/\varepsilon)^{O(|S|)}$, where $|S| = O(1/\varepsilon^2)$. We have

$$|G(S)| = 2^{O(\log(1/\varepsilon)/\varepsilon^2)}.$$

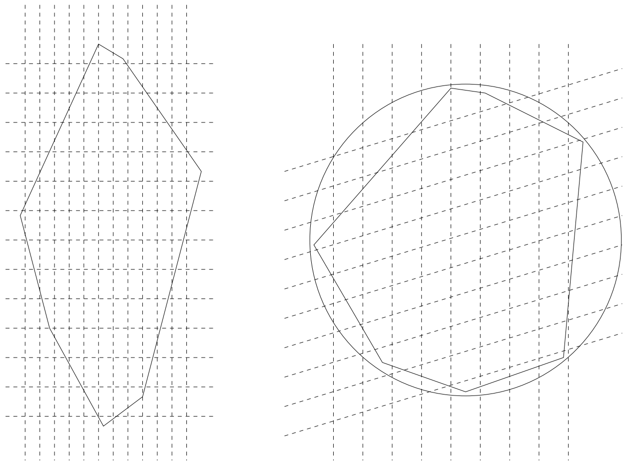

Figure 3: Any point inside the convex body has at least 1 “helper” point at distance at most $\varepsilon/2r_{opt}$ after the embedding.

Lemma 5.4 *Consider the points in $G(S)$ for all sets S. At least one of those points is at most εr_{opt} away from ℓ_{opt}.*

Proof: Project all the points into a hyperplane H_{opt}, and denote this set of points by P'. Let o be the point corresponding to the line intersected with H_{opt}. Since all the points are at distance at most r_{opt} from l, all the points projected into H will be at distance at most r_{opt} from o. Compute the minimum enclosing ball of the point-set P'. It is easy to see that if the origin of the minimum enclosing ball is not o, then we can come up with a solution for the minimum fitting line of cost lower than r_{opt} by just translating l to intersect the center of the ball. Therefore, it must be that the minimum enclosing ball of P' has the origin in o. By Theorem 2.5, there exists a set $S \subset P'$, $|S| = O(1/\varepsilon^2)$, such that the minimum enclosing ball of the S is at most $(\varepsilon/2)r_{opt}$ away from o and since the center of any minimum enclosing ball of a set of points can be written as a convex combination of the points, we conclude that there exists a point p, a convex combination of the points of S such that $D(p,o) \leq \varepsilon r_{opt}$. Also, distance from p to the closest point of $G(S)$ is at most $(\varepsilon/2)r_{opt}$.

Therefore, there exists a point in our ε-net that is at most εr_{opt} away from the optimal fitting line. ■

5.2.2 Step 2: Removing the points near h

For the simplicity of the exposition, we assume, from this point on, that h *lies* on the optimal line ℓ_{opt}. We remove from P all the points within distance $(1+\varepsilon)r_{opt}$ from h.

The removal step can be clearly implemented in linear time. Observe that after this step, for all points p, the angle between $\overrightarrow{ph}$ and ℓ_{opt} is in the range $[0, \pi/2 - \varepsilon/2] \cup [\pi/2 + \varepsilon/2, \pi]$ for small enough ε. As we will see in the next section, this property implies that the angles do not change the value from less than $\pi/2$ to greater than $\pi/2$ after applying the dimensionality reduction.

5.2.3 Step 3: Random projection

In this section we show how to find a mapping $A : \mathbb{R}^d \to \mathbb{R}^{d'}$ for $d' = O(\log n/\varepsilon^2)$ which preserves all angles $\angle\overrightarrow{hp}\ell_{opt}$, $p \in P$, up to an additive factor of $\varepsilon/3$. For this purpose we use the Johnson-Lindenstrauss lemma. It is not difficult to verify (e.g., see [EIO02]) that if we set the error parameter of the JL lemma to ε/C for large enough constant C, then all the aforementioned angles are preserved up to an additive factor of, say, $\varepsilon/4$. This implies that for each $p \in P$, the image of the point p is on the same side of the (image of) the hyperplane H as the original points p. Moreover, for any $p \in P$ the angle $\angle(\overrightarrow{hp}, \ell_{opt})$ are in the range $[0, \pi/2 - \varepsilon/4] \cup [\pi/2 + \varepsilon/4, \pi]$, i.e., are still strictly separated from $\pi/2$.

5.2.4 Step 4: Guessing the image of l

We now need to guess (approximately) the image $A\ell_{opt}$, where A is the mapping generated by the JL lemma. For this purpose, we need to know the direction of l, since we already know one point through which the line $A\ell_{opt}$ passes through. Our approach is to enumerate all "different" directions of a line $A\ell_{opt}$. Obviously, the number of such directions is infinite. However, since we use the line *exclusively* for the purpose of separating the points $p \in P$ depending on their angle $\angle\overrightarrow{hp}\ell_{opt}$, and those angles are separated from $\pi/2$ by $\varepsilon/4$, it is sufficient to find a direction vector which is within angular distance $\varepsilon/4$ from the direction of l. Thus, it is sufficient to enumerate all directions from an $\varepsilon/4$-net for the space of all directions. It is known that such spaces of cardinality $(1/\varepsilon)^{O(d')} = n^{O(\log(1/\varepsilon)/\varepsilon^2)}$ exist and are constructible. Thus, we can find the right partition of points in P by enumerating a polynomial number of directions.

After finding the right partition of P (say, into P_L and P_R), we replace each point in P_L by its reflection through h; call the resulting set $P'_L = \left\{2h - p \,\middle|\, p \in P\right\}$. Note that there is a one-to-one correspondence between the 1-cylinder solutions for P which pass through h and the 1-*ray* solutions for $P' = P'_L \cup P_R$; the 1-ray problem is defined as to find a ray r with an endpoint at h which minimizes the maximum, over all input points p, of the distance from p to r. Thus, it remains to solve the 1-ray problem for P'.

5.2.5 Step 5: Solving the 1-ray problem using convex programming

We focus on the decision version of this problem. Assume we want to check if there is a solution with cost at most T. For each point p, define a cone C_p to be the set of all rays with endpoints in h which are within distance T from p. Clearly, C_p is convex. The problem now corresponds to checking if an intersection of all cones C_p is nonempty, which is an instance of convex programming and thus can be solved up to arbitrary precision in polynomial time [GLS88].

6 Conclusions

In this paper, we presented several very fast algorithms for doing $(1+\varepsilon)$-approximate clustering. Our algorithm relied on the ability to compute small core-sets for those clustering problems. We believe that the techniques presented in this paper might be useful for other clustering problems.

In particular, it would be interesting to extend the set of problems for which we know the existence and construction of a small core-set.

References

[ADPR00] N. Alon, S. Dar, M. Parnas, and D. Ron. Testing of clustering. In *Proc. 41th Annu. IEEE Sympos. Found. Comput. Sci.*, 2000.

[AP98] P.K. Agarwal and C.M. Procopiuc. Exact and approximation algorithms for clustering. In *Proc. 9th ACM-SIAM Sympos. Discrete Algorithms*, pages 658–667, 1998.

[ARR98] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for Euclidean k-median and related problems. In *Proc. 30th Annu. ACM Sympos. Theory Comput.*, pages 106–113, 1998.

[BE97] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. In D.S. Hochbaum, editor, *Approximationg algorithms for NP-Hard problems*. PWS Publishing Company, 1997.

[DHS01] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley-Interscience, New York, 2nd edition, 2001.

[EIO02] L. Engebretsen, P. Indyk, and R. O'Donnell. Derandomized dimensionality reduction with applications. In *Proc. 13th ACM-SIAM Sympos. Discrete Algorithms*, 2002. to appear.

[GIV01] A. Goel, P. Indyk, and K.R. Varadarajan. Reductions among high dimensional proximity problems. In *Proc. 12th ACM-SIAM Sympos. Discrete Algorithms*, pages 769–778, 2001.

[GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin Heidelberg, 2nd edition, 1988. 2nd edition 1994.

[Gon85] T. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoret. Comput. Sci.*, 38:293–306, 1985.

[Har01] S. Har-Peled. Clustering motion. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 84–93, 2001.

[HV01] S. Har-Peled and K.R. Varadarajan. Approximate shape fitting via linearization. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 66–73, 2001.

[Ind01] P. Indyk. Algorithmic applications of low-distortion geometric embeddings. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 10–31, 2001. Tutorial.

[IT00] P. Indyk and M. Thorup. Approximate 1-median. manuscript, 2000.

[MOP01] N. Mishra, D. Oblinger, and L. Pitt. Sublinear time approximate clustering. In *SODA 12*, 2001.

[OR00] R. Ostrovsky and Y. Rabani. Polynomial time approximation schemes for geometric k-clustering. In *Proc. 41st Symp. Foundations of Computer Science*, pages 349–358. IEEE, Nov 2000.