



ELSEVIER

Available online at [www.sciencedirect.com](http://www.sciencedirect.com)

Computational Geometry ●●● (●●●●) ●●●-●●●

Computational  
Geometry

Theory and Applications

[www.elsevier.com/locate/comgeo](http://www.elsevier.com/locate/comgeo)

# A linear time algorithm for approximate 2-means clustering

Yogish Sabharwal<sup>a,\*</sup>, Sandeep Sen<sup>b,1</sup><sup>a</sup> IBM India Research Lab, Block-I, IIT Delhi, Hauz Khas, New Delhi-110016, India<sup>b</sup> Department of Computer Science & Engg, Indian Institute of Technology, New Delhi-110016, India

Received 26 March 2004; received in revised form 13 January 2005; accepted 14 January 2005

Communicated by P. Agarwal

## Abstract

Matousek [Discrete Comput. Geom. 24 (1) (2000) 61–84] designed an  $O(n \log n)$  deterministic algorithm for the approximate 2-means clustering problem for points in fixed dimensional Euclidean space which had left open the possibility of a linear time algorithm. In this paper, we present a simple randomized algorithm to determine an approximate 2-means clustering of a given set of points in fixed dimensional Euclidean space, with constant probability, in linear time.

We first approximate the mean of the larger cluster using random sampling. We then show that the problem can be reduced to a set of lines, on which it can be solved by carefully pruning away points of the larger cluster and randomly sampling on the remaining points to obtain an approximate to the mean of the smaller cluster.

© 2005 Published by Elsevier B.V.

## 1. Introduction

Clustering is the grouping of similar objects and a clustering of a set is a partition of its elements that minimizes some measure of dissimilarity. Clustering has applications in a variety of areas, for example, data mining, information retrieval, image processing, and web search [1,2,4,11].

\* Corresponding author.

*E-mail addresses:* [ysabharwal@in.ibm.com](mailto:ysabharwal@in.ibm.com) (Y. Sabharwal), [ssen@cse.iitd.ernet.in](mailto:ssen@cse.iitd.ernet.in) (S. Sen).

<sup>1</sup> Author's present address: Department of Computer Science & Engg, Indian Institute of Technology Kharagpur 721302, India.

1 A popular clustering method for points in Euclidean space is  $k$ -means clustering. For completeness, 1  
2 we begin with some formal definitions. 2

3  
4 **Definition 1.1** ( *$k$ -means clustering*). Given a set  $P$  of  $n$  data points in  $d$ -dimensional Euclidean space,  $\mathfrak{R}^d$ , 4  
5 and an integer  $k$ , the problem is to determine a set,  $K$ , of  $k$  points in  $\mathfrak{R}^d$ , called centers, that minimize the 5  
6 mean squared Euclidean distance from each data point to its nearest center. This measure is often called 6  
7 the squared error distortion ( $\Delta$ ). 7

$$8 \quad \Delta(P) = \sum_{p \in P} d(p, K)^2$$

9 where  $d(p, K)$  = Euclidean distance of  $p$  from its closest point in  $K$ . 9  
10 10

11  
12 The clusters  $S_1, S_2, \dots, S_k$  of  $P$  are derived from the set  $K = \{x_1, x_2, \dots, x_k\}$ . The cluster  $S_i$  is derived 12  
13 from the point  $x_i$  by taking those points of  $P$  for which  $x_i$  is the closest point in  $K$ . Thus the points are 13  
14 partitioned by the Voronoi diagram of the set of centers. 14

15 Moreover,  $x_i$  is the centroid of set  $S_i$ , i.e., 15

$$16 \quad x_i = C(S_i) = \frac{1}{|S_i|} \sum_{s \in S_i} s.$$

17  
18 Clustering based on  $k$ -means is closely related to a number of other clustering problems. These include 18  
19 the  $k$ -medians, in which the objective is to minimize the sum of the distances to the nearest center, and 19  
20 the  $k$ -center problem, in which the objective is to minimize the maximum distance. 20  
21

22 Polynomial time approximation algorithms for clustering problems are very important considering 22  
23 that the exact solutions are often intractable [5,10]. The approximation version of the  $k$ -means problem 23  
24 is to determine a solution that is within a constant factor of the optimal solution. 24

25  
26 **Definition 1.2** ( *$(1 + \varepsilon)$ -approximate  $k$ -means clustering*). Given a set  $P$  of  $n$  data points in  $d$ -dimensional 26  
27 Euclidean space,  $\mathfrak{R}^d$ , an integer  $k$ , and an approximation factor  $\varepsilon > 0$ , the problem is to determine a set of 27  
28  $k$  points in  $\mathfrak{R}^d$ , so that the squared error distortion,  $\Delta$  is within a factor of  $(1 + \varepsilon)$  of the optimal squared 28  
29 error distortion  $\Delta_0$ , i.e., 29

$$30 \quad \Delta \leq (1 + \varepsilon) \Delta_0.$$

31  
32 The 2-means clustering problem is the special case of the  $k$ -means clustering problem where  $k = 2$ . 32  
33 Even the 2-means clustering problem is NP-hard when  $d$  is not fixed [3]. 33

34 We use  $\Delta$  to denote the  $k$ -means cost of a point set with respect to a given point, i.e., 34

$$35 \quad \Delta(S, y) = \text{Cost}_{k\text{-means}}(S, y) = \sum_{x \in S} \|x - y\|^2.$$

36  
37 When the given point  $y$  is the centroid ( $y = C(S) = \frac{1}{|S|} \sum_{x \in S} x$ ) of the given point set  $S$ , then we simply 37  
38 denote it as  $\Delta(S)$ , i.e., 38

$$39 \quad \Delta(S) = \sum_{x \in S} \|x - C(S)\|^2.$$

40  
41 When the set  $S$  is a singleton  $\{x\}$ , then we simply denote it as  $\Delta(x, y)$ , i.e., 41

$$42 \quad \Delta(x, y) = \|x - y\|^2.$$

43  
44 This is simply the square of the Euclidean metric. 44

### 1.1. Our results

We present a randomized algorithm that determines the approximate  $(1 + \varepsilon)$  2-means clustering of a given point set in linear time, with constant probability, for any dimension.

It is interesting that even the 2-means clustering problem has eluded efficient and simple algorithms for a long time. Inaba et al. [7] gave an algorithm to compute the  $(1 + \frac{1}{\Omega(n \log n)})$  approximate clustering on the plane in  $O(n^{5/3} \log^2 n)$  time with probability  $(1 - \frac{1}{\Omega(n \log n)})$ . For higher dimensions, the running time of their algorithm increases exponentially in  $d$ . Recently, Matousek [9] gave an  $O(n \log n)$  deterministic algorithm for 2-means clustering. The algorithm is fairly complicated and relies on several results in computational geometry that depend heavily on the dimensions, leading to a running time of  $O(n \log n \cdot \varepsilon^{-2d} \log(1/\varepsilon) + n \varepsilon^{-(4d-2)} \log(1/\varepsilon))$ .

One of the questions left open in [9] by Matousek was whether  $\Omega(n \log n)$  is a lower bound. We present a randomized algorithm with running time of  $O((1/\varepsilon)^{O(1/\varepsilon)} (d/\varepsilon)^d n)$ , which is linear in  $n$ .<sup>2</sup>

## 2. Preliminaries

We describe here some definitions and known results that we will use later in our algorithm.

### 2.1. Approximating the centroid of a set of points

The following lemma due to Inaba et al. [7] shows that the centroid of a set,  $P$ , of points in  $d$ -dimensional Euclidean space can be approximated by random sampling.

**Lemma 2.1** [7]. *Let  $T$  be a sample of  $m$  points obtained by  $m$  independent draws at random from  $S$ . Then with probability  $1 - \delta$ ,*

$$\Delta(S, C(T)) < \left(1 + \frac{1}{\delta m}\right) \Delta(S).$$

Thus, by considering the centroid of a random sample,  $S$ , of size  $2/\varepsilon$  ( $m = 2/\varepsilon$ ), with constant probability ( $\delta = 1/2$ ), we can approximate the centroid of a set of points  $P$ , so that the cost of the clustering to this point is within a factor of  $(1 + \varepsilon)$  of the cost of the clustering to the mean of  $P$ . We call such an approximate, an  $\varepsilon$ -approximation to the centroid.

Note that this also establishes the existence of a set of  $2/\varepsilon$  points of  $P$ , whose centroid is an  $\varepsilon$ -approximation to the centroid of  $P$ .

### 2.2. $\varepsilon$ -near pairs

We present here the concept of  $\varepsilon$ -near pairs and a result on such pairs due to Matousek [9].

<sup>2</sup> In an independent work, Har-Peled and Mazumdar [6] discovered a linear time algorithm that runs in linear time for fixed  $k$  and in a more recent work [8] the authors describe a linear time algorithm for fixed  $k$  and any dimension.

**Definition 2.1.** For a real number  $\varepsilon \geq 0$ , we define a relation  $\sim_\varepsilon$  on (ordered) pairs of points in  $d$ -dimensional Euclidean space,  $\mathfrak{R}^d$ : we let  $(x, y) \sim_\varepsilon (x', y')$  if  $\|x - x'\| \leq \varepsilon\|x - y\|$  and  $\|y - y'\| \leq \varepsilon\|x - y\|$ . We say that  $(x, y)$  and  $(x', y')$  are  $\varepsilon$ -near if  $(x, y) \sim_\varepsilon (x', y')$  and  $(x', y') \sim_\varepsilon (x, y)$ . The relation  $\sim_\varepsilon$  is not “quite” symmetric.

The following lemma is due to Matousek [9].

**Lemma 2.2** [9]. *If  $(x, y)$  and  $(x', y')$  are  $\varepsilon$ -near pairs, and if  $C$  is the cost of the 2-means clustering induced by  $(x, y)$  and  $C'$  is the cost of the 2-means clustering induced by  $(x', y')$ , then,*

$$C \leq (1 + 16\varepsilon)C'.$$

This lemma states that if there are two pairs of centers that are close to each other, then the cost of the clusterings induced by the two pairs does not differ very much.

### 3. The algorithm

The main ideas behind the algorithm are as follows. We first obtain candidate centers for the centroid of the larger cluster by random sampling. We then construct a set of lines through each of these candidate centers to obtain a line close enough to the line joining the optimal centroid pair. We reduce the *approximate 2-means clustering* problem to a set of *center location on a line* problems determined by these lines. We then solve the approximate version of the *center location on a line* problem (defined below formally) on these lines by random sampling. This gives us an approximation to the centroid of the smaller cluster.

We define the center location on a line problem as follows:

**Definition 3.1** (*Center location on a line*). Given a set  $P$  of  $n$  data points in  $d$ -dimensional Euclidean space,  $\mathfrak{R}^d$ , a fixed point  $c_1$  and a line  $\ell$  passing through  $c_1$ , the problem is to find another point  $c_2$  on the line  $\ell$  so as to minimize the cost of the 2-means clustering induced by  $(c_1, c_2)$ . This measure,  $\phi$ , is the cost of the center location on the line problem and is defined as

$$\phi(P, c_1, \ell) = \min_{c_2 \in \ell} \{ \Delta(P_1, c_1) + \Delta(P_2, c_2) \}$$

where  $(P_1, P_2)$  is the partitioning induced by  $(c_1, c_2)$  (i.e., the partitioning induced by the perpendicular bisector of  $(c_1, c_2)$ ).

The approximation version of this problem is stated as follows:

**Definition 3.2** ( $(1 + \varepsilon)$ -approximate center location on a line). Given a set  $P$  of  $n$  data points in  $d$ -dimensional Euclidean space,  $\mathfrak{R}^d$ , a fixed point  $c_1$ , a line  $\ell$  passing through  $c_1$  and an approximation factor  $\varepsilon > 0$ , the problem is to determine another point  $c_2$  on the line  $\ell$  such that

$$\Delta(P_1, c_1) + \Delta(P_2, c_2) \leq (1 + \varepsilon)\phi(P, c_1, \ell)$$

where  $(P_1, P_2)$  is the partitioning induced by  $(c_1, c_2)$ .

We also define a  $(p, \theta)$ -covering as follows:

**Definition 3.3** ( $(p, \theta)$ -covering). Given a point  $p$  in  $d$ -dimensional Euclidean space,  $\mathbb{R}^d$ , and a parameter  $\theta$  ( $0 < \theta < 2\pi$ ), we define a  $(p, \theta)$ -covering to be a set of lines  $\mathcal{Y}$  passing through  $p$ , such that, for any line  $\ell$ , passing through  $p$ , there exists a line  $\ell'$  in  $\mathcal{Y}$  that makes an angle at most  $\theta$  with  $\ell$ .

The formal algorithm is described by the procedures 2-means and CenterLocation described in Figs. 1 and 2 respectively.

Algorithm 2-means reduces the *approximate 2-means clustering* problem to a set of *center location on a line* problems. For this, we start by randomly sampling a set of points, from which we can derive a set of candidate centers that contains an approximation to the centroid of the larger cluster, with constant probability. We proceed with the remaining steps for each of the candidate centers in the set. Now that we have an approximation to one of the centroids, our goal is to approximate the other centroid. We first approximate the line joining the optimal centers. For this, we construct a set of lines (covering) passing

---

**Algorithm 2-means**( $P, \varepsilon$ )

**Inputs :**  $P$ : Point set,  $\varepsilon$ : approximation factor

**Output :** A  $(1 + \varepsilon)$  *approximate 2-means clustering* of the points in  $P$

1. Let  $\alpha = \varepsilon/16$ ,  $\theta = \varepsilon/1024$ ,  $\delta = \varepsilon/4$ .
  2. (Approximate the centroid of the larger cluster)
    - (a) Randomly sample a set,  $R$ , of  $8/\alpha$  points (independently drawn) from  $P$ .
    - (b) Let  $T$  be the set of centroids of all subsets of  $R$  of size  $2/\alpha$ .
  3. For each point  $c'_1$  in  $T$ ,
    - (a) Construct a set  $\mathcal{Y}$  of lines that is a  $(c'_1, \theta)$ -covering.
    - (b) For each  $\ell$  in  $\mathcal{Y}$ , solve **CenterLocation**( $P, c'_1, \ell, \delta$ ).
  4. Return the solution which has minimum cost.
- 

Fig. 1. The 2-means algorithm.

---

**Algorithm CenterLocation**( $P, c_1, \ell, \delta$ )

**Inputs :**  $P$ : Point set,  $c_1$ : fixed center,

$\ell$ : line passing through  $c_1$  on which to locate the second center

$\delta$ : approximation factor

**Output :** The solution for a  $(1 + \delta)$ -*approximate center location on line*  $\ell$

1. Project the points of  $P$  onto line  $\ell$ . Let  $P'$  be the projected points.
  2. (Assume center to be located is to the right of  $c_1$ )
 

For  $i = 1$  to  $\log |P'|$

    - (a) Let  $M_R$  be the right most  $|P'|/2^i$  points of  $P'$ .
    - (b) Randomly sample a set,  $R$ , of  $8/\delta$  points (independently drawn) from  $M_R$ .
    - (c) Let  $T$  be the set of centroids of all subsets of  $R$  of size  $2/\delta$ .
    - (d) For each  $c'_2$  in  $T$ , compute the cost of the clustering induced by  $c_1, c'_2$ .
  3. Repeat step 2 assuming the center to be located is to the left of  $c_1$ .
  4. Return the solution which has minimum cost.
- 

Fig. 2. The approximate center location on a line algorithm.

1 through the first center and making a small angle with each other, such that at least one of the lines is 1  
2 close enough, in orientation, to the actual line joining the optimal centers. For each of the lines in the 2  
3 covering above and the corresponding approximation to the centroid, we solve the *center location on a* 3  
4 *line problem*, by calling algorithm CenterLocation (we will later show that by restricting the other center 4  
5 to lie on this line, we only suffer by a small factor in our approximation). 5

6 Algorithm CenterLocation solves the *approximate center location on a line* problem on each of the 6  
7 candidate lines constructed above (we will later show that it suffices to solve the approximation version 7  
8 of this problem, suffering only by a small factor in our approximation). We first project all the points 8  
9 onto this line, converting it to a problem in one dimension after transformation. Since there are only two 9  
10 directions to the line, the smaller cluster must lie either to the right or to the left. We repeat the remaining 10  
11 steps for each of the two directions. We solve the problem, by guessing the size of the smaller cluster 11  
12 within a constant factor. For each of the  $O(\log n)$  possible guesses, we again take a random sample from 12  
13 the points on that side of the line and obtain a set of points that contains an approximation to the centroid 13  
14 of the smaller cluster, with constant probability. 14

15 We obtain the costs of the clusterings for all possible pairs formed in this manner and report the 15  
16 clustering with the least cost. We will show that this entire procedure can be performed in time linear 16  
17 in  $n$ . 17

18 For ease of readability, we suppress the linear dependence on the dimension  $d$  when we analyze the 18  
19 algorithm, viz., linear time in our case means  $O(nd)$  since each point has  $d$  coordinates. However, we do 19  
20 state the dependence on the terms depending exponentially on  $d$ . 20

21 The proofs of correctness for the algorithms are detailed in Section 4. 21  
22

23 **Remark 3.1.** The centroid of a set of points,  $P$  in  $\mathfrak{R}^d$ , is the linear combination of the centroids of any 23  
24 partitioning of the point set. Having obtained an approximation to the centroid of the larger cluster of a 2-  
25 means clustering, say  $c$ , it would have been interesting if we could limit our search for the approximation  
26 to the other centroid to the line joining this centroid,  $c$ , to the centroid of the point set  $P$ . However,  
27 since we are dealing with the squares of the distances, it is not possible to get a decent bound on the  
28 approximation obtained by restricting ourselves to this line. 28  
29

## 30 4. Proof of correctness 30 31

### 32 4.1. Reduction to approximate center location on lines 32 33

34 In this section we prove the correctness of the 2-means algorithm and show that we can reduce the 34  
35 *approximate 2-means clustering* problem to a set of *approximate center location on a line* problems 35  
36 so that the solution to one of these problems will correspond to a solution to the *approximate 2-means*  
37 *clustering* problem, with constant probability. 37  
38

39 The following lemma shows that we can obtain an approximation to the centroid of the larger cluster 39  
40 by random sampling. 40  
41

42 **Lemma 4.1.** Let  $(P_1, P_2)$  be the optimal 2-means clustering of  $P$  and let  $(c_1, c_2)$  be the corresponding 42  
43 centroids. Without loss of generality, assume  $|P_1| \geq |P_2|$ . Let  $R$  be a random sample of size  $8/\alpha$  obtained  
44 by independent draws at random from  $P$ . Let  $T$  be the set of centroids of all possible subsets of  $R$  of size 44

1  $2/\alpha$ . Then with constant probability,  $T$  contains a point,  $c'_1$ , that is an  $\alpha$ -approximation of the centroid, 1  
2  $c_1$ , i.e.,  $\Delta(P_1, c'_1) \leq (1 + \alpha)\Delta(P_1, c_1)$ . 2

3 Moreover,  $|T| = O((1/\alpha)^{O(1/\alpha)})$ . 3  
4 4

5 **Proof.** As  $|P_1| \geq |P_2|$ , with constant probability, the sample  $R$  contains at least  $2/\alpha$  points of  $P_1$ . Hence, 5  
6 by Lemma 2.1, with constant probability,  $T$  contains an  $\alpha$ -approximation to the centroid,  $c_1$ , of  $P_1$ . 6  
7 Clearly, the size of  $T$  is  $O((1/\alpha)^{O(1/\alpha)})$ .  $\square$  7  
8 8

9 The following lemma shows that if we have a point  $c'_1$  that is an  $\alpha$ -approximation to the centroid  $c_1$  9  
10 of one of the clusters in an optimal 2-means clustering, then the cost of the optimal 2-means clustering 10  
11 induced by  $c'_1$  is within a factor of  $(1 + \alpha)$  of the cost of the optimal 2-means clustering. 11  
12 12

13 **Lemma 4.2.** Let  $(P_1, P_2)$  be the optimal 2-means clustering of  $P$  and let  $(c_1, c_2)$  be the corresponding 13  
14 centroids. Let  $c'_1$  be an  $\alpha$ -approximation to the centroid  $c_1$ . Let  $c'_2$  be the other center in the optimal 14  
15 2-means clustering  $(P'_1, P'_2)$  of  $P$  when center  $c'_1$  is fixed. Then 15  
16 16

$$\Delta(P'_1, c'_1) + \Delta(P'_2, c'_2) \leq (1 + \alpha)(\Delta(P_1, c_1) + \Delta(P_2, c_2)).$$

17 17  
18 18  
19 **Proof.** As  $c'_1$  is an  $\alpha$ -approximation to the centroid  $c_1$ , we have, 19  
20 20

$$\Delta(P_1, c'_1) + \Delta(P_2, c_2) \leq (1 + \alpha)\Delta(P_1, c_1) + \Delta(P_2, c_2) \leq (1 + \alpha)(\Delta(P_1, c_1) + \Delta(P_2, c_2)).$$

21 21  
22 Also, by the optimality of the partitioning  $(P'_1, P'_2)$  when the center  $c'_1$  is fixed, we know that 22  
23 23

$$\Delta(P'_1, c'_1) + \Delta(P'_2, c'_2) \leq \Delta(P_1, c'_1) + \Delta(P_2, c_2).$$

24 24  
25 Therefore, 25  
26 26

$$\Delta(P'_1, c'_1) + \Delta(P'_2, c'_2) \leq (1 + \alpha)(\Delta(P_1, c_1) + \Delta(P_2, c_2)). \quad \square$$

27 27  
28 28  
29 The following lemma shows that when a center  $c_1$  is fixed, and  $(P_1, P_2)$  is the optimal 2-means clus- 29  
30 tering induced by  $c_1$ , and  $c_2$  is the centroid of  $P_2$ , then, instead of finding  $c_2$ , it suffices to find a suitable 30  
31 point on a line that makes a small angle with the line joining  $c_1$  and  $c_2$  at the cost of a small factor in our 31  
32 approximation. 32  
33 33

34 **Lemma 4.3.** Let  $c_1$  and  $c_2$  be points in  $d$ -dimensional Euclidean space,  $\mathbb{R}^d$ . Let  $(P_1, P_2)$  be the clustering 34  
35 induced by  $(c_1, c_2)$ . Let  $\ell$  be the line joining  $c_1$  and  $c_2$ . Let  $0 < \beta \leq 1$  be a given constant. Let  $\ell'$  be a line 35  
36 passing through  $c_1$  that makes angle  $\beta/64$  with  $\ell$ . Let  $(P'_1, P_3)$  be the optimal 2-means clustering induced 36  
37 by  $c_1$  when the other center is also constrained to lie on  $\ell'$ . Let  $c_3$  be the optimal center corresponding 37  
38 to  $P_3$ . Further let  $c'_3$  be any other point lying on  $\ell'$ , such that, 38  
39 39

$$\Delta(P''_1, c_1) + \Delta(P'_3, c'_3) \leq \left(1 + \frac{\beta}{64}\right)(\Delta(P'_1, c_1) + \Delta(P_3, c_3)) \quad (1)$$

40 40  
41 41  
42 where  $(P''_1, P'_3)$  is the clustering induced by  $(c_1, c'_3)$ . Then, 42  
43 43

$$\Delta(P''_1, c_1) + \Delta(P'_3, c'_3) \leq (1 + \beta)(\Delta(P_1, c_1) + \Delta(P_2, c_2)).$$

**Proof.** Let  $c'_2$  be the projection of  $c_2$  on  $\ell'$  and let  $(P_1''', P_2')$  be the clustering induced by  $(c_1, c'_2)$ . By the optimality of the centroids  $(c_1, c_3)$  on the line  $\ell'$ ,

$$\Delta(P_1', c_1) + \Delta(P_3, c_3) \leq \Delta(P_1''', c_1) + \Delta(P_2', c'_2). \quad (2)$$

In the right triangle,  $\Delta c_1 c_2 c'_2$ ,  $\angle c_2 c_1 c'_2 \leq \frac{\beta}{64}$  (see Fig. 1). Since  $\tan \theta \leq 2\theta$  for  $\theta \leq 1/2$ , we have,  $c_2 c'_2 / c_1 c_2 \leq \beta/32$  and  $c_2 c'_2 / c_1 c'_2 \leq \beta/32$ . This implies that the pairs of points  $(c_1, c_2)$  and  $(c_1, c'_2)$  are  $\beta/32$ -near. Therefore, by Lemma 2.2,

$$\Delta(P_1''', c_1) + \Delta(P_2', c'_2) \leq \left(1 + \frac{\beta}{2}\right) (\Delta(P_1, c_1) + \Delta(P_2, c_2)). \quad (3)$$

From Eqs. (1), (2) and (3), it follows that

$$\begin{aligned} \Delta(P_1'', c_1) + \Delta(P_3', c'_3) &\leq \left(1 + \frac{\beta}{64}\right) \left(1 + \frac{\beta}{2}\right) (\Delta(P_1, c_1) + \Delta(P_2, c_2)) \\ &\leq (1 + \beta) (\Delta(P_1, c_1) + \Delta(P_2, c_2)). \quad \square \end{aligned}$$

The following lemma shows that given a point  $p$ , it is possible to construct a set of  $O((d/\theta)^d)$  lines that is a  $(p, \theta)$ -covering.

**Lemma 4.4.** *Given a point  $p$  in  $d$ -dimensional Euclidean space,  $\mathbb{R}^d$ , and an approximation angle  $0 < \theta \leq 1$ , we can construct a set  $\mathcal{Y}$  of  $O((d/\theta)^d)$  lines passing through  $p$ , in  $O((d/\theta)^d)$  time, such that given any line,  $\ell$ , passing through  $p$ , there exists at least one line in  $\mathcal{Y}$  that makes an angle at most  $\theta$  with  $\ell$ .*

**Proof.** Consider a unit  $d$ -hypercube centered at a point  $p$ . The  $d$  hypercube has  $2d$  surfaces which are  $(d-1)$ -hypercubes. Consider the partitioning of each of these  $(d-1)$ -hypercubes into smaller equal  $(d-1)$ -hypercubes of side length  $x$ . Thus we get a set,  $\mathcal{C}$  of  $O((1/x)^{d-1})$   $(d-1)$ -hypercubes.

Consider the set,  $\mathcal{L}$ , of lines passing through the corners of the hypercubes in  $\mathcal{C}$  and the point  $p$ . Clearly the largest angle between any two lines is formed between the lines passing through the opposite two corners of the hypercube located at the center of any surface (side of the  $d$ -hypercube) as this is the closest  $(d-1)$ -hypercube to  $p$ . Thus in order to obtain a line covering of angle at most  $\theta$ , it suffices to set  $x$  small enough to ensure that the angle between these pair of lines is less than  $\theta$ .

It can be verified that setting  $x = \theta/\sqrt{d}$  satisfies this condition, as  $\tan \theta \geq \theta$  for  $0 < \theta \leq 1$ . Therefore we obtain a set,  $\mathcal{L}$ , of lines of size  $O((d/\theta)^d)$ .

Fig. 3 illustrates such a line covering for  $d = 2$ . The same can be extended to higher dimensions.

Clearly the time required to compute this set of lines is of the same order as its size.  $\square$

The following lemma proves the correctness of the 2-means algorithm.

**Lemma 4.5.** *Given an algorithm CenterLocation that determines a  $(1 + \delta)$ -approximate solution to the center location on a line problem in time  $\mathcal{T}(n, \delta)$ , with constant probability, where  $n$  is the number of input data points; the algorithm 2-means determines a  $(1 + \varepsilon)$ -approximate 2-means clustering, with constant probability, in time  $O((1/\varepsilon)^{O(1/\varepsilon)} (d/\varepsilon)^d \cdot \mathcal{T}(n, \varepsilon/4))$ .*

**Proof.** Let  $(P_1, P_2)$  be the optimal 2-means clustering and  $(c_1, c_2)$  be the corresponding centroids.

Step 1 of algorithm 2-means sets  $\alpha, \theta$  and  $\delta$  to suitable constants (fractions of  $\varepsilon$ ).

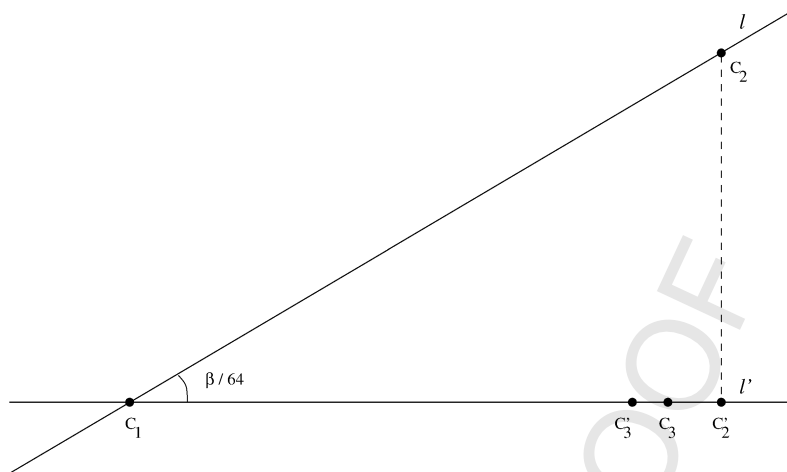


Fig. 3. Approximating the line connecting the centers.  $c'_2$  is the projection of  $c_2$  on  $l'$ .  $c_3$  is the optimal on the line  $l'$ .

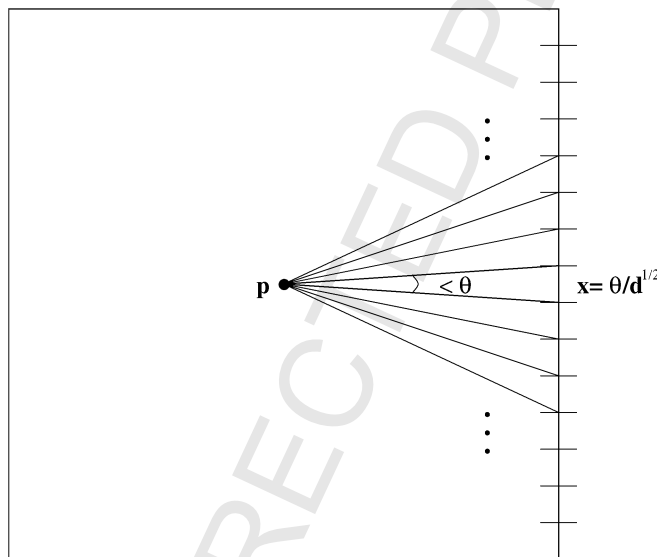


Fig. 4. A  $(p, \theta)$ -covering in 2-dimensions ( $d = 2$ ). (Not all the lines of the covering are shown in the figure.) The largest angle is formed with the hypercube at the center of the surface.

Applying Lemma 4.1, step 2 of the algorithm determines a set  $T$  of  $O((1/\alpha)^{O(1/\alpha)})$  points, such that at least one of these points is an  $\alpha$ -approximation to the centroid of the larger cluster, with constant probability. Let  $c'_1$  be such an  $\alpha$ -approximation (if it exists).

Step 3 (a) of the algorithm determines a  $(p, \theta)$ -covering  $\mathcal{Y}$  for every center,  $p$ , in  $T$  (including  $c'_1$ ). Applying Lemma 4.4, such a covering set of lines can be constructed in time  $O((d/\theta)^d)$  and has size  $O((d/\theta)^d)$ . Let  $\beta = 64\theta$ . Then, we have  $O((d/\beta)^d)$  lines passing through  $p$ , such that for any line passing through  $p$ , we have a line making an angle at most  $\beta/64$ .

1 In all, we have  $O((d/\beta)^d)$  lines passing through each of the  $O((1/\alpha)^{O(1/\alpha)})$  candidate centers, resulting 1  
2 in a total of  $O((1/\alpha)^{O(1/\alpha)}(d/\beta)^d)$  lines. One of these lines, with constant probability, passes through  $c'_1$ , 2  
3 and makes an angle of less than  $\beta/64$  with the line joining  $c'_1$  and  $c_2$  where  $c_2$  is the other center of the 3  
4 optimal 2-means clustering, when the first center,  $c'_1$  is fixed. 4

5 Therefore, applying Lemmas 4.2 and 4.3, the solution to the  $(1 + \delta)$  approximate center location on 5  
6 a line problem on this line, computed in step 3 – (b) of the algorithm, gives us a  $(1 + \alpha)(1 + \beta)(1 +$  6  
7  $\delta)$ -approximate 2-means clustering of  $P$ . With the appropriate choices of  $\alpha$ ,  $\beta$  and  $\delta$  (as specified in 7  
8 step 1 of the algorithm), we have a set of  $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d)$  lines, such that, the solution of the  $(1 +$  8  
9  $\varepsilon/4)$ -approximate center location on a line problem on one of them gives us a solution to the  $(1 + \varepsilon)$ - 9  
10 approximate 2-means clustering of  $P$ , with constant probability. 10

11 Since we are reporting the minimum cost clustering in step 4 of the algorithm, it is a  $(1 + \varepsilon)$ -approximate 11  
12 2-means clustering, with constant probability.  $\square$  12

#### 13 4.2. Solving the problem for center location on a line 14

15 In this section we prove the correctness of the algorithm CenterLocation and show that we can solve 16  
17 the approximate center location on a line problem in linear time. 17

18 In the following lemma, we show that the center location on a line problem is reducible to one dimen- 18  
19 sion. 19

20 **Lemma 4.6.** Let  $P$  be a set of points in  $d$ -dimensional Euclidean space,  $\mathbb{R}^d$ . Let  $\ell$  be a given line and let 21  
22  $c_1$  be a fixed point on  $\ell$ . Let  $P'$  be the set of points obtained by projecting the points of  $P$  on  $\ell$ . Consider 22  
23 the problem  $\mathcal{P}$ , of locating a point  $c_2$  on  $\ell$  such that the clustering of  $P'$  induced by  $(c_1, c_2)$  is optimal 23  
24 when  $c_1$  and  $c_2$  are constrained to lie on  $\ell$  with  $c_1$  fixed. 24

25 Then, the problem  $\mathcal{P}$  is equivalent to the center location on a line problem. Moreover, an approximate 25  
26 solution to  $\mathcal{P}$  is also an approximate solution to the center location on a line problem within the same 26  
27 approximation factor. 27

28 **Proof.** Consider the transformation of the original axis to an orthogonal axis in which one of the axis 29  
30 is parallel to  $\ell$ . Let the point  $p_i \in P$  be denoted by the vector  $(x_{i1}, x_{i2}, \dots, x_{id})$  on the transformed 30  
31 orthogonal axis with  $x_{i1}$  being the contribution towards the axis parallel to line  $\ell$ . Let  $c_1$  and  $c_2$  be 31  
32 represented by the vectors  $(c_{11}, c_{12}, \dots, c_{1d})$  and  $(c_{21}, c_{22}, \dots, c_{2d})$  respectively on the transformed axis. 32  
33 Then the cost of partition  $(P_1, P_2)$  with centers  $(c_1, c_2)$  can be computed as 33

$$34 \Delta = \sum_{p_i \in P_1} \sum_{j=1}^d (x_{ij} - c_{1j})^2 + \sum_{p_i \in P_2} \sum_{j=1}^d (x_{ij} - c_{2j})^2. \quad 34$$

35 Now, since  $c_1$  lies on  $\ell$  and  $c_2$  is also constrained to lie on  $\ell$ , therefore minimization of the above cost 35  
36 function can be reduced to the axis parallel to  $\ell$  as follows: 36

$$37 \Delta_1 = \sum_{p_i \in P_1} (x_{i1} - c_{11})^2 + \sum_{p_i \in P_2} (x_{i2} - c_{22})^2. \quad 37$$

38 This is because, the difference in the cost of a point  $p_i \in P$  towards two distinct centers constrained 38  
39 to lie on line  $\ell$  is only affected by the component along the axis parallel to  $\ell$ . The cost incurred along all 39  
40 the remaining axis remains unaffected. 40  
41  
42  
43  
44

Moreover, we also claim that a  $(1 + \delta)$ -approximate clustering on the single dimension problem corresponds to a  $(1 + \delta)$ -approximate clustering in the original space.

Let  $(P'_1, P'_2)$  be a  $(1 + \delta)$ -approximate clustering in a single dimension (axis parallel to  $\ell$ ) with centers  $(c_1, c'_2)$  both lying on  $\ell$ . Then

$$\Delta((P'_1, c_1) + \Delta(P'_2, c'_2)) \leq (1 + \delta)(\Delta(P_1, c_1) + \Delta(P_2, c_2)),$$

i.e.,

$$\sum_{p_i \in P'_1} (x_{i1} - c_{11})^2 + \sum_{p_i \in P'_2} (x_{i2} - c'_{22})^2 \leq (1 + \delta) \left( \sum_{p_i \in P_1} (x_{i1} - c_{11})^2 + \sum_{p_i \in P_2} (x_{i2} - c_{22})^2 \right).$$

Therefore,

$$\begin{aligned} & \sum_{p_i \in P'_1} \sum_{j=1}^d (x_{ij} - c_{1j})^2 + \sum_{p_i \in P'_2} \sum_{j=1}^d (x_{ij} - c'_{2j})^2 \\ & \leq (1 + \delta) \left( \sum_{p_i \in P_1} \sum_{j=1}^d (x_{ij} - c_{1j})^2 + \sum_{p_i \in P_2} \sum_{j=1}^d (x_{ij} - c_{2j})^2 \right) \end{aligned}$$

as components along all other dimensions are unaffected.  $\square$

We state here some observations that will be useful in our next proof.

**Observation 4.1.** Given two points  $(c_1, c_2)$ , let  $\mathcal{H}$  be the perpendicular bisector of these points. This hyperplane gives the optimal clustering with fixed centers  $(c_1, c_2)$ . As we move a hyperplane parallel to  $\mathcal{H}$  in either direction, the cost function of the clustering due to the hyperplane with fixed centers  $(c_1, c_2)$  is a monotonic non-decreasing function. This is clear since, as we move the hyperplane, the cost remains unchanged till we cross a point. When we cross a point, we disassociate this point from its closer center and associate it to the further center and this can only result in an increase in the cost of the clustering.

**Observation 4.2.** Given a set of points, along with the centroid of the set and the cost of the cluster, on addition or removal of a point to/from this set, we can compute the new centroid and the cost of the new cluster in constant time. If  $\mathbf{x}$  is the old centroid of a set  $P$  of  $n$  points and  $\mathbf{x}_{n+1}$  is a new point, so that  $P' = P \cup \{\mathbf{x}_{n+1}\}$  then the new centroid  $\mathbf{x}_{\text{new}}$  is obtained as follows:

$$\mathbf{x}_{\text{new}} = \frac{n}{n+1} \mathbf{x} + \frac{1}{n+1} \mathbf{x}_{n+1},$$

and the cost of the clustering at  $\mathbf{x}_{\text{new}}$  is obtained as follows:

$$\Delta(P', \mathbf{x}_{\text{new}}) = \Delta(P, \mathbf{x}) + n \Delta(\mathbf{x}, \mathbf{x}_{\text{new}}) + \Delta(\mathbf{x}_{n+1}, \mathbf{x}_{\text{new}}).$$

It follows that given a 2-partitioning of  $P$  along with the centroids and the cost of the clustering due to the two partitions, if we move one point from one partition to another partition, then the cost of the new clustering and the new centroids can be determined in constant time.

**Remark 4.1.** The Center Location on a line problem is solvable exactly by traversing the points from left to right one at a time and updating the cost of the clustering using the above observation. This can

1 be done in two passes, first assuming the center lies to the right and then assuming that it lies to the left. 1  
2 However, note that this requires  $O(n \log n)$  time as the points have to be sorted before the traversal can 2  
3 be performed. Our goal is to achieve linear running time. 3  
4

5 In the following lemma, we prove the correctness of the algorithm CenterLocation and determine its 5  
6 running time. 6  
7

8 **Lemma 4.7.** *Let  $P$  be a set of points in  $d$ -dimensional Euclidean space,  $\mathbb{R}^d$ , and  $\ell$  be a given line. Let  $c_1$  8  
9 be a fixed point on  $\ell$ . Then, algorithm CenterLocation determines in time  $O((1/\delta)^{O(1/\delta)}n)$ , a point  $c'_2$  such 9  
10 that with constant probability, the cost of the clustering induced by  $(c_1, c'_2)$  is within a factor of  $(1 + \delta)$  10  
11 of the cost of the clustering induced by  $(c_1, c_2)$ , i.e., 11*

$$\Delta(P'_1, c_1) + \Delta(P'_2, c'_2) \leq (1 + \delta)(\Delta(P_1, c_1) + \Delta(P_2, c_2))$$

12 where  $c_2$  is the point on  $\ell$  for which the clustering of  $P$  induced by  $(c_1, c_2)$  is optimal when  $c_1$  is fixed, 12  
13  $(P_1, P_2)$  is the corresponding partitioning of the points and  $(P'_1, P'_2)$  is the partitioning of  $P$  induced by 13  
14  $(c_1, c'_2)$ . 14  
15  
16

17 **Proof.** Step 1 of algorithm CenterLocation projects the points  $P$  onto the line  $\ell$ . Applying Lemma 4.6, 17  
18 the problem of solving the *approximate center location on a line* problem is equivalent to solving the 18  
19 approximate version of the problem after projecting the points on  $\ell$ . 19  
20

21 Step 2 of the algorithm assumes the center of the smaller cluster lies to the right of  $c_1$  (The case when 21  
22 the center lies to the left of  $c_1$  is handled similarly in step 3.) In each iteration half the remaining points 22  
23 from the left side are eliminated as in step 2(a) of the algorithm. Suppose in the current iteration we have 23  
24 a set  $M$  of  $m$  points. Initially  $m = n$ . We first locate the mid-point in  $M$ , i.e.,  $m/2$ th point, say  $q$ . This 24  
25 takes  $O(m)$  time and partitions the point set,  $M$ , into two sets  $M_L$  and  $M_R$ .  $M_R$  is the remaining points 25  
26 after elimination. Thus,  $M_R$  can be obtained from the points remaining in the previous iteration in  $O(m)$  26  
27 time. 27

28 Clearly, in some iteration, the number of points remaining,  $|M_R|$ , will be at most twice the size of  $P_2$  28  
29 and  $P_2$  will be contained in  $M_R$ . Also, since the number of points are being halved in each iteration, the 29  
30 number of iterations required is at most  $\log n$ . 30

31 Assuming that we have the right partition, in step 2(b) of the algorithm, we randomly sample a set 31  
32  $R$  of  $8/\delta$  points from  $M_R$ . Note that although we only require  $2/\delta$  points to approximate the centroid 32  
33 (Lemma 2.1), we take ample points in order to sample  $2/\delta$  points from  $P_2$  with constant probability, 33  
34 since we are approximating its size within a factor of 2. In step 2(c) of the algorithm, we now take all 34  
35 possible subsets of the random sample  $R$  of size  $2/\delta$  and compute their centroids. Note that there are 35  
36  $O((1/\delta)^{O(1/\delta)})$  such points. With constant probability, one of these is a  $\delta$ -approximation to the centroid 36  
37 of  $P_2$ , i.e., 37

$$\Delta(P_2, c'_2) \leq (1 + \delta)\Delta(P_2, c_2).$$

38 Therefore, 38  
39

$$\begin{aligned} \Delta(P'_1, c_1) + \Delta(P'_2, c'_2) &\leq \Delta(P_1, c_1) + \Delta(P_2, c'_2) \leq \Delta(P_1, c_1) + (1 + \delta)\Delta(P_2, c_2) \\ &\leq (1 + \delta)(\Delta(P_1, c_1) + \Delta(P_2, c_2)). \end{aligned}$$

40 In step 2(d) of the algorithm, we compute the cost of the clustering. We now show that the clustering 40  
41 cost can be computed in time linear in the number of remaining points. In each iteration, we are dealing 41  
42  
43  
44

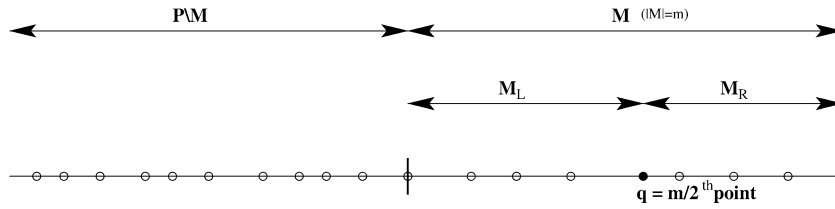


Fig. 5. Sampling recursively by guessing the size of  $|P_2|$ .

with half the points from the previous iteration. Let  $q$  be the mid point of  $M$  as above. The cost of the clustering obtained by partitioning at  $q$  can be determined in  $O(m)$  time. This is possible because we already know the cost of the clustering due to  $(M, P \setminus M)$  from the previous iteration and therefore we can partition the set  $M$  into  $M_L$  and  $M_R$  in  $O(m)$  time and then apply Observation 4.2 over the set  $M_L$ . Note that in the current iteration we perform the sampling necessary for the guess  $m/4 \leq |P_2| \leq m/2$ . Let  $\mathcal{H}'$  be the corresponding perpendicular bisecting line. If  $\mathcal{H}'$  lies to the right of  $q$ , as we have already computed the cost of the clustering due to the partitioning at  $q$ , we only need to alter the cost over the remaining points in  $M_R$  which is at most  $m/2$  and can be done in time  $O(m)$  by applying Observation 4.2 at most  $m/2$  times. If  $\mathcal{H}'$  is to the left of  $q$ , let  $\mathcal{H}$  be the optimal line bisecting the line joining the centroids  $(c_1, c_2)$ . In our current guess,  $m/4 \leq |P_2| \leq m/2$ . Therefore, the line  $\mathcal{H}$  must lie to the right of  $q$ . Due to Observation 4.1, the cost of the clustering due to the line  $\mathcal{H}'$ , passing through  $q$  and parallel to  $\mathcal{H}$ , must be between that of  $\mathcal{H}$  and  $\mathcal{H}'$ . Thus, if  $\mathcal{H}'$  defines a  $(1 + \delta)$ -approximate 2-means clustering, then so does  $\mathcal{H}''$  for which we already know the cost. Hence we have restricted ourselves to altering the cost known at  $q$  only over the set  $M_R$  for each of the candidate centers. Hence the time required to compute the cost of the clustering in each iteration is  $O(m)$ .

As we are always recurring into the right half, the running time of this algorithm is characterized by the recurrence relation,  $T(n) = T(n/2) + cn$ , which is linear in  $n$ . Performing the same steps in the other direction in step 3 of the algorithm at most doubles the running time.

In step 4, we return the minimum of the costs of the clusterings due to the pairs formed from these centers with  $c_1$  fixed. This gives us a  $(1 + \delta)$ -approximation to the center location on the line problem, with constant probability.  $\square$

Putting everything together, we obtain the main result of the paper:

**Theorem 4.8.** Algorithm 2-means, using algorithm CenterLocation, determines a  $(1 + \varepsilon)$ -approximate 2-means clustering of a point set  $P$  in  $\mathbb{R}^d$  in time  $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d n)$ , with constant probability.

**Proof.** By Lemma 4.7, algorithm CenterLocation solves the  $(1 + \delta)$ -approximate center location on a line problem in time  $O((1/\delta)^{O(1/\delta)} n)$ .

Therefore, applying Lemma 4.5 and using the fact that the algorithm CenterLocation is invoked with  $\delta = \varepsilon/4$ , we conclude that the algorithm 2-means solves the approximate 2-means clustering problem in time  $O((1/\varepsilon)^{O(1/\varepsilon)}(d/\varepsilon)^d n)$ .  $\square$

**Acknowledgement**

The authors would like to thank Pankaj Agarwal for his useful feedback regarding improving the readability of the paper.

**References**

- [1] A. Broder, S. Glassman, M. Manasse, G. Zweig, Syntactic clustering of the web, in: Proc. of 6th International World Wide Web Conference, 1997.
- [2] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, R.A. Harshman, Indexing by latent semantic analysis, *J. Amer. Soc. Inform. Sci.* 41 (6) (1990) 391–407.
- [3] Drineas, Frieze, Kannan, Vempala, Vinay, Clustering in large graphs and matrices, in: SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms), 1999.
- [4] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, W. Equitz, Efficient and effective querying by image content, *J. Intell. Inf. Syst.* 3 (3–4) (1994) 231–262.
- [5] M.R. Garey, D.S. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, New York, 1979.
- [6] S. Har-Peled, S. Mazumdar, On coresets for k-means and k-median clustering, in: Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, ACM Press, New York, 2004, pp. 291–300.
- [7] M. Inaba, N. Katoh, H. Imai, Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering (extended abstract), in: Proceedings of the Tenth Annual Symposium on Computational Geometry, ACM Press, New York, 1994, pp. 332–339.
- [8] A. Kumar, Y. Sabharwal, S. Sen, A simple linear time  $(1 + \epsilon)$ -approximation algorithm for k-means clustering in any dimensions, in: Proceedings of the 45th Annual Symposium on Foundations of Computer Science, IEEE Computer Society, 2004, pp. 454–462.
- [9] J. Matousek, On approximate geometric k-clustering, *Discrete Comput. Geom.* 24 (1) (2000) 61–84.
- [10] N. Megiddo, K.J. Supowit, On the complexity of some common geometric location problems, *SIAM J. Comput.* 13 (1) (1984) 182–196.
- [11] M.J. Swain, D.H. Ballard, Color indexing, *Internat. J. Computer Vision* (1991) 11–32.