Fitting Flats To Points With Outliers

Guilherme D. da Fonseca[*]

Appeared at International Journal of Computational Geometry and Applications, 21(5):559-569, 2011. A preliminary version was presented at the 26th European Workshop on Computational Geometry, 2010.

Abstract

Determining the best shape to fit a set of points is a fundamental problem in many areas of computer science. We present an algorithm to approximate the k-flat that best fits a set of n points with $n - m$ outliers. This problem generalizes the smallest m-enclosing ball, infinite cylinder, and slab. Our algorithm gives an arbitrary constant factor approximation in $O(n^{k+2}/m)$ time, regardless of the dimension of the point set. While our upper bound nearly matches the lower bound, the algorithm may not be feasible for large values of k. Fortunately, for some practical sets of inliers, we reduce the running time to $O(n^{k+2}/m^{k+1})$, which is linear when $m = \Omega(n)$.

1 Introduction

Determining the best shape to fit a set of points is a fundamental problem in statistics, machine learning, data mining, computer vision, clustering, metrology, and assembly planning. Recently, the problem received considerable attention in the computational geometry literature [1, 3, 5–8, 11, 13–15, 17–19]. The case of fitting a lower-dimensional space is particularly important since it can be used to minimize the effects of the curse of dimensionality when the points have low intrinsic dimension.

A widely used measure of how well a shape S fits a set P of n points in d-dimensional space is $\max_{p \in P} \min_{s \in S} \|ps\|$, the maximum Euclidean distance between any point $p \in P$ and the shape S. Unfortunately, this measure is very sensitive to the presence of outliers. In this paper, we consider a more robust measure in the presence of $n - m$ outliers and m inliers. The measure consists of minimizing the following *cost* function: given a parameter $m \leq n$, the cost is the m-th smallest distance between a point in P and the shape S.

Our results. We consider an approximation to the case when S is a k-flat, for a given value of $k \in \{0, \ldots, d-1\}$. We show that, for an arbitrary $\varepsilon > 0$, we can find

[*] Universidade Federal do Estado do Rio de Janeiro (UNIRIO), Rio de Janeiro, RJ, Brazil, fonseca@uniriotec.br. Research supported by FAPERJ grant E26-110.091/2010 and CNPq/FAPERJ grant E26-110.552/2010.

in $O_\varepsilon(n^{k+2}/m)$ time[1] a k-flat S with cost at most $1+\varepsilon$ times the optimum. We refer to this problem as *flat fitting*. We assume that the dimensions k, d are constants, but $1/\varepsilon$ is an asymptotic quantity. It is noteworthy that the complexity depends only on the target dimension k, regardless of the dimension d of the point set. Our algorithm is Monte Carlo, but can be made deterministic at the expense of an $O(m)$ factor in the running time.

In the most interesting case when m is a constant fraction of n, the running time of our Monte Carlo algorithm is $O_\varepsilon(n^{k+1})$. While our upper bound is close to the $\Omega(n^k)$ lower bound, the algorithm is still super-linear for $k \geq 1$. Algorithms for robust estimators that benefit from well-behaved sets of inlier are presented in [17, 18], and evidence that some practical data sets resemble to points uniformly distributed on a lower dimensional flat is suggested in [16]. Informally, we say that the set of inliers is *outer-dense* if any halfspace with normal vector v that contains $1/4$ of the width of the point set in direction v also contains a constant fraction of the inliers. We show that, if the set of inliers is outer-dense, then flat fitting can be solved in time $O_\varepsilon(n^{k+2}/m^{k+1})$, which is linear for $m = \Omega(n)$. Point sets uniformly distributed in a convex region or on the boundary of a convex region are outer-dense with high probability. Consequently, despite the high worst-case complexity of the problem, there is a feasible solution for some practical large data sets.

Related work. The case of $k = 0$ corresponds to the well-studied problem of approximating the smallest ball enclosing m points [8, 11, 13]. The first linear time solution is presented in [13]. The ε-dependencies can be improved to an expected running time of $O(n/\varepsilon^{d-1})$ by using techniques from [4, 8]. Existing algorithms for $k = 0$ rely on the fact that balls are bounded fat objects and therefore these algorithms do not generalize to larger values of k. An easier variation of the problem, where an inlier is known, is used as a base case for our algorithm.

The case of $k = d - 1$ corresponds to approximating the narrowest slab enclosing m points. In contrast to the linear complexity for $k = 0$, the most efficient solution for $k = d - 1$ is a high probability Monte Carlo algorithm [7] with running time $O(n^d(\log^{O(1)} \frac{1}{\varepsilon})/m\varepsilon)$. Major improvements are unlikely, since there is a lower bound of $\Omega((n-m)^{d-1} + (n/m)^d)$ for obtaining a constant approximation [7], assuming a conjecture for the affine degeneracy problem holds. A related problem is the Least Median of Squares (LMS) estimator, where the vertical distance is minimized, instead of the Euclidean distance [3, 7]. Algorithms for $k = d - 1$ use point-hyperplane duality and arrangements, and therefore cannot be generalized for other values of k.

The case of $k = 1$ corresponds to approximating the smallest infinite cylinder enclosing m points, which is stated as an open problem by Har-Peled and Mazumdar [13]. A linear time solution for arbitrary values of m is unlikely, since even the planar approximation problem is 3SUM-hard [12]. To see that, note that it is 3SUM-hard to decide if there are three points on a line and that there is a planar cylinder of radius 0 enclosing three points if and only if there are three points on a line.

While we know of no previous approximate algorithm for arbitrary k and d with running time polynomial in $1/\varepsilon$, there are several relevant results for the flat fitting problem

[1] We use the $O_\varepsilon(\cdot)$ notation to hide polynomial ε-dependencies.

under different assumptions. When the number $n-m$ of outliers is small compared to n, we can use the coreset framework to reduce the number of points to $O((n-m)/\varepsilon^{(d-1)/2})$ and then solve the problem for the reduced point set [1], using either exact algorithms or our approximate algorithm. The case when d is an asymptotic variable is considered in [14], where an algorithm with running time linear in d but exponential in $1/\varepsilon$ is presented. Approaches based on random sampling such as RANSAC [9] are widely used in practice, but do not guarantee approximation with respect to the optimum.

The non-robust version of the problem (when $m = n$) is generally approximated using coresets [6]. The case when d is an asymptotic variable is considered in [15]. When $k = 0$, it is well known that the non-robust exact version can be solved in $O(n)$ time. Exact solutions for other values of k are considerably less efficient, even in the non-robust version. Chan [5] mentions an $O(n^{\lceil d/2 \rceil})$ algorithm for $k = d-1$ and an $O(n^{2d-1+\delta})$ algorithm for $k = 1$, where δ is an arbitrarily small constant. When $k = 1$ and $d = 3$, the problem can be solved in $O(n^4 \log^{O(1)} n)$ time [19].

The exact robust version seems even harder. A trivial solution takes $O(n^{(d-k)(k+1)+2})$ time, by counting the number of points for each potential set of up to $(d-k)(k+1)+1$ farthest inliers. When $k = d-1$, the problem can be solved in $O(n^d)$ expected time [3, 7], improving the trivial solution by a factor of $O(n^2)$. When $k = 0$ and $d = 2$, the problem can be solved in $O(nm)$ time [13].

A lower bound of $\Omega((n-m)^{d-1} + (n/m)^d)$ for obtaining a constant approximation when $k = d-1$ in presented in [7]. The lower bound is based on a conjecture that the affine degeneracy problem in d-dimensional space requires $\Omega(n^d)$ time. We can linearly reduce the flat fitting problem with $k = d-1$ to the flat fitting problem in higher dimension $d' \geq d$ and the same value of k. Therefore, the lower bound for $k = d-1$ implies a lower bound of $\Omega((n-m)^k + (n/m)^{k+1})$ for arbitrary k. In the most interesting case when m is a constant fraction of n, the lower bound is $\Omega(n^k)$ and we present an upper bound of $O(n^{k+1})$.

In Section 2, we present approximate algorithms for the flat fitting problem: a Monte Carlo algorithm with running time $O_\varepsilon(n^{k+2}/m)$ and a deterministic algorithm with running time $O_\varepsilon(n^{k+2})$. In Section 3, we show how to reduce the running time of the Monte Carlo algorithm to $O_\varepsilon(n^{k+2}/m^{k+1})$ for some sets of inliers. Concluding remarks and open problems are discussed in Section 4.

2 Approximate Algorithm

The general idea of the algorithm is the following.

1. Find a set of vectors V that contains a vector that is approximately parallel to the best fitting flat.
2. For each vector $v \in V$, project the points onto a hyperplane perpendicular to v.
3. Recursively solve a lower dimensional problems, returning the best solution found, and using $k = 0$ as a base case.

We start by providing some definitions, illustrated in Figure 1(a).

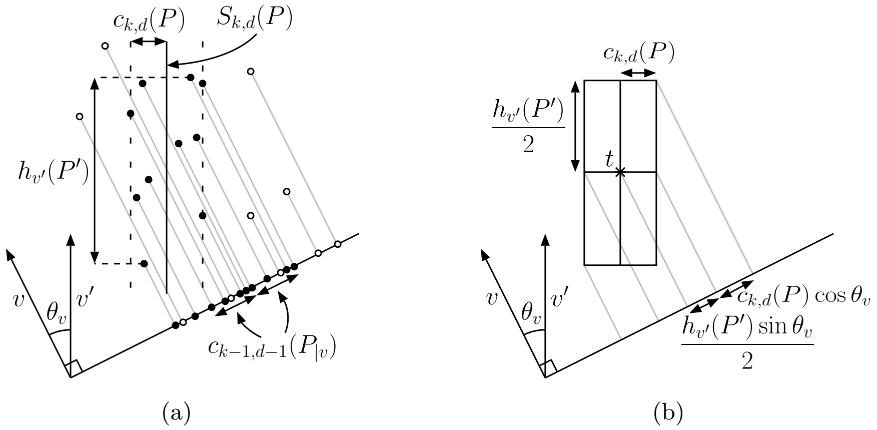

Figure 1: (a) Diagram of definitions. The $m = 10$ inliers are represented by solid circles. (b) Figure for the proof of Lemma 2.1.

Let $S_{k,d}(P)$ and $c_{k,d}(P)$ respectively denote the optimal k-flat for point set P in d-dimensional space and its cost. We refer to the m points $P' \subseteq P$ within distance $c_{k,d}(P)$ of $S_{k,d}(P)$ as *inliers*. Given a d-dimensional set of points P and a vector v, let $P_{|v}$ denote a $(d-1)$-dimensional point set obtained by projecting P onto a hyperplane perpendicular to v. Given a vector v let v' be the unit length projection of v onto the optimal flat $S_{k,d}(P)$, $h_{v'}(P') = \max_{p \in P'} v' \cdot p - \min_{p \in P'} v' \cdot p$ be the directional width in direction v' of the inliers, and θ_v be the acute angle between v and v'. The following lemma shows how to use the solution of a lower dimensional problem in order to approximate the original problem.

Lemma 2.1. *For any vector v we have*

$$c_{k,d}(P) \leq c_{k-1,d-1}(P_{|v}) \leq c_{k,d}(P) + h_{v'}(P')\,\theta_v/2.$$

Proof. To see that $c_{k,d}(P) \leq c_{k-1,d-1}(P_{|v})$, note that a $(k-1)$-flat in $(d-1)$-dimensional space can be extended in direction v creating a k-flat in d-dimensional space with the same cost.

We now show that $c_{k-1,d-1}(P_{|v}) \leq c_{k,d}(P) + h_{v'}(P')\theta_v/2$. Let t the midpoint of the projection of the segment that defines $h_{v'}(P')$ onto $S_{k,d}(P)$ and let S' denote the $(k-1)$-flat obtained by intersecting $S_{k,d}(P)$ with the hyperplane perpendicular to v that passes through t. The distance between S' and $P'_{|v}$ is at most $c_{k,d}(P)\cos\theta_v + h_{v'}(P')\sin\theta_v/2 \leq c_{k,d}(P) + h_{v'}(P')\theta_v/2$ (Figure 1(b)). □

By Lemma 2.1, it is possible to obtain a constant approximation by finding a vector v with angle

$$\theta_v \leq \frac{c_{k,d}(P)}{h_{v'}(P')}$$

and recursively solving the lower dimensional problem. Instead of explicitly finding such vector v, our algorithm builds a set V that contains the desired vector and recursively solves the problem for all $v \in V$. The solution of minimum cost found is therefore a valid approximation. The following lemma is the key to obtain the set of vectors V.

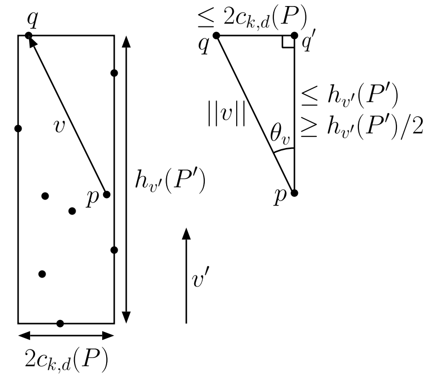

Figure 2: Proof of Lemma 2.2.

Lemma 2.2. *For every inlier $p \in P'$, there is an inlier $q \in P'$ such that the vector $v = q - p$ has*

$$\theta_v \leq \frac{4c_{k,d}(P)}{h_{v'}(P')} \quad \textit{and} \quad \frac{h_{v'}(P')}{2} \leq \|v\| \leq 2c_{k,d}(P) + h_{v'}(P').$$

Proof. Consider the inlier $q \in P'$ that is farthest from p with distances measured between their projections onto the optimal flat. In other words, q is the inlier that maximizes $|v' \cdot p - v' \cdot q|$ (see Figure 2). Let q' denote the projection of q onto the line that passes through p in direction v'. It follows that $h_{v'}(P')/2 \leq \|pq'\| \leq h_{v'}(P')$ and $\|qq'\| \leq 2c_{k,d}(P)$. Consequently, $h_{v'}(P')/2 \leq \|v\| = \|pq\| \leq 2c_{k,d}(P) + h_{v'}(P')$. Also, $\tan(\theta_v) \leq 4c_{k,d}(P)/h_{v'}(P')$. Since $\theta_v \leq \tan(\theta_v)$, the lemma follows. □

Finding an inlier. Before we apply the previous lemmas, we consider the problem of finding a set that contains an inlier. In the deterministic version the set of all n input points is guaranteed to contain an inlier. In the Monte Carlo version, we use the following random sampling technique from [13] to find a set that contains an inlier with constant probability. By definition, the set P contains m inliers. Therefore, a random element of P is an inlier with probability m/n and a random sample of n/m elements of P contains an inlier with probability at least $1 - 1/e$.

Base case. The base case for our algorithm consists of approximating the smallest m-enclosing ball given an inlier p (a point inside the smallest m-enclosing ball). We refer to the time complexity of the base case as $t_{0,d}$. We start by presenting a simple and practical algorithm to solve the problem in $t_{0,d} = O(n + m/\varepsilon^d)$ time.

1. Obtain a 2-approximation a of the optimum radius by finding the m-th farthest point from p.

2. Create a set Q containing the $\Theta(m)$ points within distance $2a$ of p.

3. Consider a grid with cells of diameter εa. Compute the radius of the ball enclosing m points from Q centered at each of the $O(1/\varepsilon^d)$ grid vertices within distance a from p, returning the smallest radius found.

Slightly better ε-dependencies can be obtained by using much more sophisticated techniques. Using binary search and the algorithm from [8] for the decision version of the problem, the running time becomes $t_{0,d} = O(n + m(\log \frac{1}{\varepsilon})/\varepsilon^{d-1})$ with high probability. If we use Chan's randomized optimization [4] instead of binary search, we obtain expected running time $t_{0,d} = O(n + m/\varepsilon^{d-1})$.

Alternatively, we can achieve expected running time $t_{0,d} = O(n + (\log^{d+1} \frac{1}{\varepsilon})/\varepsilon^{d})$ as follows. We reduce the decision problem to an absolute-model fixed-radius spherical range searching data structure as in [11]. We then use the data structure with $O(1)$ query time from [2], preprocessed in $O(n + (\log^{d+1} \frac{1}{\varepsilon})/\varepsilon^{d})$ time by the recursive construction from [10].

For simplicity, we use the slightly weaker bound of $t_{0,d} = O(n/\varepsilon^{d})$ throughout the paper.

Constant approximation. Before we address the $(1+\varepsilon)$-approximation, we present an algorithm that provides a constant factor approximation. The algorithm starts by finding a set I of n/m points that contains an inlier with constant probability or, in the deterministic version, a set of n points that is guaranteed to contain an inlier. For each point $p \in I$, we repeat the following steps, which will be executed recursively. For each point $q \in P$ we obtain a vector $v = q - p$ and project P onto a hyperplane perpendicular to v. The problem is solved recursively with $k \leftarrow k-1$, $d \leftarrow d-1$, and using the projection of p in place of p. The recursion stops by solving the problem directly when $k = 0$.

The running time is the product of $O(n)$ time for the base case, $O(n^{k})$ time for the k recursive calls, and $O(n/m)$ for the set that contains an inlier ($O(n)$ in the deterministic version). Therefore the total running time is $O(n^{k+2}/m)$ in the Monte Carlo version and $O(n^{k+2})$ in the deterministic version. The fact that the algorithm provides a constant factor approximation follows from Lemmas 2.1 and 2.2. Next we show how to reduce the approximation factor to $1+\varepsilon$ for arbitrarily small $\varepsilon > 0$.

$(1+\varepsilon)$-approximation. By Lemma 2.1, if we project the points onto a hyperplane perpendicular to a vector u such that

$$\theta_u \leq \frac{2\varepsilon' c_{k,d}(P)}{h_{u'}(P')} = \phi$$

at each iteration and solve the base case with approximation factor $1+\varepsilon'$, then we obtain a total approximation factor of $(1+\varepsilon')^{k+1}$. Setting $\varepsilon' = (1+\varepsilon)^{1/(k+1)} - 1$, we obtain a $(1+\varepsilon)$-approximation. Note that for $\varepsilon < 1$, we have $\varepsilon' > \varepsilon/2(k+1)$, therefore the asymptotic complexity remains the same. The $(1+\varepsilon)$-approximation algorithm is similar to the constant approximation, except that for each vector considered in the constant approximation, we also consider multiple vectors near the original vector. This way, we obtain a set that contains a vector u with $\theta_u \leq \phi$. Next, we describe how to find a set that contains such vector u, given a vector v satisfying the properties of Lemma 2.2.

Consider a d-dimensional unit hypercube centered at the origin. We refer to the set of vertices of a grid on each hypercube face as a $(d-1)$-*dimensional grid of directions*. If $4c_{k,d}(P) \geq h_{v'}(P')$, then we obtain a set of size $O(1/\varepsilon^{d-k})$ containing a vector u with

$\theta_u \leq \phi$ in the following manner. The intersection of an arbitrary $(d-k+1)$-flat F in general position and the optimal flat $S_{k,d}(P)$ is a line ℓ. Using a $(d-k)$-dimensional grid of directions in F, we create a set of $O(1/\varepsilon^{d-k})$ vectors that contain a vector u within angle at most $\varepsilon'/2$ of ℓ, and consequently has $\theta_u \leq \phi$. In the following paragraph, we focus on the more interesting case when $4c_{k,d}(P) < h_{v'}(P')$.

By Lemma 2.2, we have that $\|v\|$ is a constant factor approximation of $h_{v'}(P')$. Using Lemma 2.1, we can recursively solve the $(d-1)$-dimensional problem with point set $P_{|v}$ in order to obtain a constant factor approximation to $c_{k,d}(P)$. Putting both approximations together and using that $\theta_v \leq 4c_{k,d}(P)/h_{v'}(P')$, we obtain a constant factor upper bound to θ_v. Since $4c_{k,d}(P) < h_{v'}(P')$, we have that $h_{v'}(P')$ and consequently $\|v\|$ are constant factor approximations to $h_{u'}(P')$. Therefore, we can also calculate a constant factor approximation to $c_{k,d}(P)/h_{u'}(P')$. The set of vectors is defined by a grid of directions in $(d-k+1)$-dimensional space as before, but using the fact that the angle between v and u is upper bounded by the approximation of θ_v. We create a set of $O(1/\varepsilon^{d-k})$ vectors that contain a vector u within angle at most ϕ of ℓ, and consequently has $\theta_u \leq \phi$. The size of the set follows from the fact that we only need to consider vectors within angle at most θ_v of ℓ.

Let $p \in P'$ be an inlier. By Lemma 2.2, the set $V = \{p - q : q \in P\}$ of size $O(n)$ contains a vector satisfying the condition of Lemma 2.2. Therefore, we can obtain a set U of size $O(n/\varepsilon^{d-k})$ that contains a vector u with $\theta_u \leq \phi$. For each vector $u \in U$, we project the points onto a hyperplane perpendicular to u and recursively solve the lower dimensional problem. The running time $t_{k,d}$ of the flat fitting algorithm, given an inlier is

$$t_{k,d} = \begin{cases} O(n/\varepsilon^{d-k})t_{k-1,d-1} & \text{if } k > 0 \\ t_{0,d} = O(n/\varepsilon^{d}) & \text{if } k = 0. \end{cases}$$

Consequently,

$$t_{k,d} = O\left(\frac{n^k}{\varepsilon^{k(d-k)}}\right) t_{0,d-k} = O\left(\frac{n^{k+1}}{\varepsilon^{(k+1)(d-k)}}\right).$$

Considering that a set of $O(n/m)$ random points contain an inlier with constant probability and a set of all $O(n)$ points is guaranteed to contain an inlier, we conclude with the following theorem.

Theorem 2.3. *There is a Monte Carlo algorithm to compute, with constant probability, a $(1+\varepsilon)$-approximation of the k-flat that best fits m out of n points in d-dimensional space in time $O_\varepsilon(n^{k+2}/m)$ and, showing ε-dependencies,*

$$O\left(\frac{n^{k+1}}{m\varepsilon^{k(d-k)}}\right) t_{0,d-k} = O\left(\frac{n^{k+2}}{m\varepsilon^{(k+1)(d-k)}}\right).$$

There is also also a deterministic algorithm with running time $O_\varepsilon(n^{k+2})$ and, showing ε-dependencies,

$$O\left(\frac{n^{k+1}}{\varepsilon^{k(d-k)}}\right) t_{0,d-k} = O\left(\frac{n^{k+2}}{\varepsilon^{(k+1)(d-k)}}\right).$$

3 Outer-dense Inliers

In this section, we show that for many data sets a random pair of inliers define a vector v satisfying the properties of Lemma 2.2 with constant probability. Consequently, we obtain a Monte Carlo algorithm with running time $O_\varepsilon(n^{k+2}/m^{k+1})$, which is linear for $m = \Omega(n)$. The idea is that, in lemma 2.2, all we need is a pair of inliers that is sufficiently far from each other. If there are many inliers clustered near the center of the set, and few inliers near the extremes, randomly finding such pair is hard. On the other hand, if the points are uniformly distributed or more concentrated near the extremes, then the problem gets much easier.

We say that a halfspace H with normal vector v' is *deep* if $h_{v'}(P' \setminus H) \leq 3h_{v'}(P')/4$. For a constant $\alpha \leq 1/2$, we say that the set P' is *α-outer-dense* if any deep halfspace H has $|P' \cap H| \geq \alpha|P'|$. The set P' is *outer-dense* if there is a constant α such that P' is α-outer-dense (see Figures 3(a) and 3(b) for an intuitive idea). We believe that many practical sets of inliers are outer-dense. For example, point sets uniformly distributed in a convex region or on the boundary of a convex region are outer-dense with high probability. The following lemma is analogous to Lemma 2.2 when the set P' is α-outer-dense.

Lemma 3.1. *If the inliers P' are α-outer-dense, then the vector $v = q - p$ defined by two random elements $p, q \in P'$ has*

$$\theta_v \leq \frac{4c_{k,d}(P)}{h_{v'}(P')} \quad \text{and} \quad \frac{h_{v'}(P')}{2} \leq \|v\| \leq 2c_{k,d}(P) + h_{v'}(P')$$

with probability at least $2\alpha^2$.

Proof. Consider two disjoint deep halfspaces H_1, H_2 with normal vector v' such that v' is parallel to the optimal flat $S_{k,d}(P)$ and each of H_1 and H_2 contain $1/4$ of the directional width in direction v' (see Figure 3(c)). Since P' is outer-dense $|P' \cap H_1|, |P' \cap H_2| \geq \alpha|P'|$. Therefore, the probability that two random elements $p, q \in P'$ are one in H_1 and the other in H_2 is at least $2\alpha^2$. The lemma then follows from the same arguments as in the proof of Lemma 2.2. □

Note that if a set of points is α-outer-dense, then the projection of the set onto a $(d-1)$-dimensional hyperplane is α-outer-dense in dimension $d-1$. Therefore, we obtain a Monte Carlo algorithm by sampling $n/m\alpha^2$ pairs of points at each step, and then solving the lower dimensional problems. The following theorem presents this upper bound.

Theorem 3.2. *When the set of inliers is outer-dense, there is a Monte Carlo algorithm to compute, with constant probability, a $(1+\varepsilon)$-approximation of the k-flat that best fits m out of n points in d-dimensional space in time $O_\varepsilon(n^{k+2}/m^{k+1})$ and, showing ε-dependencies,*

$$O\left(\frac{n^{k+1}}{m^{k+1}\varepsilon^{k(d-k)}}\right) t_{0,d-k} = O\left(\frac{n^{k+2}}{m^{k+1}\varepsilon^{(k+1)(d-k)}}\right).$$

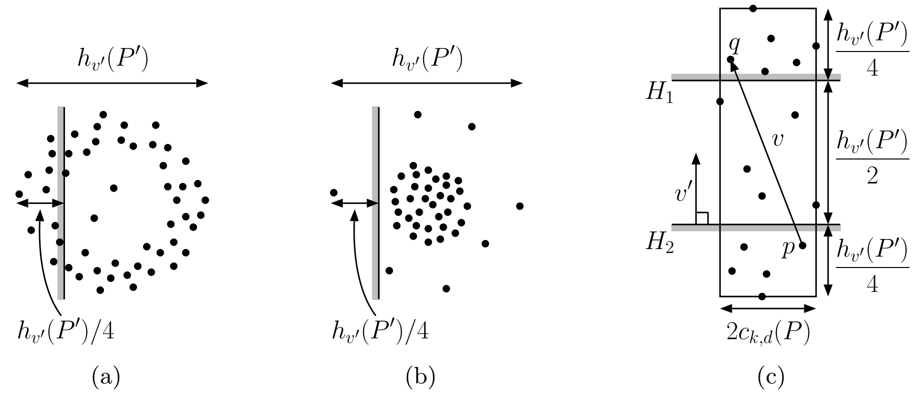

Figure 3: (a) Intuitive idea of an outer-dense set. (b) Intuitive idea of a set that is not outer-dense. (c) Figure for the proof of Lemma 3.1.

4 Conclusions and Open Problems

We present an approximate algorithm to solve several natural problems such as the smallest m-enclosing ball ($k = 0$), infinite cylinder ($k = 1$), and slab ($k = d-1$). Except for the two extreme cases, we present the first solution for the flat fitting problem for constant d. When m is a constant fraction of n, the gap between the lower bound and our Monte Carlo upper bound of $O_\varepsilon(n^{k+1})$ is only $\Theta(n)$. While our upper bound is close to the lower bound and does not depend on d, the algorithm is not efficient for large k. Fortunately, if the set of inliers is outer-dense, then the problem becomes exceedingly easier, with a linear time solution.

A related decision problem which may be useful to reduce the running time of our Monte Carlo algorithm for general point sets by an $O_\varepsilon(n)$ factor is the following. Given a set P of n points in d-dimensional space and an integer $m \leq n$, determine if there is a line ℓ that passes through the origin and is within distance 1 to m points of P. The algorithm may give an approximate answer in the sense that points within distance between 1 and $1 + \varepsilon$ may be counted either way. Except for the planar case, we know of no near linear solution, nor do we know if the problem is 3SUM-hard.

References

[1] P. K. Agarwal, S. Har-Peled, and H. Yu. Robust shape fitting via peeling and grating coresets. *Discrete Comput. Geom.* 39(1):38–58, 2008, doi:10.1007/s00454-007-9013-2, `http://valis.cs.uiuc.edu/~sariel/papers/05/outliers_inc/outliers_inc.pdf`.

[2] S. Arya, G. D. da Fonseca, and D. M. Mount. A unified approach to approximate proximity searching. *Proc. 18th Annu. Euro. Sympos. Algo. (ESA)*, p. to appear, 2010, `http://www.uniriotec.br/~fonseca/proximity.pdf`.

[3] T. Bernholt. Computing the least median of squares estimator in time $O(n^d)$. *Proc. Inter. Conf. Comput. Sci. Appl. (ICCSA)*, pp. 697–706,

2005, doi:10.1007/11424758_72, ls2-www.cs.uni-dortmund.de/.../Computing_the_Least_Median_of_Squares_Estimator.pdf.

[4] T. M. Chan. Geometric applications of a randomized optimization technique. *Discrete Comput. Geom.* 22(4):547–567, 1999, doi:10.1007/PL00009478, http://www.cs.uwaterloo.ca/~tmchan/rand.ps.gz.

[5] T. M. Chan. Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus. *Internat. J. Comput. Geom. Appl.* 12(1/2):67–85, 2002, doi:10.1142/S0218195902000748, http://www.cs.uwaterloo.ca/~tmchan/apx.ps.gz.

[6] T. M. Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom.* 35(1):20–35, 2006, doi:10.1016/j.comgeo.2005.10.002, http://www.cs.uwaterloo.ca/~tmchan/core.ps.

[7] J. Erickson, S. Har-Peled, and D. M. Mount. On the least median square problem. *Discrete Comput. Geom.* 36(4):593–607, 2006, doi:10.1007/s00454-006-1267-6, http://www.cs.umd.edu/~mount/Papers/dcg06-lms.pdf.

[8] C. M. H. de Figueiredo and G. D. da Fonseca. Enclosing weighted points with an almost-unit ball. *Inform. Process. Lett.* 109:1216–1221, 2009, doi:10.1016/j.ipl.2009.09.001, http://www.uniriotec.br/~fonseca/enclosing-IPL.pdf.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24(6):381–395, 1981, doi:10.1145/358669.358692.

[10] G. D. da Fonseca and D. M. Mount. Approximate range searching: The absolute model. *Comput. Geom.* 43(4):434–444, 2010, doi:10.1016/j.comgeo.2008.09.009, http://www.uniriotec.br/~fonseca/ARS-CGTA.pdf.

[11] S. Funke, T. Malamatos, and R. Ray. Finding planar regions in a terrain: in practice and with a guarantee. *Internat. J. Comput. Geom. Appl.* 15(4):379–401, 2005, doi:10.1142/S0218195905001750, www.mpi-inf.mpg.de/~funke/Papers/SoCG04/IJCGA-SoCG04.pdf.

[12] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom.* 5(3):165–185, 1995, doi:10.1016/0925-7721(95)00022-2.

[13] S. Har-Peled and S. Mazumdar. Fast algorithms for computing the smallest k-enclosing circle. *Algorithmica* 41(3):147–157, 2005, doi:10.1007/s00453-004-1123-0, http://valis.cs.uiuc.edu/~sariel/papers/03/min_disk/min_disk.pdf.

[14] S. Har-Peled and K. R. Varadarajan. Projective clustering in high dimensions using core-sets. *Proc. 18th Annu. ACM Sympos. Comput. Geom. (SoCG)*, pp. 312–318, 2002, doi:10.1145/513400.513440, http://valis.cs.uiuc.edu/~sariel/papers/01/kflat/kflat.pdf.

[15] S. Har-Peled and K. R. Varadarajan. High-dimensional shape fitting in linear time. *Discrete Comput. Geom.* 32(2):269–288, 2004, doi:10.1007/s00454-004-1118-2, `http://valis.cs.uiuc.edu/~sariel/papers/02/pcluster/pcluster.pdf`.

[16] S. Maneewongvatana and D. M. Mount. On the efficiency of nearest neighbor searching with data clustered in lower dimensions. *Proc. Inter. Conf. Comput. Sci. (ICCS)*, pp. 842–851, 2001.

[17] D. M. Mount, N. S. Netanyahu, C. D. Piatko, and R. Silverman. A practical approximation algorithm for the LTS estimator. in preparation.

[18] D. M. Mount, N. S. Netanyahu, K. Romanik, R. Silverman, and A. Y. Wu. A practical approximation algorithm for the LMS line estimator. *Comput. Stat. Data Anal.* 51(5):2461–2486, 2007, doi:10.1016/j.csda.2006.08.033, `http://www.cs.umd.edu/~mount/Papers/csda07-alms.pdf`.

[19] E. Schömer, J. Sellen, M. Teichmann, and C. Yap. Smallest enclosing cylinders. *Algorithmica* 27(2):170–186, 2000, `www.staff.uni-mainz.de/schoemer/publications/SEC.pdf`.