Fast Algorithms for Computing the Smallest k-Enclosing Disc

Sariel Har-Peled^{*} Soham Mazumdar[†]

November 10, 2003

Abstract

We consider the problem of finding, for a given n point set P in the plane and an integer $k \leq n$, the smallest circle enclosing at least k points of P. We present a randomized algorithm that computes in O(nk) expected time such a circle, improving over previously known algorithms.

Since this problem is believed to require $\Omega(nk)$ time, we present a linear time δ -approximation algorithm that outputs a circle that contains at least k points of P, and of radius less than $(1 + \delta)r_{opt}(P, k)$, where $r_{opt}(P, k)$ is the radius of the minimal disk containing at least k points of P. The expected running time of this approximation algorithm is $O(n + n \cdot \min(\frac{1}{k\delta^3}\log^2 \frac{1}{\delta}, k))$.

1 Introduction

Shape fitting, a fundamental problem in computational geometry, computer vision, machine learning, data mining, and many other areas, is concerned with finding the best shape which "fits" a given input. This problem has attracted a lot of research both for the exact and approximation versions, see [BE97, HV01] and references therein.

Furthermore, solving such problems in the real world is quite challenging, as noise in the input is omnipresent and one has to assume that some of the input points are noise, and as such should be ignored. See [Cha02, EE94, Mat95b] for some recent relevant results. Unfortunately, under such noisy conditions, the shape fitting problem becomes notably harder.

An important class of shape fitting problems involve finding an optimal k point subsets from a set of n points based on some optimizing criteria. The optimizing criteria could be the smallest convex hull volume, the smallest enclosing ball, the smallest enclosing box, the smallest diameter amongst others [EE94, AST94].

An interesting problem of this class is that of computing the smallest disc which contains k points from a given set of n points in a plane. The initial approaches to solving this problem

^{*}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. Work on this paper was partially supported by a NSF CAREER award CCR-0132901.

[†]Department of Computer Science; University of Illinois; 1304 West Springfield Ave.; Urbana, IL 61801; USA; smazumda@uiuc.edu.

involved first constructing the order-k Voronoi diagram, followed by a search in all or some of the Voronoi cells. The best known algorithm to compute the order-k Voronoi diagram has time complexity $O(nk + n \log^3 n)$, see [AdBMS98]. Eppstein and Erickson [EE94] observed that instead of Voronoi cells, one can work with some O(k) nearest neighbors to each point. The resulting algorithm had a running time of $O(nk \log n + nk \log^2 k)$ and space complexity $O(nk + k^2 \log k)$. Using the technique of parametric search, Efrat *et al.* [ESZ94] solved the problem in time $O(nk \log^2 n)$ and space O(nk). Finally, Matoušek [Mat95a] by using a suitable randomized search gave a very simple algorithm which used O(nk) space and had $O(n \log n + nk)$ expected running time.

We revisit this classical problem, and present an algorithm with O(nk) expected running time, that uses $O(n + k^2)$ space. The main reason why this result is interesting is because it beats the lower bound of $\Omega(n \log n)$ on the running time for small k, which follows from element uniqueness in the comparison model. We achieve this by using randomization and the floor function (interestingly enough, this is also the computation model used by Matoušek [Mat95a]). Despite this somewhat small gain, removing the extra $\log n$ factor from the running time was a non-trivial undertaking, requiring some new ideas.

The key ingredient in our algorithm is a new *linear* time 2-approximation algorithm, described in Section 3. This significantly improves over the previous best result of Matoušek [Mat95a] that runs in $O(n \log n)$ time. Using our algorithm and the later half of the algorithm of Matoušek (with some minor modifications), we get the new improved exact algorithm. Finally, in Section 5, we observe that from the 2-approximation algorithm one can get a δ -approximation algorithm with running time linear in n and polynomial dependency on $1/\delta$.

2 Preliminaries

For a point p = (x, y) in \mathbb{R}^2 , define $G_r(p)$ to be the point $(\lfloor x/r \rfloor r, \lfloor y/r \rfloor r)$. We call r the width of the grid G_r . Observe that G_r partitions the whole space into square regions, which we call grid *cells*. Formally, for any $i, j \in \mathbb{Z}$, the intersection of the half-planes $x \geq ri$, $x < r(i+1), y \geq rj$ and y < r(j+1) is said to be a grid cell. Further, we call a block of 3×3 contiguous grid cells as a grid cluster.

For a point set P, and parameter r, the partition of P into subsets by the grid G_r , is denoted by $G_r(P)$. More formally, two points $p, q \in P$ belong to the same set in the partition $G_r(P)$, if both points are being mapped to the same grid point or equivalently belong to the same grid cell.

Let $gd_P(r)$ denote the maximum number of points of P mapped to a single point by the mapping G_r . Define depth(P, r) to be the maximum number of points of P that a disc of radius r can contain. Let $D_{opt}(P, k)$ be a disc of minimal radius which contains k points of P. Let $r_{opt}(P, k)$ denote the radius of $D_{opt}(P, k)$.

The above notation is originally from Matoušek [Mat95a]. Using simple packing arguments one can prove the following results [Mat95a]:

 $\begin{array}{ll} \textbf{Lemma 2.1} & (i) \hspace{0.1cm} \texttt{depth}(P,Ar) \leq (A+1)^2 \hspace{0.1cm} \texttt{depth}(P,r), \\ (ii) \hspace{0.1cm} \texttt{gd}_P(r) \leq \hspace{0.1cm} \texttt{depth}(P,r) = O(\hspace{0.1cm}\texttt{gd}_P(r)). \end{array}$

(iii) If $r_{opt}(P,k) \leq r \leq 2r_{opt}(P,k)$ then $\operatorname{gd}_P(r) \leq 5k$.

Lemma 2.2 Any disk of radius r can be covered by some grid cluster in G_r .

Definition 2.3 (Gradation) Given a set P of n points, a sampling sequence (S_1, \ldots, S_m) of P is a sequence of sets, such that (i) $S_1 = P$, (ii) S_i is formed by picking each point of S_{i-1} into S_i with probability half, and (iii) $|S_m| \leq n/\log n$, and $|S_{m-1}| > n/\log n$. The sequence $(S_m, S_{m-1}, \ldots, S_1)$ is a gradation of P.

Lemma 2.4 Given P, a sampling sequence can be computed in expected linear time.

Proof: Observe that the sampling time is $O(\sum_{i=1}^{m} |S_i|)$, where m is the length of the sequence. Note, that

$$\mathbf{E}[|S_i|] = \mathbf{E}\left[\mathbf{E}\left[|S_i| \mid |S_{i-1}|\right]\right] = \mathbf{E}\left[\frac{|S_{i-1}|}{2}\right] = \frac{n}{2^{i-1}}$$

Thus, $O(\mathbf{E}[\sum_{i=1}^{m} |S_i|]) = O(n).$

3 A Slow 2-Approximation Algorithm

In this section, we develop a slow approximation algorithm that would be used in our faster algorithm, presented in Section 4. Surprisingly, any 2-approximation algorithm would do for our purposes, as long as its running time is $O(n \cdot (n/k)^c)$, where c is a constant.

3.1 A Deterministic Algorithm

For $\varepsilon = k/n$, we compute an optimal sized ε -net for the set system (P, \mathcal{R}) , where \mathcal{R} is the set of discs in the plane and the VC dimension of this set system is d = 4. We compute, in $O(n(n/k)^{2d} \log(n/k)^d) = O(n(n/k)^9)$ time, an ε -net S for (P, \mathcal{R}) using a deterministic construction [Cha00, Section 4.3], where $|S| = O((n/k) \log(n/k))$.

From the definition of ε -nets, it follows that $\exists z \in S$, such that $z \in D_{opt}(P, k)$. Observe, that if s' is the (k-1)th closest point to z from P then then $dist(z, s') \leq 2r_{opt}(P, k)$. This holds as (k-1) points in $P \setminus \{z\}$ are in $D_{opt}(P, k)$ and hence they are at a distance $\leq 2r_{opt}(P, k)$ from z, by the triangle inequality. For each point in S, we compute its distance from the (k-1)th closest point to it in P. Let r be the smallest of these |S| distances. From the above argument, it follows that $r_{opt}(P, k) \leq r \leq 2r_{opt}(P, k)$. The selection of the (k-1)th closest point can be done deterministically in linear time, by using deterministic median selection [CLRS01]. Overall, this stage takes O(n|S|) time.

Lemma 3.1 Given a set P of n points in the plane, and parameter k, one can compute in $O(n(n/k)^9)$ deterministic time, a disc D that contains k points of P, and $\operatorname{radius}(D) \leq 2r_{opt}(P,k)$.

Corollary 3.2 Given a set of P of n points and a parameter $k = \Omega(n)$, one can compute in expected linear time, a disc D that contains k points of P and radius $(D) \leq 2r_{opt}(P,k)$.

3.2 A Randomized Algorithm

Let R be a random sample from P, generated by choosing every point of P with probability 1/k. Next, compute for every $p \in R$, the smallest disc centered at p containing k points of P. Overall, this takes O(n(n/k)) expected time, as $\mathbf{E}[|R|] = n/k$ and computing the smallest disc for every point takes linear time using median selection. Let r be the minimum radius computed.

Let X be the k points of P covered by $D_{opt}(P, k)$. We have

$$\mathbf{Pr}[X \cap R \neq \emptyset] = 1 - \left(1 - \frac{1}{k}\right)^k \ge 1 - \frac{1}{e}.$$

If R contains a point covered by $D_{opt}(P, k)$, then r is a 2-approximation to $r_{opt}(P, k)$. We need to verify that this indeed occurred. To do so, compute the grid G_r , and snap the points of P into it. If any cluster of G_r contains more than, say, 36k points, then r is clearly too large, and the algorithm failed. As such, we run it again.

Otherwise, we run on each cluster of G_r that contains at least k points, the algorithm of Corollary 3.2. Let D be the smallest disk computed that contains k points of P. Clearly, D is a 2-approximation to $D_{opt}(P, k)$. The overall running time of this stage is O((n/k)k) = O(n).

Since the algorithm has constant probability to succeed, and it repeat till success, it follows that the expected running time of our algorithm is O(n(n/k)).

Lemma 3.3 Given a set P of n points in the plane, and parameter k, one can compute in expected O(n(n/k)) time, a disc D that contains k points of P, and $radius(D) \leq 2r_{opt}(P,k)$.

We refer to the algorithm of Lemma 3.3 as APPROXHEAVY(P, k).

Remark 3.4 For a point set P of n points, the radius r returned by the algorithm of Lemma 3.3 is the distance between a pair of points of P. As such, a grid G_r computed using this distance is one of $O(n^2)$ possible grids.

3.3 A Faster Algorithm

In an earlier version of this paper [HM03], we had a considerably more involved algorithm that performed the 2-approximation in $O(n \log(n/k))$ time. However, for our purposes, the slow algorithms of Lemma 3.1 and Lemma 3.3 are both sufficient. The algorithm was based on the idea of using Corollary 3.2 to find a grid size, such that each cell of the grid contains only a fraction of the *n* points (i.e. running APPROXHEAVY(P, |P|/c) where *c* is a constant). Next, the algorithm recursed on cells in the grid that contained considerably more than *k* points. In the end of this process, the plane was partition into squares such that the smallest of them that contained at least *k* points, provided a good approximation to the value of $r_{opt}(P, k)$. See [HM03] for details.

```
 \begin{array}{l} \operatorname{GROW}(P_i,r_{i-1},k) \\ \operatorname{Output:} r_i \\ \textbf{begin} \\ \operatorname{G}_{i-1} \leftarrow \operatorname{G}_{r_{i-1}}(P_i) \\ \textbf{for every grid cluster } c \in \operatorname{G}_{i-1} \text{ with } |c \cap P_i| \geq k \ \textbf{do} \\ P_c \leftarrow c \cap P_i \\ r_c \leftarrow \operatorname{APPROXHEAVY}(P_c,k) \\ // \ \operatorname{APPROXHEAVY} \text{ is the algorithm of Lemma 3.3} \\ \operatorname{We have} r_{opt}(P_c,k) \leq r_c \leq 2r_{opt}(P_c,k), \\ \textbf{return minimum } r_c \ \text{computed.} \\ \textbf{end} \end{array}
```

Figure 1: Algorithm for the ith round

4 A 2-Approximation in Linear Time

4.1 Description

As done in the previous section, we construct a grid which partitions the points into small (O(k) sized) groups. The key idea behind speeding up the grid computation is to construct the appropriate grid over several rounds. Specifically, we start with a small set of points as seed and construct a suitable grid for this subset. Next, we incrementally insert the remaining points, while adjusting the grid width appropriately at each step.

Let $\mathcal{P} = (P_1, \ldots, P_m)$ be a gradation of P (see Definition 2.3), where $|P_1| \geq \max(k, n/\log n)$ (i.e. if $k \geq n/\log n$ we start from the first set in \mathcal{P} that has more than k elements). The sequence \mathcal{P} can be computed in expected linear time as shown in Lemma 2.4. Now using the algorithm of [Mat95a], we obtain a length r_1 such that $r_{opt}(P_1, k) \leq r_1 \leq 2r_{opt}(P_1, k)$ and $gd_{r_1}(P_1) \leq 5k$. The set P_1 is the seed subset mentioned earlier. Observe that it takes $O(|P_1|\log|P_1|) = O(n)$ time to perform this step. The remaining algorithm works in m rounds, where m is the length of the sequence \mathcal{P} . At the end of the *i*th round, we have a distance r_i such that $gd_{r_i}(P_i) \leq 5k$, and there exists a grid cluster in G_{r_i} containing more than k points of P_i and $r_{opt}(P_i) \leq r_i$.

At the *i*th round, we first construct a grid G_{i-1} for points in P_i using r_{i-1} as grid width. We know that there is no grid cell containing more than 5k points of P_{i-1} . As such, intuitively, we expect every cell of G_{i-1} to contain at most 10k points of P_i . (This is of course too good to be true, but something slightly weaker does hold.) Thus allowing us to use the slow algorithm of Lemma 3.3 on those grid clusters. Note that, for $k = \Omega(n)$, the algorithm of Lemma 3.3 runs in expected linear time, and thus the overall running time is linear.

The algorithm used in the *i*th round is more concisely stated in Figure 1. At the end of the *m* rounds we have r_m , which is a 2-approximation to the radius of the optimal *k* enclosing disc of $P_m = P$. The overall algorithm is summarized in Figure 2.

LINEARAPPROX(P,k) Output: r - a 2-approximation to $r_{opt}(P, k)$ begin Compute a gradation $\{P_1, \ldots, P_m\}$ of P as in Lemma 2.4 $r_1 \leftarrow \text{APPROXHEAVY}(P_1, k)$ // APPROXHEAVY is the algorithm of Lemma 3.3 for $j \leftarrow 2$ to m do $r_j \leftarrow \text{GROW}(P_j, r_{j-1}, k)$ for every grid cluster $c \in G_{r_m}$ with $|c \cap P| \ge k$ do $r_c \leftarrow \text{APPROXHEAVY}(c \cap P, k)$ return minimum r_c computed over all clusters end

Figure 2: 2-Approximation Algorithm

4.2 Analysis

Lemma 4.1 For i = 1, ..., m, we have $r_{opt}(P_i, k) \leq r_i \leq 2r_{opt}(P_i, k)$, and the heaviest cell in $G_{r_i}(P_i)$ contains at most 5k points of P_i .

Proof: Consider the optimal disk D_i that realizes $r_{opt}(P_i, k)$. Observe that there is a cluster c of $G_{r_{i-1}}$ that contains D_i , as $r_{i-1} \ge r_i$. Thus, when GROW handles the cluster c, we have $D_i \cap P_i \subseteq c$. The first part of the lemma then follows from the correctness of the algorithm in Lemma 3.3.

As for the second part, observe that any grid cell of width r_i can be covered with 5 disks of radius $r_i/2$. It follows that the grid cell of $r_{opt}(P_i, k)$ contains at most 5k points.

Definition 4.2 For a point set P, and a parameters k and r, the excess of $G_r(P)$ is

$$\mathcal{E}(P, k, \mathbf{G}_r) = \sum_{c \in \text{Cells}(\mathbf{G}_r)} \left\lfloor \frac{|c \cap P|}{50k} \right\rfloor,$$

where $Cells(G_r)$ is the set of cells of the grid G_r .

Remark 4.3 The quantity $100k \cdot \mathcal{E}(P, k, G_r)$ is an upper bound on the number of points of P in an heavy cell of $G_r(P)$, where a cell of $G_r(P)$ is *heavy* if it contains more than 50k points.

Lemma 4.4 For any positive real t, the probability that $G_{r_{i-1}}(P_i)$ has excess $\mathcal{E}(P_i, k, G_{r_{i-1}}) \geq t + 2 \lceil \log(n) \rceil$, is at most 2^{-t} .

Proof: Let \mathfrak{G} be the set of $O(n^2)$ possible grids that might be considered by the algorithm (see Remark 3.4), and fix a grid $\mathsf{G} \in \mathfrak{G}$ with excess $M = \mathcal{E}(P_i, k, \mathsf{G}_{r_{i-1}})$.

Let $U = \left\{ P_i \cap c \mid c \in \mathsf{G}, |P_i \cap c| > 50k \right\}$ be all the heavy cells in $\mathsf{G}(P_i)$. Furthermore, let $V = \bigcup_{X \in U} \psi(X, 50k)$, where $\psi(X, \nu)$ denotes an arbitrary partition of the set X into disjoint subsets such at each one of them contains ν points, except maybe the last subset that might contain between ν and $2\nu - 1$ points.

It is clear that $|V| = \mathcal{E}(P_i, k, \mathsf{G})$. From the Chernoff inequality, for any $S \in V$,

$$\mathbf{Pr}[|S \cap P_{i-1}| \le 5k] < exp\left(-\frac{25k(1-1/5)^2}{2}\right) < \frac{1}{2}$$

Furthermore, $G = G_{r_{i-1}}$ only if each cell of $G(P_{i-1})$ contains at most 5k points. Thus we have

$$\begin{aligned} \mathbf{Pr}\big[(\mathbf{G}_{r_{i-1}} = \mathsf{G}) \cap (\mathcal{E}(P_i, k, \mathsf{G}) = M)\big] &\leq \mathbf{Pr}\Big[\mathbf{G}_{r_{i-1}} = \mathsf{G} \mid \mathcal{E}(P_i, k, \mathsf{G}) = M\Big] \\ &\leq \prod_{S \in V} \mathbf{Pr}[|S \cap P_{i-1}| \leq k] \\ &\leq \frac{1}{2^{|V|}} = \frac{1}{2^M}. \end{aligned}$$

Since there are $\binom{n}{2}$ different grids in \mathfrak{G} , we have

$$\mathbf{Pr}\big[\mathcal{E}(P_i,k,\mathbf{G}_{r_{i-1}})=M\big]=\mathbf{Pr}\left[\bigcup_{\mathsf{G}\in\mathfrak{G}}(\mathsf{G}=\mathbf{G}_{r_{i-1}})\cap\mathcal{E}(P_i,k,\mathsf{G})=M\right]\leq \binom{n}{2}\frac{1}{2^M}\leq\frac{1}{2^t}.$$

Thus, by Lemma 4.4, for $k < 4 \log n$, we have that if the excess is smaller than $\gamma = 2 \lceil \log n \rceil + 1$, then we have at most γ heavy cells, and every heavy cell contains at most $O(\gamma k)$ points. It follows, that the total running time of Lemma 3.3 on those heavy cells, is at most $O(\gamma \cdot \gamma k(\gamma k/k)) = O(k \log^3 n)$. Thus, we have that the expected running time of the *i*th step is at most

$$O\left(|P_i| + \sum_{c \in G_{r_{i-1}}} |c \cap P_i| \frac{|c \cap P_i|}{k}\right) = O\left(|P_i| + k \log^3 n + \sum_{t=1+2\lceil \log n \rceil}^{\infty} t \cdot \frac{(tk)^2}{k} \cdot \frac{1}{2^t}\right) = O\left(|P_i| + k \log^3 n\right) = O(|P_i|),$$

since if the excess is t, then there are t cells with more than $\Omega(k)$ points, and each such cell contains at most O(tk) points.

We next handle the case where $k \ge 4 \log n$.

Lemma 4.5 The probability that $G_{r_{i-1}}(P_i)$ has excess larger than t, is at most 2^{-t} , for $k \ge 4 \log n$.

Proof: We use the same technique as in Lemma 4.4. By the Chernoff inequality, the probability that any 50k size subset of P_i would contain at most 5k points of P_{i-1} , is less than

$$\leq \exp\left(-25k \cdot \frac{16}{25} \cdot \frac{1}{2}\right) \leq \exp(-5k) \leq \frac{1}{n^4}.$$

In particular, arguing as in Lemma 4.4, it follows that the probability that $\mathcal{E}(P_i, k, r_{i-1})$ exceeds t, is smaller than $\binom{n}{2}/n^{4t} \leq 2^{-t}$.

Thus, if $k \ge 4 \log n$, the expected running time of the *i*th step is at most

$$O\left(\sum_{c \in G_{r_{i-1}}} |c \cap P_i| \log \frac{|c \cap P_i|}{k}\right) = O\left(|P_i| + \sum_{t=1}^{\infty} t \cdot \frac{(tk)^2}{k} \frac{1}{2^t}\right) = O(|P_i| + k) = O(|P_i|),$$

by Lemma 4.5.

Thus, the total expected running time is $O(\sum_i |P_i|) = O(n)$, by the analysis of Lemma 2.4. To compute a 2-approximation, consider the grid $G_{r_m}(P)$. Each grid cell contains at most 5k points, and hence each grid cluster contains at most 45k points. Also the smallest k enclosing disc is contained in a certain grid cluster. In each cluster that contain more than k points, we use the algorithm of Corollary 3.2 and then finally output the minimum over all the clusters. The overall running time is O((n/k)k) = O(n) for this step, since each point belongs to at most 9 clusters.

Theorem 4.6 Given a set P of n points in the plane, and a parameter k, one can compute, in expected linear time, a radius r, such that $r_{opt}(P, k) \leq r \leq 2r_{opt}(P, k)$.

Once we compute r such that $r_{opt}(P, k) \leq r \leq 2r_{opt}(P, k)$, using the algorithm of Theorem 4.6, we apply the exact algorithm of Matoušek [Mat95a] to each cluster of the grid $G_r(P)$ which contains more than k points.

Matoušek's algorithm has running time of $O(n \log n + nk)$ and space complexity O(nk). Since r is a 2 approximation to $r_{opt}(P, k)$, each cluster has O(k) points. Thus the running time of the exact algorithm in each cluster is $O(k^2)$ and requires $O(k^2)$ space. The number of clusters which contain more than k points is O(n/k). Hence the overall running time of our algorithm is O(nk), and the space used is $O(n + k^2)$.

Theorem 4.7 Given a set P of n points in the plane and a parameter k, one can compute, in expected O(nk) time, using $O(n + k^2)$ space, the radius $r_{opt}(P,k)$, and a disk $D_{opt}(P,k)$ that covers k points of P.

5 From constant approximation to $(1+\delta)$ -approximation

Suppose r is a 2-approximation to $r_{opt}(P, k)$. Now if we construct $G_r(P)$ each grid cell contains less than 5k points of P (each grid cell can be covered fully by 5 circles of radius $r_{opt}(P, k)$). Furthermore, the smallest k-enclosing circle is covered by a certain grid cluster. We compute a $(1 + \delta)$ -approximation to the radius of the minimal k enclosing circle in each grid cluster and output the smallest amongst them. The technique to compute $(1 + \delta)$ -approximation when all the points belong to a particular grid cluster is as follows.

Let P_c be the set of points in a particular grid cluster with $k < |P_c| = O(k)$. Let R be a bounding square of the points of P_c . We partition R into a uniform grid G of size $10r\delta$. Next, snap every point of P_c into the closest grid point of G, and let P'_c denote the resulting point set. Clearly, $|P'_c| = O(1/\delta^2)$. Assume that we guess the radius $r_{opt}(P_c, k)$ up to a factor of $1 + \delta$ (there are only $O(\log_{1+\delta} 2) = O(1/\delta)$ possible guesses), and let r' be

the current guess. We need to compute for each point p of P'_c , how many points of P'_c are contained in D(p, r'). This can be done in $O((1/\delta) \log(1/\delta))$ time per point, by constructing a quadtree over the points of P'_c . Thus, computing a $\delta/4$ -approximation to the $r_{opt}(P'_c, k)$ takes $O((1/\delta^3) \log^2(1/\delta))$ time.

We repeat the above algorithm for all the clusters that have more than k points inside them. Clearly, the smallest disk computed is the required approximation. The running time is $O(n + n/(k\delta^3) \log^2(1/\delta))$. Putting this together with the algorithm of Theorem 4.6, we have:

Theorem 5.1 Given a set P of n points in the plane, and parameters k and $\delta > 0$, one can compute, in expected

$$O\left(n+n\cdot\min\left(\frac{1}{k\delta^3}\log^2\frac{1}{\delta},k\right)\right)$$

time, a radius r, such that $r_{opt}(P,k) \leq r \leq (1+\delta)r_{opt}(P,k)$.

6 Conclusions

We presented a linear time 2-approximation algorithm for the smallest enclosing disk that contains at least k points in the plane. Note that our algorithm can be easily extended to high dimensions. This algorithm improves over previous results, and it can in some sense be interpreted as an extension of Golin *et al.* [GRSS95] closest pair algorithm to the clustering problem (see also the algorithm by Rabin [Rab76] and the survey of Smid on such algorithms [Smi00]).

Getting similar results for other shape fitting problems, like the minimum radius cylinder in three dimensions, remains elusive. Current approaches for approximating it, in the presence of outliers, essentially reduces to the computation of the shortest vertical segment that stabs at least k hyperplanes. See [HW02] for the details. However, the results of Erickson and Seidel [ES95, Eri99] imply that approximating the shortest vertical segment that stabs d + 1 hyperplanes takes $\Omega(n^d)$ time, under a reasonable computation model, thus implying that this approach is probably bound to fail if we are interested in a near linear time algorithm.

It would be interesting to figure out which of the shape fitting problems can be approximated in near linear time, in the presence of outliers, and which ones can not. We leave this as an open problem for further research.

Acknowledgments

The authors thank Alon Efrat and Edgar Ramos for helpful discussions on the problems studied in this paper.

References

- [AdBMS98] P. K. Agarwal, M. de Berg, J. Matoušek, and O. Schwarzkopf. Constructing levels in arrangements and higher order Voronoi diagrams. SIAM J. Comput., 27:654-667, 1998.
- [AST94] P. K. Agarwal, M. Sharir, and S. Toledo. Applications of parametric searching in geometric optimization. J. Algorithms, 17:292–318, 1994.
- [BE97] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. In D. S. Hochbaum, editor, Approximationg algorithms for NP-Hard problems, pages 296-345. PWS Publishing Company, 1997.
- [Cha00] B. Chazelle. The Discrepancy Method. Cambridge University Press, 2000.
- [Cha02] T. M. Chan. Low-dimensional linear programming with violations. In Proc. 43th Annu. IEEE Sympos. Found. Comput. Sci., pages 570–579, 2002.
- [CLRS01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. Introduction to Algorithms. MIT Press / McGraw-Hill, Cambridge, Mass., 2001.
- [EE94] D. Eppstein and J. Erickson. Iterated nearest neighbors and finding minimal polytopes. *Discrete Comput. Geom.*, 11:321–350, 1994.
- [Eri99] J. Erickson. New lower bounds for convex hull problems in odd dimensions. SIAM J. Comput., 28:1198–1214, 1999.
- [ES95] J. Erickson and R. Seidel. Better lower bounds on detecting affine and spherical degeneracies. *Discrete Comput. Geom.*, 13:41–57, 1995.
- [ESZ94] A. Efrat, M. Sharir, and A. Ziv. Computing the smallest k-enclosing circle and related problems. *Comput. Geom. Theory Appl.*, 4:119–136, 1994.
- [GRSS95] M. Golin, R. Raman, C. Schwarz, and M. Smid. Simple randomized algorithms for closest pair problems. *Nordic J. Comput.*, 2:3–27, 1995.
- [HM03] S. Har-Peled and S. Mazumdar. Fast algorithms for computing the smallest k-enclosing disc. In Proc. 11th Annu. European Sympos. Algorithms, volume 2832 of Lect. Notes in Comp. Sci., pages 278–288, 2003.
- [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.
- [HW02] S. Har-Peled and Y. Wang. Shape fitting with outliers. In Proc. 19th Annu. ACM Sympos. Comput. Geom., pages 29–38, 2002.
- [Mat95a] J. Matoušek. On enclosing k points by a circle. Inform. Process. Lett., 53:217–221, 1995.

- [Mat95b] J. Matoušek. On geometric optimization with few violated constraints. Discrete Comput. Geom., 14:365–384, 1995.
- [Rab76] M. O. Rabin. Probabilistic algorithms. In J. F. Traub, editor, Algorithms and Complexity: New Directions and Recent Results, pages 21–39. Academic Press, New York, NY, 1976.
- [Smi00] M. Smid. Closest-point problems in computational geometry. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 877–935. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.