Discrete Comput Geom (2008) 39: 38–58
DOI 10.1007/s00454-007-9013-2

Robust Shape Fitting via Peeling and Grating Coresets

Pankaj K. Agarwal · Sariel Har-Peled · Hai Yu

Published online: 12 September 2007

Abstract Let P be a set of n points in $\mathbb{R}^d$. A subset $\mathcal{S}$ of P is called a (k, ε)-*kernel* if for every direction, the directional width of $\mathcal{S}$ ε-approximates that of P, when k "outliers" can be ignored in that direction. We show that a (k, ε)-kernel of P of size $O(k/\varepsilon^{(d-1)/2})$ can be computed in time $O(n + k^2/\varepsilon^{d-1})$. The new algorithm works by repeatedly "peeling" away $(0, \varepsilon)$-kernels from the point set.

We also present a simple ε-approximation algorithm for fitting various shapes through a set of points with at most k outliers. The algorithm is incremental and works by repeatedly "grating" critical points into a working set, till the working set provides the required approximation. We prove that the size of the working set is independent of n, and thus results in a simple and practical, near-linear ε-approximation algorithm for shape fitting with outliers in low dimensions.

We demonstrate the practicality of our algorithms by showing their empirical performance on various inputs and problems.

Keywords Shape fitting · Coresets · Geometric approximation algorithms

A preliminary version of this paper appeared in *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 182–191. P.A. and H.Y. are supported by NSF under grants CCR-00-86013, EIA-01-31905, CCR-02-04118, and DEB-04-25465, by ARO grants W911NF-04-1-0278 and DAAD19-03-1-0352, and by a grant from the U.S.–Israel Binational Science Foundation. S.H.-P. is supported by a NSF CAREER award CCR-0132901.

P.K. Agarwal (✉) · H. Yu
Department of Computer Science, Duke University, Durham, NC 27708, USA
e-mail: pankaj@cs.duke.edu

H. Yu
e-mail: fishhai@cs.duke.edu

S. Har-Peled
Department of Computer Science, University of Illinois, Urbana, IL 61801, USA
e-mail: sariel@cs.uiuc.edu

1 Introduction

In many areas such as computational geometry, computer graphics, machine learning, and data mining, considerable work has been done on computing descriptors of the extent of a set P of n points in $\mathbb{R}^d$. These descriptors, called *extent measures*, either compute certain statistics of P itself such as diameter and width, or compute some geometric shape enclosing P with respect to a certain optimization criterion, such as computing the smallest radius of a sphere or cylinder, the minimum volume or surface area of a box, and the smallest spherical or cylindrical shell that contain P. Motivated by more recent applications, there has also been work on maintaining extent measures of a set of moving points, e.g., using the kinetic data structure framework [6, 11].

The existing exact algorithms for computing extent measures are generally expensive. For example, the best known algorithm for computing the smallest enclosing cylindrical shell in $\mathbb{R}^3$ requires $O(n^5)$ time [5]. Consequently, attention has shifted to developing faster approximation algorithms; see, e.g., [4, 5, 10, 12]. Agarwal et al. [7] proposed a unified framework for computing numerous extent measures approximately in low dimensions. Their approach is to first extract a small subset from the input, known as a *coreset*, and then return the extent measure of this subset as an approximation to that of the original input. The running time of their algorithm, substantially improving upon many previous results, is typically of the form $O(n+1/\varepsilon^c)$, where n is the input size, c is a constant that may depend on the dimension d, and ε is the approximation error.

Most of the existing work assumes that the input does not contain noisy data. However in the real world, noise may come from different sources during data acquisition, transmission, storage and processing, and is unavoidable in general. Meanwhile, most extent measures are very sensitive to noise; a small number of inaccurate data points (i.e., the so-called *outliers*) may substantially affect extent measures of the entire input. In order to compute more reliable extent measures on the input, it is thus natural to require that the outliers should be excluded from consideration. For example, the smallest enclosing cylinder problem with k outliers is formulated as finding the smallest cylinder that covers all but at most k of the input points.

Following up the work in [7, 19], we consider the problem of finding *robust coresets* for various extent measures that are able to handle outliers. Assuming there are at most k outliers in the input, our goal is to compute a coreset of small size, so that the best solution on the coreset with at most k outliers would provide an ε-approximation to the original input with at most k outliers. We are mainly concerned with the case in which the number k of outliers is small compared to the input size n. Otherwise, random-sampling techniques have been effective in handling outliers [9].

Problem Statement Let P be a set of n points in $\mathbb{R}^d$. For a direction $u \in \mathbb{S}^{d-1}$ and an integer $0 \le k < n$, the *level* of a point $a \in \mathbb{R}^d$ in direction u is the size of the set $\{p \in P \mid \langle u, p\rangle > \langle u, a\rangle\}$, i.e., the number of points in P that lie in the open halfspace $\langle u, x-a\rangle > 0$. This notion of level is the dual of the level of a point in an arrangement of hyperplanes [23]. In this paper, when we refer to a direction $u \in \mathbb{S}^{d-1}$, we always assume for the sake of simplicity that no two points in P lie on the same level in that direction; more careful but similar arguments would work for directions that do not satisfy this assumption.

Let $P^k[u]$ (resp. $P_k[u]$) denote the point of P whose level is k (resp. $n-k-1$) in a direction $u \in \mathbb{S}^{d-1}$. Let $\mathbf{U}_k(u, P) = \langle u, P^k[u] \rangle$ denote the *k-level of P* in direction u. Let $\mathbf{L}_k(u, P) = \langle u, P_k[u] \rangle = -\mathbf{U}_k(-u, P)$. For parameters k and ℓ, the (k, ℓ)-*directional width* of P in direction u, denoted by $\mathcal{E}_{k,\ell}(u, P)$, is defined as

$$\mathcal{E}_{k,\ell}(u, P) = \mathbf{U}_k(u, P) - \mathbf{L}_\ell(u, P).$$

For simplicity, we denote $\mathcal{E}_{k,k}(u, P)$ by $\mathcal{E}_k(u, P)$ and $\mathcal{E}_0(u, P)$ by $\mathcal{E}(u, P)$. Similarly, we denote $\mathbf{U}_0(u, P)$ by $\mathbf{U}(u, P)$ and $\mathbf{L}_0(u, P)$ by $\mathbf{L}(u, P)$ respectively.

Given a set P of n points in $\mathbb{R}^d$, a parameter $\varepsilon > 0$ and an integer $0 \le k < n/2$, a subset $\mathcal{S} \subseteq P$ is called a (k, ε)-*kernel* of P if for every $u \in \mathbb{S}^{d-1}$ and every $0 \le a, b \le k$,

$$(1-\varepsilon) \cdot \mathcal{E}_{a,b}(u, P) \le \mathcal{E}_{a,b}(u, \mathcal{S}) \le \mathcal{E}_{a,b}(u, P).$$

It implies that

$$\mathbf{U}_a(u, \mathcal{S}) \ge \mathbf{U}_a(u, P) - \varepsilon \cdot \mathcal{E}_{a,b}(u, P),$$
$$\mathbf{L}_b(u, \mathcal{S}) \le \mathbf{L}_b(u, P) + \varepsilon \cdot \mathcal{E}_{a,b}(u, P).$$

Note that $(0, \varepsilon)$-kernel is the same as the notion of ε-*kernel* defined by Agarwal et al. [7].

We are interested in computing a (k, ε)-kernel of small size for any given point set $P \subset \mathbb{R}^d$ and parameters k and ε. Once we can compute small (k, ε)-kernels efficiently, we will immediately be able to compute robust coresets for various extent measures, using the standard linearization and duality transforms; see [7] for details.

Related Results The notion of ε-kernels was introduced by Agarwal et al. [7] and efficient algorithms for computing an ε-kernel of a set of n points in $\mathbb{R}^d$ were given in [7, 14, 24]. Yu et al. [24] also gave a simple and fast incremental algorithm for fitting various shapes through a given set of points. See [8] for a review of known results on coresets.

Although there has been much work on approximating a level in an arrangement of hyperplanes using the random-sampling and ε-approximation techniques [15, 21], this line of work has focused on computing a piecewise-linear surface of small complexity that lies within levels $(\pm\varepsilon)k$ for a given integer $k \ge 0$. These algorithms do not extend to approximating a level in the sense defined in this paper.

Perhaps the simplest case in which one can easily show the existence of a small (k, ε)-kernel is when all points of P are collinear in $\mathbb{R}^d$. One simply returns the first and last $k+1$ points along this line as the desired (k, ε)-kernel. In fact, this kernel has exactly the same k-level directional width as P, for all directions. Note that the size of this kernel is $2k+2$, which is independent of the input size. Generalizing this simple example, Har-Peled and Wang [19] showed that for any point set $P \subset \mathbb{R}^d$, one can compute a (k, ε)-kernel of size $O(k/\varepsilon^{d-1})$. Their algorithm is based on a recursive construction, and runs in $O(n + k/\varepsilon^{d-1})$ time. Their result led to approximation algorithms for computing various extent measures with k outliers, whose running times are of the form $O(n + (k/\varepsilon)^c)$.

Our Results In Sect. 2 we prove that there exists a (k, ε)-kernel of size $O(k/\varepsilon^{(d-1)/2})$ for any set P of n points in $\mathbb{R}^d$. This result matches the lower bound $\Omega(k/\varepsilon^{(d-1)/2})$. Our construction is relatively simple and intuitive: it works by repeatly *peeling* away $(\varepsilon/4)$-kernels from the input point set P. The running time is bounded by $O(n + k^2/\varepsilon^{d-1})$. The algorithm also leads to a one-pass algorithm for computing (k, ε)-kernels. We tested our algorithm on a variety of inputs for $d \leq 8$; the empirical results show that it works well in low dimensions in terms of both the size of the kernel and the running time.

Our result immediately implies improved approximation algorithms on a wide range of problems discussed in [19]. To name a few, we can compute an ε-approximation of the diameter with k outliers in $O(n + k^2/\varepsilon^{d-1})$ time, an ε-approximation of the minimum-width spherical shell with k outliers in $O(n + k^{2d+1}/\varepsilon^{2d(d+1)})$ time, and a subset of size $O(k/\varepsilon^d)$ for a set of linearly moving points in $\mathbb{R}^d$ so that at any time the diameter (width, smallest-enclosing box, etc.) with k outliers of this subset is an ε-approximation of that of the original moving point set.

In Sect. 3 we present an incremental algorithm for shape fitting with k outliers, which is an extension of the incremental algorithm by Yu et al. [24]. The algorithm works by repeatedly *grating* points from the original point set into a working set; the points that violate the current solution for the working set the most are selected by the algorithm. We prove that the number of iterations of the algorithm is $O((k^2/\varepsilon^{d-1})^{d-1})$, which is independent of n. Our empirical results show that the algorithm converges fairly quickly in practice. Interestingly, while the algorithm itself does not make explicit use of (k, ε)-kernels at all, its analysis crucially relies on the properties of our new algorithm for constructing (k, ε)-kernels.

2 Construction of (k, ε)-Kernel

In this section we describe an iterative algorithm for constructing a (k, ε)-kernel for a set P of n points in $\mathbb{R}^d$. Without loss of generality, we assume that $\varepsilon \leq 1/2$.

2.1 Algorithm

Set $\delta = \varepsilon/4$. Our algorithm consists of $2k + 1$ iterations. At the beginning of the ith iteration, for $0 \leq i \leq 2k$, we have a set $P_i \subseteq P$; initially $P_0 = P$. We compute a δ-kernel $\mathcal{T}_i$ of P_i, using an existing algorithm for computing δ-kernels [7, 14, 24]. We set $P_{i+1} = P_i \setminus \mathcal{T}_i$. After $2k + 1$ iterations, the algorithm returns $\mathcal{S} = \bigcup_{i=0}^{2k} \mathcal{T}_i$ as the desired (k, ε)-kernel.

Intuitively, $\mathcal{T}_i$ approximates the extent measure of P_i. By peeling away $\mathcal{T}_i$ from P_i, important points (in the sense of approximating the extent measures) on the next level of P get "exposed" and can then be subsequently captured in the next iteration of the algorithm. By repeating this peeling process enough times, the union of these point sets approximates the extents of all the first k levels. Similar peeling ideas have been used for halfspace range searching [3, 16, 17] and computing k-hulls [18]. However, unlike our approach, in which we peel away only a small number of points, these algorithms peel away all points of level 0 in each step.

2.2 Proof of Correctness

Let $u \in \mathbb{S}^{d-1}$ be an arbitrary direction. For $0 \le j < n/2$, let

$$\mathcal{V}_j(u, P) = \langle P^0[u], P^1[u], \ldots, P^j[u], P_j[u], P_{j-1}[u], \ldots, P_0[u] \rangle$$

denote the *ordered* sequence of points realizing the top/bottom j levels of P in direction u. We call the ith iteration of the algorithm *successful* with respect to direction u if $\mathcal{V}_{k-1}(u, P) \cap \mathcal{T}_i \neq \emptyset$ or *unsuccessful* otherwise. Since $|\mathcal{V}_{k-1}(u, P)| = 2k$ and the algorithm consists of $2k + 1$ iterations, at least one of them is unsuccessful with respect to u.

Lemma 2.1 *If the ith iteration is unsuccessful with respect to direction u, then $\mathcal{E}(u, P_i) \le (1 + \varepsilon/2)\, \mathcal{E}_k(u, P)$. In fact,*

$$\mathbf{U}(u, P_i) \le \mathbf{U}_k(u, P) + (\varepsilon/2)\mathcal{E}_k(u, P),$$
$$\mathbf{L}(u, P_i) \ge \mathbf{L}_k(u, P) - (\varepsilon/2)\mathcal{E}_k(u, P).$$

Proof Since $\mathcal{T}_i \cap \mathcal{V}_{k-1}(u, P) = \emptyset$, we have $\mathcal{E}(u, \mathcal{T}_i) \le \mathcal{E}_k(u, P)$; see Fig. 1. By construction, $\mathcal{T}_i$ is a δ-kernel of P_i. Therefore,

$$\mathcal{E}(u, P_i) \le \mathcal{E}(u, \mathcal{T}_i)/(1 - \delta) \le (1 + \varepsilon/2)\, \mathcal{E}_k(u, P),$$

proving the first inequality of the lemma. Note that $\mathbf{U}(u, \mathcal{T}_i) \le \mathbf{U}_k(u, P)$. We have

$$\begin{aligned} \mathbf{U}(u, P_i) &\le \mathbf{U}(u, \mathcal{T}_i) + (\mathcal{E}(u, P_i) - \mathcal{E}(u, \mathcal{T}_i)) \\ &\le \mathbf{U}_k(u, P) + \delta \mathcal{E}(u, P_i) \\ &\le \mathbf{U}_k(u, P) + (\varepsilon/2)\mathcal{E}_k(u, P). \end{aligned}$$

The third claim in this lemma can be proved in a similar manner. □

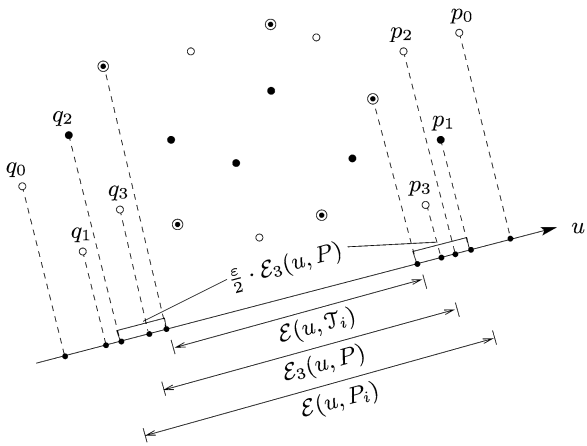

Fig. 1 Illustration of Lemma 2.1 (for $k = 3$). *Double circles* represent points in $\mathcal{T}_i$. The *union of double circles* and *solid circles* represent points in P_i. *Hollow circles* represent points in $P \setminus P_i = \bigcup_{j=0}^{i-1} \mathcal{T}_j$. Here $p_j = P^j[u]$ and $q_j = P_j[u]$

Lemma 2.2 *$\mathcal{S}$ is a (k, ε)-kernel of P.*

Proof To prove the claim, we fix an arbitrary direction $u \in \mathbb{S}^{d-1}$ and argue that

$$\mathcal{E}_{a,b}(u, \mathcal{S}) \geq (1 - \varepsilon)\mathcal{E}_{a,b}(u, P)$$

for all $0 \leq a, b \leq k$. We only discuss the case $a = b = k$; other cases can be handled by slightly modifying the argument given below.

We first show that

$$\mathbf{U}_k(u, \mathcal{S}) \geq \mathbf{U}_k(u, P) - (\varepsilon/2)\,\mathcal{E}_k(u, P). \tag{1}$$

If $P^\ell[u] \in \mathcal{S}$ for all $0 \leq \ell \leq k$, then $\mathbf{U}_k(u, \mathcal{S}) \geq \mathbf{U}_k(u, P)$ and hence (4) is clearly true. So let us assume that there exists $\ell \leq k$ such that $P^\ell[u] \notin \mathcal{S}$. Observe that for any iteration i, we must have $P^\ell[u] \in P_i$.

We define

$$Q = \left\{p \in P \mid \langle u, p\rangle \geq \mathbf{U}_k(u, P) - (\varepsilon/2)\,\mathcal{E}_k(u, P)\right\}.$$

Then (1) is equivalent to $|\mathcal{S} \cap Q| \geq k + 1$.

Consider the ith iteration of the algorithm that is unsuccessful with respect to direction u. Since $P^\ell[u] \in P_i$ and $\mathcal{T}_i$ is a δ-kernel of P_i,

$$\begin{aligned}\mathbf{U}(u, \mathcal{T}_i) &\geq \mathbf{U}(u, P_i) - \delta\,\mathcal{E}(u, P_i)\\ &\geq \mathbf{U}_\ell(u, P) - (\varepsilon/4)(1 + \varepsilon/2)\,\mathcal{E}_k(u, P) \quad \text{(by Lemma 2.1)}\\ &\geq \mathbf{U}_k(u, P) - (\varepsilon/2)\,\mathcal{E}_k(u, P).\end{aligned}$$

Hence ${\mathcal{T}_i}^0[u] \in Q$. Furthermore, since this iteration is unsuccessful, ${\mathcal{T}_i}^0[u] \notin \{P^0[u], \ldots, P^{k-1}[u]\}$, implying that $|\mathcal{T}_i \cap (Q \setminus \{P^0[u], \ldots, P^{k-1}[u]\})| \geq 1$.

Let m be the total number of successful iterations of the algorithm. Then $|\mathcal{S} \cap \mathcal{V}_{k-1}(u, P)| \geq m$ and therefore $|\mathcal{S} \cap \{P^0[u], \ldots, P^{k-1}[u]\}| \geq m - k$. Furthermore, as there are $2k + 1 - m$ unsuccessful iterations, the preceding argument implies that $|\mathcal{S} \cap (Q \setminus \{P^0[u], \ldots, P^{k-1}[u]\})| \geq 2k + 1 - m$. Hence, $|\mathcal{S} \cap Q| \geq (m - k) + (2k + 1 - m) = k + 1$, which in turn implies (1).

Using a similar argument, we can prove that

$$\mathbf{L}_k(u, \mathcal{S}) \leq \mathbf{L}_k(u, P) + (\varepsilon/2)\,\mathcal{E}_k(u, P). \tag{2}$$

Putting (1) and (2) together, we get that

$$\mathcal{E}_k(u, \mathcal{S}) = \mathbf{U}_k(u, \mathcal{S}) - \mathbf{L}_k(u, \mathcal{S}) \geq (1 - \varepsilon)\mathcal{E}_k(u, P).$$

Since u is an arbitrary direction, $\mathcal{S}$ is indeed a (k, ε)-kernel of P. □

2.3 Time Complexity

Chan [14] has shown that a δ-kernel of size $O(1/\delta^{(d-1)/2})$ can be computed in $O(n + 1/\delta^{d-1})$ time. Using this result, we obtain an algorithm for computing (k, ε)-kernels of size $O(k/\delta^{(d-1)/2})$ with running time $O(nk + k/\varepsilon^{d-1})$. We can improve

the running time to $O(n + k^2/\varepsilon^{d-1})$, using the following observation: for any point set $P \subset \mathbb{R}^d$, if $\mathcal{R} \subset P$ is a (k, ε_1)-kernel of P, and $\mathcal{S} \subset \mathcal{R}$ is a (k, ε_2)-kernel of $\mathcal{R}$, then $\mathcal{S}$ is a $(k, \varepsilon_1 + \varepsilon_2)$-kernel of P.

We first invoke the $O(n + k/\varepsilon^{d-1})$-time algorithm of Har-Peled and Wang [19] to compute a $(k, \varepsilon/2)$-kernel $\mathcal{R}$ of P of size $O(k/\varepsilon^{d-1})$, and then apply the above $O(nk + k/\varepsilon^{d-1})$-time algorithm on $\mathcal{R}$ to compute a $(k, \varepsilon/2)$-kernel $\mathcal{S}$ of $\mathcal{R}$ of size $O(k/\varepsilon^{(d-1)/2})$. The resulting set $\mathcal{S}$ is the desired (k, ε)-kernel of P, and the total running time is bounded by $O(n + k^2/\varepsilon^{d-1})$. We conclude with the following theorem.

Theorem 2.3 *Given a set P of n points in $\mathbb{R}^d$ and parameters $k, \varepsilon > 0$, one can compute, in $O(n + k^2/\varepsilon^{d-1})$ time, a (k, ε)-kernel of P of size $O(k/\varepsilon^{(d-1)/2})$.*

It is easy to verify that for a point set P' which is an $\Omega(\sqrt{\varepsilon})$-net of the unit hypersphere (i.e., the minimum distance in P' is $\Omega(\sqrt{\varepsilon})$), all points of P' must be in every $(0, \varepsilon)$-kernel of P'. By replicating $k+1$ times every point of P' and perturbing slightly, the resulting point set P has the property that any (k, ε)-kernel of P must contain $\Omega(k/\varepsilon^{(d-1)/2})$ points. Thus, in the worst case, a (k, ε)-kernel for P is of size $\Omega(k/\varepsilon^{(d-1)/2})$, matching the upper bound given in Theorem 2.3.

We also note that performing $2k + 1$ iterations in the above algorithm is not only sufficient but also necessary to compute a (k, ε)-kernel. For example, in $\mathbb{R}^2$, consider $\Omega(n)$ (slightly perturbed) copies of the two points $(-1, 0)$ and $(1, 0)$ on the x-axis, together with the following $2k + 2$ points on the y-axis: $(0, 1/\varepsilon^{k-1}), (0, 1/\varepsilon^{k-2}), \ldots, (0, 1)$; $(0, -\varepsilon), (0, -\varepsilon^2), \ldots, (0, -\varepsilon^k)$; $(0, -\varepsilon^{k+1})$, $(0, \varepsilon^{k+1})$. If the number of iterations is $2k$, the algorithm may only output the first $2k$ points listed above along the y-axis together with a set of other points on the x-axis, which is clearly not a (k, ε)-kernel in the y-direction.

2.4 Extensions

One-pass Algorithms In many applications it is desirable to compute a certain function in a single pass over the input data, using a small working memory and processing each point quickly. Agarwal et al. [7], Agarwal and Yu [2], and Chan [14] described such one-pass algorithms for computing ε-kernels. Our (k, ε)-kernel algorithm suggests how to develop a one-pass algorithm for computing a (k, ε)-kernel by using such an algorithm for ε-kernel as a subroutine. Suppose there is a one-pass algorithm $\mathcal{A}$ that computes ε-kernels using $N(\varepsilon)$ space and $T(\varepsilon)$ time per point. To compute a (k, ε)-kernel of a point set P in one pass, we proceed as follows. We simultaneously run $2k + 1$ instances of $\mathcal{A}$, namely $\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_{2k}$, each of which maintains an $(\varepsilon/4)$-kernel $\mathcal{T}_i$ $(0 \le i \le 2k)$ of its own input seen so far. The input of $\mathcal{A}_0$ is $P_0 = P$, and the input P_i of $\mathcal{A}_i$, for $i \ge 1$, is initially empty. The algorithm $\mathcal{A}_0$ processes each point in P_0 in turn. For $i \ge 1$, we insert a point $p \in P_{i-1}$ into P_i whenever any of the following two events happens:

1. p is not added into $\mathcal{T}_{i-1}$ after being processed by $\mathcal{A}_{i-1}$;
2. p is deleted from $\mathcal{T}_{i-1}$ by $\mathcal{A}_{i-1}$.

It is easy to see that in the end $P_i = P_{i-1} \setminus \mathcal{T}_{i-1}$ and $\mathcal{T}_i$ is an $(\varepsilon/4)$-kernel of P_i, for each $0 \le i \le 2k$. Therefore, $\mathcal{S} = \bigcup_{i=0}^{2k} \mathcal{T}_i$ is a (k, ε)-kernel of P as desired. The total

space needed is $O(k \cdot N(\varepsilon/4))$ and the amortized time to process each point in P is $O(k \cdot T(\varepsilon/4))$. Thus we obtain the following result.

Theorem 2.4 *Given a one-pass algorithm for computing ε-kernels in $N(\varepsilon)$ space and $T(\varepsilon)$ time per point, there is a one-pass algorithm for computing (k, ε)-kernels in $O(k \cdot N(\varepsilon/4))$ space and $O(k \cdot T(\varepsilon/4))$ amortized time per point.*

Polynomials Let $\mathcal{F}$ be a family of d-variate polynomials. The (k, ℓ)-*extent* of $\mathcal{F}$ at $x \in \mathbb{R}^d$, denoted by $\mathcal{E}_{k,\ell}(x, \mathcal{F})$, is defined by

$$\mathcal{E}_{k,\ell}(x, \mathcal{F}) = f_i(x) - f_j(x),$$

where f_i (resp. f_j) is the function in $\mathcal{F}$ that has the k-th largest (resp. ℓ-th smallest) value in the set $\mathcal{F}(x) = \{f(x) \mid f \in \mathcal{F}\}$. A subset $\mathcal{G} \subseteq \mathcal{F}$ is a (k, ε)-*kernel* of $\mathcal{F}$ if for any $0 \le a, b \le k$ and any $x \in \mathbb{R}^d$,

$$(1 - \varepsilon)\mathcal{E}_{a,b}(x, \mathcal{F}) \le \mathcal{E}_{a,b}(x, \mathcal{G}) \le \mathcal{E}_{a,b}(x, \mathcal{F}).$$

We say that the *dimension of linearization* of $\mathcal{F}$ is m if there exists a map $\varphi : \mathbb{R}^d \to \mathbb{R}^m$ so that each function $f \in \mathcal{F}$ maps to a linear function $h_f : \mathbb{R}^m \to \mathbb{R}$ in the sense that $f(x) = h_f(\varphi(x))$ for all $x \in \mathbb{R}^d$. Using Theorem 2.3 together with the standard linearization and duality transforms as described in [7], we immediately have the following.

Theorem 2.5 *Let $\mathcal{F}$ be a family of n polynomials, and let m be the dimension of linearization of $\mathcal{F}$. Given parameters $k, \varepsilon > 0$, one can compute a (k, ε)-kernel of $\mathcal{F}$ of size $O(k/\varepsilon^{m/2})$ in $O(n + k^2/\varepsilon^m)$ time.*

Roots of Polynomials To compute (k, ε)-kernels of fractional powers of polynomials, we need the following observation from [24] (see also [7]):

Lemma 2.6 *Let $0 < \varepsilon < 1$, $0 \le a \le A \le B \le b$, and r be a positive integer. If $B^r - A^r \ge (1 - \varepsilon^r)(b^r - a^r)$, then $B - A \ge (1 - \varepsilon)(b - a)$.*

Hence, in order to compute a (k, ε)-kernel of $\{f_1^{1/r}, \ldots, f_n^{1/r}\}$, where each f_i is a polynomial and r is a positive integer, it is sufficient to compute a (k, ε^r)-kernel of $\{f_1, \ldots, f_n\}$. Applying Theorem 2.5, we then have the following.

Theorem 2.7 *Let $\mathcal{F} = \{f_1^{1/r}, \ldots, f_n^{1/r}\}$ be a family of n functions, where each f_i is a polynomial and r is a positive integer. Let m be the dimension of linearization of $\{f_1, \ldots, f_n\}$. Given parameters $k, \varepsilon > 0$, one can compute a (k, ε)-kernel of $\mathcal{F}$ of size $O(k/\varepsilon^{rm/2})$ in $O(n + k^2/\varepsilon^{rm})$ time.*

Theorems 2.5 and 2.7 immediately imply improved results for various shape-fitting problems mentioned in [19], some of which have been listed in Introduction. The details can be found in [19].

3 Incremental Shape-Fitting Algorithm

In this section we present a simple incremental algorithm for shape fitting with k outliers. Compared to the shape-fitting algorithms derived directly from Theorems 2.5 and 2.7, the incremental algorithm does not enjoy a better bound on the running time, but usually performs faster in practice. The algorithm does not make explicit use of (k, ε)-kernels. However, by exploiting the construction of (k, ε)-kernels from the previous section, we show that the number of iterations performed by the algorithm is independent of the input size n. We first describe and analyze the algorithm for the special case in which we wish to find a minimum-width slab that contains all but at most k points of a point set. We then show that the same approach can be extended to a number of other shapes, including cylinders, spherical shells, and cylindrical shells.

3.1 Algorithm

A *slab* $\sigma \subseteq \mathbb{R}^d$ is the region bounded by two parallel hyperplanes. The *width* of σ is the distance between the two hyperplanes. The hyperplane passing through the middle of σ is called the *center hyperplane* of σ. For a given parameter $c > 0$, we will use $c \cdot \sigma$ to denote the slab obtained by scaling σ by the factor of c with respect to its center hyperplane. Let $u_\sigma \in \mathbb{S}^{d-1}$ denote the direction in the upper hemisphere normal to the hyperplanes bounding σ.

Let $A_{\text{opt}}(R, k)$ be an algorithm that returns a slab of the minimum width that contains all but at most k points of R. The incremental algorithm proceeds as follows. We start with an arbitrary subset $R \subseteq P$ of constant size and compute $\sigma = A_{\text{opt}}(R, k)$. If $\frac{1}{1-\varepsilon} \cdot \sigma$ can cover all but at most k points of P, then we stop because we have found an ε-approximation of $A_{\text{opt}}(P, k)$. Otherwise, we add the points of $\mathcal{V}_k(u_\sigma, P)$ (as defined in Sect. 2.2) to R and repeat the above step.

Note that the algorithm always terminates. If the number of iterations of the algorithm is small, its running time would also be small. We next prove a bound on the number of iterations that is independent of n.

3.2 Analysis

We will show that there exists a family $\mathcal{H}$ of $O(k^2/\varepsilon^{d-1})$ great hyperspheres on $\mathbb{S}^{d-1}$ with the following property: the algorithm stops as soon as it computes a slab σ_1 such that, for some slab σ_2 computed in an earlier iteration, u_{σ_1} and u_{σ_2} lie in the same cell of the arrangement $\mathcal{A}(\mathcal{H})$ of $\mathcal{H}$. This would immediately imply an

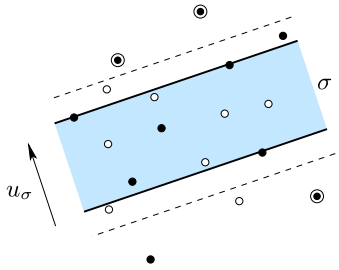

Fig. 2 One iteration of the incremental algorithm (for $k = 1$). *Solid circles* represent points in R, and *double circles* represent points to be added into R in this iteration

$O((k^2/\varepsilon^{d-1})^{d-1})$ bound on the number of iterations. First we prove a useful property of affine transforms. We then describe how to choose the great hyperspheres. Finally we prove the desired property of the chosen hyperspheres and the convergence of the algorithm.

We write an affine transform $\tau : \mathbb{R}^d \to \mathbb{R}^d$ as $\tau(x) = \boldsymbol{A}^T \cdot x + q_0$ for $x \in \mathbb{R}^d$, where $\boldsymbol{A}$ is a $d \times d$ nonsingular matrix and $q_0 \in \mathbb{R}^d$ is a fixed vector. Given τ, let $\widetilde{\tau} : \mathbb{S}^{d-1} \to \mathbb{S}^{d-1}$ be the map defined by $\widetilde{\tau}(u) = \boldsymbol{A}^{-1}u/\|\boldsymbol{A}^{-1}u\|$ for $u \in \mathbb{S}^{d-1}$. If the transform τ is clear from the context, we simply use $\widetilde{u}$ to denote $\widetilde{\tau}(u)$.

Lemma 3.1 *Let $\tau : \mathbb{R}^d \to \mathbb{R}^d$ be an affine transform. For any direction $u \in \mathbb{S}^{d-1}$, any four points $p, q, r, s \in \mathbb{R}^d$, and any parameter $c \in \mathbb{R}$,*

$$\langle u, p - q\rangle \le c \cdot \langle u, r - s\rangle \quad \Longleftrightarrow \quad \langle \widetilde{u}, \tau(p) - \tau(q)\rangle \le c \cdot \langle \widetilde{u}, \tau(r) - \tau(s)\rangle.$$

In particular, for any point set P and $\widetilde{P} = \tau(P)$, we have $\widetilde{P}^i[\widetilde{u}] = \tau(P^i[u])$ for $0 \le i < |P|$.

Proof Suppose $\tau(x) = \boldsymbol{A}^T \cdot x + q_0$ for $x \in \mathbb{R}^d$. Then

$$\begin{aligned}\langle \widetilde{u}, \tau(p) - \tau(q)\rangle &= \langle \boldsymbol{A}^{-1}u/\|\boldsymbol{A}^{-1}u\|, \boldsymbol{A}^T(p - q)\rangle \\ &= u^T(\boldsymbol{A}^T)^{-1}\boldsymbol{A}^T(p - q)/\|\boldsymbol{A}^{-1}u\| \\ &= u^T(p - q)/\|\boldsymbol{A}^{-1}u\| = \langle u, p - q\rangle/\|\boldsymbol{A}^{-1}u\|.\end{aligned}$$

Similarly, we have $\langle \widetilde{u}, \tau(r) - \tau(s)\rangle = \langle u, r - s\rangle/\|\boldsymbol{A}^{-1}u\|$. Hence the first claim of the lemma follows. Setting $c = 0$ in the first claim, we know that $\langle u, p\rangle \le \langle u, q\rangle$ if and only if $\langle \widetilde{u}, \tau(p)\rangle \le \langle \widetilde{u}, \tau(q)\rangle$. The second claim then follows. □

We need the following two lemmas to describe how to choose the desired family of great hyperspheres.

Lemma 3.2 *Let S be a set of n points in $\mathbb{R}^d$. There exists a set H of $O(n^2)$ great hyperspheres in $\mathbb{S}^{d-1}$ so that for any $u, v \in \mathbb{S}^{d-1}$ lying in the same cell of $\mathcal{A}(H)$, we have $S^i[u] = S^i[v]$, for $i = 0, \ldots, n - 1$.*

Proof For any pair of points $p, q \in S$, let h_{pq} be the great hypersphere in $\mathbb{S}^{d-1}$, defined by the equation

$$\langle u, p\rangle = \langle u, q\rangle, \quad u \in \mathbb{S}^{d-1}.$$

We let $H = \{h_{pq} \mid p, q \in S\}$. Clearly $|H| = O(n^2)$. Consider any cell $\Delta \in \mathcal{A}(H)$. By construction, it is easy to see that the relative ordering of the elements in $\{\langle u, p\rangle \mid p \in S\}$ is the same for all $u \in \Delta$. Hence, $S^i[u] = S^i[v]$ for any $u, v \in \Delta$, as desired. □

Lemma 3.3 *Let S be a set of n points in $\mathbb{R}^d$ whose affine hull spans $\mathbb{R}^d$, and let $\delta > 0$ be a parameter. There exist an affine transform τ and a set H of $O(1/\delta)$ great*

hyperspheres in $\mathbb{S}^{d-1}$ *so that for any* $u, v \in \mathbb{S}^{d-1}$ *lying in the same cell of* $\mathcal{A}(H)$ *and for any two points* $p, q \in S$,

$$|\langle \widetilde{u} - \widetilde{v}, \tau(p) - \tau(q)\rangle| \le \delta \cdot \mathcal{E}(\widetilde{v}, \tau(S)).$$

Proof Let $\mathbb{B}^d \subseteq \mathbb{R}^d$ be a unit ball centered at the origin. By John's Ellipsoid Theorem [23], there exists an affine transform τ so that $(1/d) \cdot \mathbb{B}^d \subseteq \mathrm{conv}(\tau(S)) \subseteq \mathbb{B}^d$.

Agarwal et al. [7] proved that there exists a set H of $O(1/\delta)$ great hyperspheres in $\mathbb{S}^{d-1}$, such that for any $u, v \in \mathbb{S}^{d-1}$ lying in the same cell of $\mathcal{A}(H)$, $\|\widetilde{u} - \widetilde{v}\| \le \delta/d$. Note that $\mathcal{E}(\widetilde{v}, \tau(S)) \ge 2/d$, and for any $p, q \in S$, $\|\tau(p) - \tau(q)\| \le 2$. Thus,

$$\begin{aligned} |\langle \widetilde{u} - \widetilde{v}, \tau(p) - \tau(q)\rangle| &\le \|\widetilde{u} - \widetilde{v}\| \cdot \|\tau(p) - \tau(q)\| \\ &\le 2\delta/d \le \delta \cdot \mathcal{E}(\widetilde{v}, \tau(S)), \end{aligned}$$

as claimed. □

We assume $\varepsilon \le 1/2$. Fix $\delta = \varepsilon/6 \le 1/12$. Let $\mathcal{S}$ be a (k, δ)-kernel of P computed by the algorithm in Sect. 2.1, and let $\mathcal{X} = P \setminus \mathcal{S}$. Using Lemmas 3.2 and 3.3, we construct a decomposition of $\mathbb{S}^{d-1}$ as follows. For each point $p \in \mathcal{S}$, let H_p be a family of $O(1/\delta)$ great hyperspheres that satisfy Lemma 3.3 for $\mathcal{X} \cup \{p\}$, and let τ_p be the corresponding affine transform. Let $\Gamma = \{\tau_p \mid p \in \mathcal{S}\}$. Let G be the set of $O(|\mathcal{S}|^2)$ great hyperspheres that satisfy Lemma 3.2 for the set $\mathcal{S}$. Set

$$\mathcal{H} = G \cup \Big(\bigcup_{p \in \mathcal{S}} H_p \Big).$$

Note that $|\mathcal{H}| = O(|\mathcal{S}|^2 + |\mathcal{S}|/\delta) = O(k^2/\varepsilon^{d-1})$. The number of cells in $\mathcal{A}(\mathcal{H})$ is $O(|\mathcal{H}|^{d-1}) = O((k^2/\varepsilon^{d-1})^{d-1})$.

Next we prove crucial properties of the decomposition $\mathcal{A}(\mathcal{H})$.

Lemma 3.4 *Let* $u \in \mathbb{S}^{d-1}$ *be a direction. Set* $Q = \mathcal{X} \cup \{\mathcal{S}^k[u]\}$. *For any affine transform* τ, *we have*

$$\mathcal{E}(\widetilde{u}, \tau(Q)) \le (1 + \delta) \cdot \mathcal{E}_k(\widetilde{u}, \tau(P)). \tag{3}$$

Proof Since the algorithm described in Sect. 2.1 performs $2k + 1$ iterations, at least one of them, say the iteration i, was unsuccessful with respect to direction u. By Lemma 2.1 and the fact $\mathcal{X} \subseteq P_i$, we know that

$$\mathbf{U}(u, \mathcal{X}) \le \mathbf{U}(u, P_i) \le \mathbf{U}_k(u, P) + (\delta/2) \cdot \mathcal{E}_k(u, P), \tag{4}$$

$$\mathbf{L}(u, \mathcal{X}) \ge \mathbf{L}(u, P_i) \ge \mathbf{L}_k(u, P) - (\delta/2) \cdot \mathcal{E}_k(u, P). \tag{5}$$

Therefore,

$$\begin{aligned} \mathcal{E}\big(u, \mathcal{X} \cup \{\mathcal{S}^k[u]\}\big) &\le \max\big(\langle u, \mathcal{S}^k[u]\rangle, \mathbf{U}(u, \mathcal{X})\big) - \min\big(\langle u, \mathcal{S}_k[u]\rangle, \mathbf{L}(u, \mathcal{X})\big) \\ &= \max\big(\mathbf{U}_k(u, \mathcal{S}), \mathbf{U}(u, \mathcal{X})\big) - \min\big(\mathbf{L}_k(u, \mathcal{S}), \mathbf{L}(u, \mathcal{X})\big) \end{aligned}$$

$$\begin{aligned}
&\le \big(\mathbf{U}_k(u,P) + (\delta/2)\cdot\mathcal{E}_k(u,P)\big)\\
&\quad - \big(\mathbf{L}_k(u,P) - (\delta/2)\cdot\mathcal{E}_k(u,P)\big) \quad \text{(by (4) and (5))}\\
&\le (1+\delta)\mathcal{E}_k(u,P).
\end{aligned}$$

Hence by Lemma 3.1, $\mathcal{E}(\widetilde{u},\tau(Q)) \le (1+\delta)\cdot\mathcal{E}_k(\widetilde{u},\tau(P))$. □

Lemma 3.5 *Let $u, v \in \mathbb{S}^{d-1}$ be any two directions lying in the same cell of $\mathcal{A}(\mathcal{H})$. Then, for any $0 \le a, b \le k$, we have*

$$\mathcal{E}_{a,b}(v, \mathcal{V}_k(u,P)) \ge (1-\varepsilon)\cdot\mathcal{E}_{a,b}(v,P).$$

Proof We prove the claim for the case $a, b = k$; the argument easily adapts to other cases. To this end, we show that for $\ell \le k$,

$$\langle v, P^\ell[u]\rangle \ge \mathbf{U}_k(v,P) - (\varepsilon/2)\cdot\mathcal{E}_k(v,P).$$

We start by considering the case $P^\ell[u] \in \mathcal{S}$. Observe that $\mathcal{V}_k(u,\mathcal{S}) = \mathcal{V}_k(v,\mathcal{S})$ (we remind the reader that $\mathcal{V}$ is an ordered set, as such equality here means also identical ordering by level). In particular, since $P^\ell[u]$ is clearly at level $\le \ell$ of $\mathcal{S} \subseteq P$ in direction u, $P^\ell[u]$ is also at level $\le \ell$ of $\mathcal{S}$ in direction v. Hence,

$$\langle v, P^\ell[u]\rangle \ge \mathbf{U}_\ell(v,\mathcal{S}) \ge \mathbf{U}_k(v,\mathcal{S}) \ge \mathbf{U}_k(v,P) - \delta\cdot\mathcal{E}_k(v,P),$$

where the last inequality follows from the fact that $\mathcal{S}$ is a (k,δ)-kernel of P.

Now consider the case $P^\ell[u] \in \mathcal{X}$. Set $Q = \mathcal{X} \cup \{\mathcal{S}^k[u]\}$, and let $\tau \in \Gamma$ be the affine transform that satisfies Lemma 3.3 for the set Q. Since $\ell \le k$, we have

$$\langle u, \mathcal{S}^k[u]\rangle \le \langle u, P^k[u]\rangle \le \langle u, P^\ell[u]\rangle,$$

implying that $\langle u, \mathcal{S}^k[u] - P^\ell[u]\rangle \le 0$, or equivalently by applying Lemma 3.1 with $c = 0$,

$$\langle \widetilde{u}, \tau(\mathcal{S}^k[u]) - \tau(P^\ell[u])\rangle \le 0.$$

Therefore,

$$\begin{aligned}
&\langle \widetilde{v}, \tau(\mathcal{S}^k[u]) - \tau(P^\ell[u])\rangle\\
&\quad = \langle \widetilde{u}, \tau(\mathcal{S}^k[u]) - \tau(P^\ell[u])\rangle + \langle \widetilde{v} - \widetilde{u}, \tau(\mathcal{S}^k[u]) - \tau(P^\ell[u])\rangle\\
&\quad \le \langle \widetilde{v} - \widetilde{u}, \tau(\mathcal{S}^k[u]) - \tau(P^\ell[u])\rangle. \qquad (6)
\end{aligned}$$

Note that u, v lie in the same cell of $\mathcal{A}(\mathcal{H})$, and $P^\ell[u], \mathcal{S}^k[u] \in Q = \mathcal{X} \cup \{\mathcal{S}^k[u]\}$. By applying Lemma 3.3 to the right-hand side of (6), we obtain

$$\begin{aligned}
&\langle \widetilde{v}, \tau(\mathcal{S}^k[u]) - \tau(P^\ell[u])\rangle\\
&\quad \le \delta\cdot\mathcal{E}(\widetilde{v},\tau(Q)) \le \delta(1+\delta)\cdot\mathcal{E}_k(\widetilde{v},\tau(P)) \le 2\delta\cdot\mathcal{E}_k(\widetilde{v},\tau(P)), \qquad (7)
\end{aligned}$$

where the second inequality follows from Lemma 3.4. By Lemma 3.1, (7) implies

$$\langle v, \mathcal{S}^k[u] - P^\ell[u]\rangle \le 2\delta\cdot\mathcal{E}_k(v,P). \qquad (8)$$

Observing that $\mathcal{S}^k[u] = \mathcal{S}^k[v]$ and using (8), we obtain

$$\begin{aligned}\langle v, P^\ell[u]\rangle &\geq \langle v, \mathcal{S}^k[u]\rangle - 2\delta \cdot \mathcal{E}_k(v, P) = \langle v, \mathcal{S}^k[v]\rangle - 2\delta \cdot \mathcal{E}_k(v, P)\\ &\geq \mathbf{U}_k(v, P) - 3\delta \cdot \mathcal{E}_k(v, P) \geq \mathbf{U}_k(v, P) - (\varepsilon/2) \cdot \mathcal{E}_k(v, P).\end{aligned}$$

Similarly, we can prove that for any $0 \leq \ell \leq k$, $\langle v, P_\ell[u]\rangle \leq \mathbf{L}_k(v, P) + (\varepsilon/2) \cdot \mathcal{E}_k(v, P)$.

Hence, $\mathcal{E}_k(v, \mathcal{V}_k(u, P)) \geq (1 - \varepsilon) \cdot \mathcal{E}_k(v, P)$, as claimed. □

We are now ready to bound the number of iterations of the incremental algorithm.

Theorem 3.6 *The number of iterations of the incremental algorithm for fitting the minimum-width slab with k outliers is bounded by $O((k^2/\varepsilon^{d-1})^{d-1})$, which is independent of n.*

Proof Let $u_i \in \mathbb{S}^{d-1}$ be the direction orthogonal to the slab computed in the ith iteration. We say that a cell $\Delta \in \mathcal{A}(\mathcal{H})$ is *visited* if $u_i \in \Delta$. Suppose a cell is visited by two iterations i and j during the execution of the algorithm. Assume $i < j$. Then in iteration j, we have $\mathcal{V}_k(u_i, P) \subseteq R$. Let σ be the slab returned by $A_{\text{opt}}(R, k)$ in iteration j. Then $|\sigma|$—the width of σ—is equal to $\mathcal{E}_{a,b}(u_j, R)$ for some appropriate $a, b \leq k$ with $a + b = k$. By Lemma 3.5, we have

$$|\sigma| = \mathcal{E}_{a,b}(u_j, R) \geq \mathcal{E}_{a,b}(u_j, \mathcal{V}_k(u_i, P)) \geq (1 - \varepsilon)\mathcal{E}_{a,b}(u_j, P),$$

or equivalently $\frac{1}{1-\varepsilon}|\sigma| \geq \mathcal{E}_{a,b}(u_j, P)$. This implies that the algorithm would satisfy the stopping criterion in iteration j. Thus the number of iterations is bounded by $|\mathcal{A}(\mathcal{H})| + 1 = O((k^2/\varepsilon^{d-1})^{d-1})$. □

3.3 Other Shapes

The incremental algorithm of Sect. 3.1 for computing an ε-approximation of the minimum-width slab with k outliers can be extended to fitting other shapes as well, such as minimum-width spherical shells or cylindrical shells, minimum-radius cylinders, etc. In this section we describe these extensions.

Spherical Shells and Cylindrical Shells A *spherical shell* is a closed region lying between two concentric spheres in $\mathbb{R}^d$. A *cylinderical shell* is a closed region lying between two co-axial cylinders in $\mathbb{R}^d$. Because fitting spherical shells or cylindrical shells can be formulated as computing the minimum extent of a family $\mathcal{F}$ of m-variate functions for some parameter m [7], we describe a general incremental algorithm for the latter problem. For $x \in \mathbb{R}^m$ and $0 \leq k < n$, we denote

$$\widehat{\mathcal{E}}_k(x, \mathcal{F}) = \min_{a+b=k} \mathcal{E}_{a,b}(x, \mathcal{F}),$$

where $\mathcal{E}_{a,b}(x, \mathcal{F})$ is as defined in Sect. 2.4. Let $A_{\text{opt}}(\mathcal{F}, k)$ be an algorithm that returns

$$x^* = \arg\min_{x \in \mathbb{R}^m} \widehat{\mathcal{E}}_k(x, \mathcal{F}).$$

The incremental algorithm starts by picking an arbitrary subset $\mathcal{R} \subseteq \mathcal{F}$ of constant size and compute $x^* = A_{\text{opt}}(\mathcal{R}, k)$. If $\widehat{\mathcal{E}}_k(x^*, \mathcal{R}) \geq (1-\varepsilon)\widehat{\mathcal{E}}_k(x^*, \mathcal{F})$, then $\widehat{\mathcal{E}}_k(x^*, \mathcal{R})$ is an ε-approximation of $\min_{x\in\mathbb{R}^m} \widehat{\mathcal{E}}_k(x, \mathcal{F})$ and we can stop. Otherwise, we add $\mathcal{V}_k(x^*, \mathcal{F})$—union of the $2(k+1)$ functions of $\mathcal{F}$ that attain the $k+1$ largest values and the $k+1$ smallest values in $\mathcal{F}(x^*) = \{f(x^*) \mid f \in \mathcal{F}\}$—to $\mathcal{R}$, and repeat the above step.

To analyze the above algorithm, we need the following lemma which is the dual version of Lemma 3.5.

Lemma 3.7 *Let $\mathcal{F}$ be a finite family of m-variate linear functions, and $0 < \delta \leq 1/2$ be a parameter. Then there exists a set $\mathcal{H}$ of $O(k^2/\delta^m)$ hyperplanes in $\mathbb{R}^m$ such that for any $u, v \in \mathbb{R}^m$ lying in the same cell of $\mathcal{A}(\mathcal{H})$, and any $0 \leq a, b \leq k$, we have*

$$\mathcal{E}_{a,b}(v, \mathcal{V}_k(u, \mathcal{F})) \geq (1-\delta) \cdot \mathcal{E}_{a,b}(v, \mathcal{F}).$$

Lemma 3.8 *Let $\mathcal{F}$ be a finite family of m-variate polynomials that admits a linearization of dimension ℓ, and $0 < \delta \leq 1/2$ be a parameter. Then there exists a decomposition of $\mathbb{R}^m$ into $O(k^{2m}/\delta^{m\ell})$ cells such that for any $u, v \in \mathbb{R}^m$ lying in the same cell of the decomposition, and any $0 \leq a, b \leq k$, we have*

$$\mathcal{E}_{a,b}(v, \mathcal{V}_k(u, \mathcal{F})) \geq (1-\delta) \cdot \mathcal{E}_{a,b}(v, \mathcal{F}).$$

Proof Let $\varphi : \mathbb{R}^m \to \mathbb{R}^\ell$ be the map so that each function $f \in \mathcal{F}$ maps to a linear function h_f in the sense that $f(x) = h_f(\varphi(x))$ for all $x \in \mathbb{R}^m$. Note that $\Gamma = \{\varphi(x) \mid x \in \mathbb{R}^m\}$, the image of φ, is an m-dimensional surface in $\mathbb{R}^\ell$. Let $\mathcal{F}' = \{h_f \mid f \in \mathcal{F}\}$. Applying Lemma 3.7 to $\mathcal{F}'$, we obtain a set $\mathcal{H}$ of $O(k^2/\delta^\ell)$ hyperplanes in $\mathbb{R}^\ell$. Set $\mathcal{H}^{-1} = \{h^{-1} = \varphi^{-1}(h \cap \Gamma) \mid h \in \mathcal{H}\}$, where each $h^{-1} \in \mathcal{H}^{-1}$ is an $(m-1)$-dimensional algebraic surface in $\mathbb{R}^m$. If $u, v \in \mathbb{R}^m$ lie in the same cell of $\mathcal{A}(\mathcal{H}^{-1})$, then $\varphi(u), \varphi(v)$ lie in the same cell of $\mathcal{A}(\mathcal{H})$. Since $f(x) = h_f(\varphi(x))$ for all $f \in \mathcal{F}$ and $x \in \mathbb{R}^m$, Lemma 3.7 implies that $\mathcal{E}_{a,b}(v, \mathcal{V}_k(u, \mathcal{F})) \geq (1-\delta) \cdot \mathcal{E}_{a,b}(v, \mathcal{F})$. The lemma now follows because $\mathcal{A}(\mathcal{H}^{-1})$ induces a decomposition of $\mathbb{R}^m$ into $O((k^2/\delta^\ell)^m)$ cells [1]. □

Theorem 3.9 *Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a family of m-variate nonnegative functions, and $0 < \varepsilon \leq 1/2$ be a parameter. Suppose there exists an m-variate positive function $\psi(x)$ and an integer $r \geq 1$, so that each $g_i(x) = \psi(x) f_i^r(x)$ is a polynomial. Furthermore, suppose $\mathcal{G} = \{g_1, \ldots, g_n\}$ admits a linearization of dimension ℓ. Then there exists a decomposition $\mathcal{D}$ of $\mathbb{R}^m$ into $O(k^{2m}/\varepsilon^{rm\ell})$ cells such that for any $u, v \in \mathbb{R}^m$ lying in the same cell of $\mathcal{D}$, and any $0 \leq a, b \leq k$, we have*

$$\mathcal{E}_{a,b}(v, \mathcal{V}_k(u, \mathcal{F})) \geq (1-\varepsilon) \cdot \mathcal{E}_{a,b}(v, \mathcal{F}).$$

In addition, the incremental algorithm computes an ε-approximation of $\min_{x\in\mathbb{R}^m} \widehat{\mathcal{E}}_k(x, \mathcal{F})$ in $O(k^{2m}/\varepsilon^{rm\ell})$ iterations.

Proof We first make the following observation: for any $\delta \leq 1$, $1 \leq i, j, h, \ell \leq n$, and $x \in \mathbb{R}^m$,

$$g_i(x) - g_j(x) \geq (1-\delta)\big(g_h(x) - g_\ell(x)\big)$$
$$\Leftrightarrow \quad f_i^r(x) - f_j^r(x) \geq (1-\delta)\big(f_h^r(x) - f_\ell^r(x)\big). \tag{9}$$

An immediate consequence of (9) is that $g_i(x) \geq g_j(x)$ if and only if $f_i(x) \geq f_j(x)$.

Consider the decomposition $\mathcal{D}$ of $\mathbb{R}^m$ obtained by applying Lemma 3.8 to the family $\mathcal{G}$ with parameter $\delta = \varepsilon^r$. Note that $|\mathcal{D}| = O(k^{2m}/\varepsilon^{rm\ell})$. For any $u, v \in \mathbb{R}^m$ lying in the same cell of $\mathcal{D}$, by Lemma 3.8 we have

$$\mathcal{E}_{a,b}(v, \mathcal{V}_k(u, \mathcal{G})) \geq (1-\varepsilon^r) \cdot \mathcal{E}_{a,b}(v, \mathcal{G}).$$

Using (9) and Lemma 2.6, we obtain that $\mathcal{E}_{a,b}(v, \mathcal{V}_k(u, \mathcal{F})) \geq (1-\varepsilon) \cdot \mathcal{E}_{a,b}(v, \mathcal{F})$, as desired.

Using the proved result and the same argument as in Theorem 3.6, we immediately obtain the second half of the theorem. □

The problem of computing the minimum-width spherical shell containing all but at most k points of a point set P in $\mathbb{R}^d$ satisfies Theorem 3.9 with $m = d$, $r = 2$, and $\ell = d+1$ [7]. Hence the incremental algorithm for this problem terminates in $k^{O(d)}/\varepsilon^{O(d^2)}$ iterations. Similarly, the incremental algorithm for computing the minimum-width cylindrical shell terminates in $k^{O(d)}/\varepsilon^{O(d^3)}$ iterations, as it satisfies Theorem 3.9 with $m = 2d-2$, $r = 2$, and $\ell = O(d^2)$ [7]. We thus obtain the following.

Corollary 3.10 *Let P be a set of n points in $\mathbb{R}^d$, and let $0 < \varepsilon \leq 1/2$ be a parameter. The incremental algorithm computes an ε-approximation of the smallest spherical shell containing all but k points of P in $k^{O(d)}/\varepsilon^{O(d^2)}$ iterations, and the smallest cylindrical shell containing all but k points of P in $k^{O(d)}/\varepsilon^{O(d^3)}$ iterations.*

Cylinders Unlike cylindrical shells and spherical shells, the problem of fitting cylinders cannot be directly formulated as computing the minimum extent of a family of functions. Instead, it can be reduced to computing $\min_{x\in\mathbb{R}^m} \mathbf{U}_k(x, \mathcal{F})$ for a family $\mathcal{F}$ of nonnegative functions, where $\mathbf{U}_k(x, \mathcal{F})$ is defined as the $(k+1)$-th largest value in the set $\mathcal{F}(x)$. The incremental algorithm for such type of problems is as follows. Let $A_{\text{opt}}(\mathcal{F}, k)$ be an algorithm that returns

$$x^* = \arg\min_{x\in\mathbb{R}^m} \mathbf{U}_k(x, \mathcal{F}).$$

The algorithm starts by picking an arbitrary subset $\mathcal{R} \subseteq \mathcal{F}$ of constant size and compute $x^* = A_{\text{opt}}(\mathcal{R}, k)$. If $\mathbf{U}_k(x^*, \mathcal{R}) \geq (1-\varepsilon) \cdot \mathbf{U}_k(x^*, \mathcal{F})$, then we can stop because $\mathbf{U}_k(x^*, \mathcal{R})$ is an ε-approximation of $\min_{x\in\mathbb{R}^m} \mathbf{U}_k(x, \mathcal{F})$. Otherwise, we add $\mathcal{U}_k(x^*, \mathcal{F})$—the $k+1$ functions of $\mathcal{F}$ that attain the $k+1$ largest values in $\mathcal{F}(x^*) = \{f(x^*) \mid f \in \mathcal{F}\}$—to $\mathcal{R}$, and repeat the above step.

Theorem 3.11 *Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a family of m-variate nonnegative functions, and $0 < \varepsilon \leq 1/2$ be a parameter. Suppose there exists an m-variate positive function $\psi(x)$ and an integer $r \geq 1$, so that each $g_i(x) = \psi(x) f_i^r(x)$ is a polynomial. Furthermore, suppose $\mathcal{G} = \{g_1, \ldots, g_n\}$ admits a linearization of dimension ℓ. Then there*

exists a decomposition $\mathcal{D}$ of $\mathbb{R}^m$ into $O(k^{2m}/\varepsilon^{m\ell})$ cells such that for any $u, v \in \mathbb{R}^m$ lying in the same cell of $\mathcal{D}$, and any $0 \le a \le k$, we have

$$\mathbf{U}_a(v, \mathcal{U}_k(u, \mathcal{F})) \ge (1-\varepsilon) \cdot \mathbf{U}_a(v, \mathcal{F}).$$

In addition, the incremental algorithm computes an ε-approximation of $\min_{x\in\mathbb{R}^m} \mathbf{U}_k(x, \mathcal{F})$ in $O(k^{2m}/\varepsilon^{m\ell})$ iterations.

Proof Let $\mathcal{G}' = \{g_1, \ldots, g_n\} \cup \{-g_1, \ldots, -g_n\}$. Since $\mathcal{G}$ admits a linearization of dimension ℓ, $\mathcal{G}'$ also admits a linearization of dimension ℓ. Let $\mathcal{D}$ be the decomposition of $\mathbb{R}^m$ obtained by applying Lemma 3.8 to $\mathcal{G}'$ with parameter $\delta = \varepsilon$. Then $|\mathcal{D}| = O(k^{2m}/\varepsilon^{m\ell})$. For any $u, v \in \mathbb{R}^m$ lying in the same cell of $\mathcal{D}$, we have

$$\mathcal{E}_{a,a}(v, \mathcal{V}_k(u, \mathcal{G}')) \ge (1-\varepsilon)\mathcal{E}_{a,a}(v, \mathcal{G}').$$

By the symmetry of $\mathcal{G}'$, we have $\mathcal{E}_{a,a}(v, \mathcal{V}_k(u, \mathcal{G}')) = 2\psi(v)(\mathbf{U}_a(v, \mathcal{U}_k(u, \mathcal{F})))^r$, and $\mathcal{E}_{a,a}(v, \mathcal{G}') = 2\psi(v)(\mathbf{U}_a(v, \mathcal{F}))^r$. Hence the above inequality implies

$$\big(\mathbf{U}_a(v, \mathcal{U}_k(u, \mathcal{F}))\big)^r \ge (1-\varepsilon)\big(\mathbf{U}_a(v, \mathcal{F})\big)^r.$$

It follows that $\mathbf{U}_a(v, \mathcal{U}_k(u, \mathcal{F})) \ge (1-\varepsilon) \cdot \mathbf{U}_a(v, \mathcal{F})$, as desired.

As a direct consequence, the second half of the theorem follows. □

Note that the cylinder problem has the same linearization as the cylindrical shell problem mentioned above. We then obtain the following.

Corollary 3.12 *Let P be a set of n points in $\mathbb{R}^d$, and let $0 < \varepsilon \le 1/2$ be a parameter. The incremental algorithm computes an ε-approximation of the smallest cylinder containing all but k points of P in $k^{O(d)}/\varepsilon^{O(d^3)}$ iterations.*

We have not tried to optimize our bounds on the number of iterations, but we believe that they can be improved. As shown in Sect. 4, the number of iterations in practice is usually much smaller than the proved bounds here.

4 Experiments

In this section, we demonstrate the effectiveness of our algorithms by evaluating their performances on various synthetic and real data. All our experiments were conducted on a Dell PowerEdge 650 server equipped with 3 GHz Pentium IV processor and 3 GB memory, running Linux 2.4.20.

Computing (k, ε)-Kernels We implemented a simpler version of our (k, ε)-kernel algorithm, which does not invoke Har-Peled and Wang's algorithm [19] first. We used an implementation of Yu et al. [24] for computing δ-kernels in each iteration. Although the worst-case running time of the algorithm is larger than that mentioned in Theorem 2.3, it is simple and works well in practice.

We used three types of synthetic inputs as well as a few large 3D geometric models [20]:

Table 1 Performance of the (k, ε)-kernel algorithm on various synthetic data with $k = 5$. Running time is measured in seconds

Input type	Input size	Approximation error				Running time			
		$d = 3$	$d = 4$	$d = 5$	$d = 8$	$d = 3$	$d = 4$	$d = 5$	$d = 8$
Sphere	10^4	0.022	0.052	0.091	0.165	2.7	4.8	7.7	20.6
	10^5	0.022	0.054	0.103	0.192	9.2	14.5	19.0	42.3
	10^6	0.024	0.055	0.100	0.224	101.4	155.6	194.7	337.3
Cylinder	10^4	0.005	0.027	0.086	0.179	2.7	4.5	7.2	20.6
	10^5	0.015	0.059	0.117	0.243	10.3	13.7	19.7	42.5
	10^6	0.019	0.058	0.125	0.283	129.9	157.9	192.6	335.4
Clustered	10^4	0.001	0.010	0.026	0.061	2.5	4.4	7.7	19.0
	10^5	0.012	0.020	0.031	0.078	7.7	11.5	16.8	36.4
	10^6	0.016	0.021	0.045	0.087	80.1	104.8	138.6	266.5

Table 2 Performance of the (k, ε)-kernel algorithm on various 3D geometric models with $k = 5$. Running time is measured in seconds

Input type	Input size	Kernel size	Approx error	Running time
Bunny	35,947	1001	0.012	3.9
Dragon	437,645	888	0.016	32.8
Buddha	543,652	1064	0.014	35.6

1. Points uniformly distributed on a sphere (*sphere*);
2. Points uniformly distributed on a cylindrical surface (*cylinder*);
3. Clustered point sets (*clustered*), consisting of 20 equal-sized clusters whose centers are uniformly distributed in the unit square and radii uniformly distributed between $[0, 0.2]$;
4. 3D geometric models: *bunny* (∼36 K points), *dragon* (∼438 K points), *buddha* (∼544 K points).

For each input data, we ran our (k, ε)-kernel algorithm with $k = 5$. The algorithm performs 11 iterations and chooses roughly 100 points for the kernel in each iteration. The output size of the algorithm varies between 800 and 1100. To compute the approximation error between the k-level extents of the kernel $\mathcal{S}$ and of the input P, we choose a set $\triangle$ of 1000 random directions from $\mathbb{S}^{d-1}$ and compute

$$\mathrm{err}_{\triangle}(P, \mathcal{S}) = \max_{u \in \triangle} \frac{\mathcal{E}_k(u, P) - \mathcal{E}_k(u, \mathcal{S})}{\mathcal{E}_k(u, P)}.$$

Tables 1 and 2 summarize the approximation error and the running time of the algorithm, for each input data. As can be seen, our algorithm works well in low dimensions both in terms of the approximation error and the running time. Our algorithm also performed quite well on several 3D geometric models. In high dimensions, the performance of our algorithm deteriorates because of the curse of dimensionality.

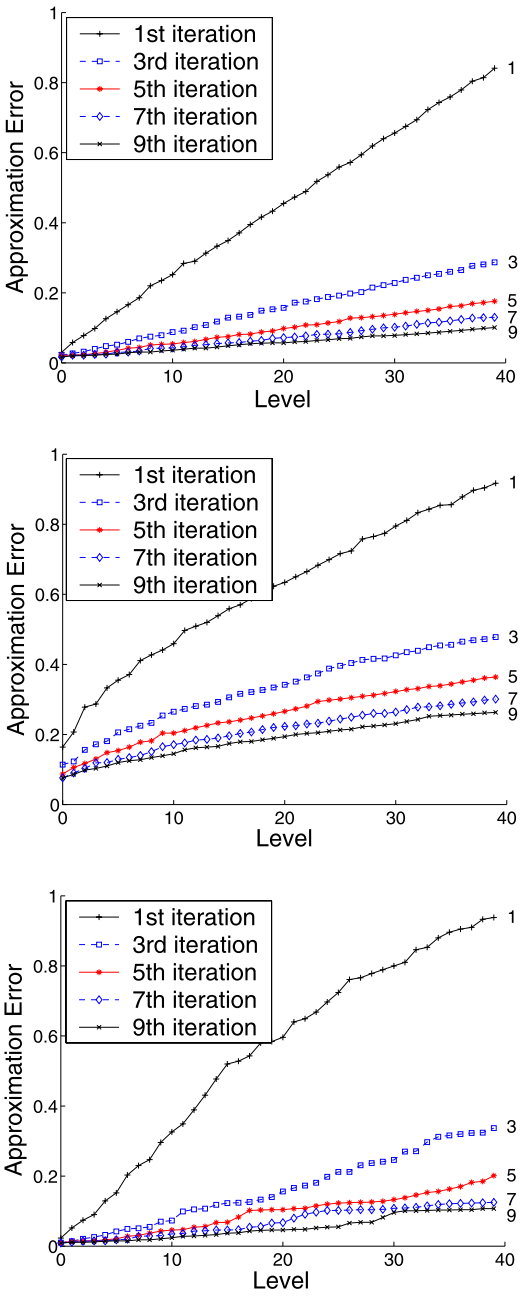

Fig. 3 Approximation errors for different levels in each iteration of the (k, ε)-kernel algorithm. **a** *Sphere* with 10^5 points in $\mathbb{R}^3$; **b** *sphere* with 10^5 points in $\mathbb{R}^5$; **c** the *budda* model. Similar results were observed for other types of inputs as well

We also recorded how the approximation error decreases for each of the first 40 levels, after each iteration of the algorithm. The results are shown in Fig. 3. Observe that the approximation error for every level monotonically decreases during the execution of the algorithm. Moreover, the error decreases rapidly in the first few it-

Table 3 Performance of the incremental algorithm for computing (a) the minimum-width annulus with $k = 10$ outliers, (b) the smallest enclosing circle with $k = 10$ outliers. The numbers of iterations performed by the algorithm are in parentheses. Running time is measured in seconds

Input width	Input size	Running time	Output width
$w = 0.05$	10^4	6.15(4)	0.0503
	10^5	14.51(5)	0.0498
	10^6	18.26(5)	0.0497
$w = 0.50$	10^4	5.74(4)	0.4987
	10^5	6.45(4)	0.4999
	10^6	22.18(5)	0.4975
$w = 5.00$	10^4	53.46(5)	4.9443
	10^5	67.26(5)	4.9996
	10^6	75.42(5)	4.9951

(a)

Input radius	Input size	Running time	Output radius
$r = 1.000$	10^4	0.05(3)	0.993
	10^5	0.14(4)	0.999
	10^6	0.41(4)	0.999

(b)

erations and then it stabilizes. For example, in our experiments for $d = 3$, the error reduces to less than 0.1 within 7 iterations even for the level $k = 40$ and then it decreases very slowly with each iteration. This phenomenon suggests that in practice it is unnecessary to run the algorithm for full $2k + 1$ iterations in order to compute (k, ε)-kernels. The larger the number of iterations is, the larger the kernel size becomes, but the approximation error does not decrease much further.

Incremental Algorithm We applied the incremental shape-fitting algorithm for computing an ε-approximate minimum-width annulus of a point set with k outliers in $\mathbb{R}^2$. We first implemented a brute-force $O(n^5)$ exact algorithm for this problem. Clearly, this algorithm is slow even on medium-sized input. Here our focus is to study the number of iterations of the incremental algorithm; a faster implementation of the exact algorithm would naturally result in a faster implementation of the incremental algorithm. We used the slow exact algorithm as a subroutine to solve the small subproblems in each iteration of the incremental algorithm. We tested this algorithm on a set of synthetic data, generated by uniformly sampling from annuli with fixed inner radius $r = 1.00$ and widths w varying from 0.05 to 5.00, and then artificially introducing $k = 10$ extra outlier points. The experimental results are summarized in Table 3a; see also Fig. 4 for a few snapshots of the running algorithm. As can be seen, the number of iterations of the incremental algorithm is never more than 5. In other words, the algorithm is able to converge to an approximate solution very quickly.

We also applied the incremental algorithm for computing an ε-approximate smallest enclosing circle of a point set with k outliers in $\mathbb{R}^2$. Again, we implemented a brute-force $O(n^4)$ exact algorithm for this problem to solve the small subproblems in each iteration; implementing a faster algorithm (such as an algorithm by Matoušek [22] or by Chan [13]) would result in a faster incremental algorithm. We tested our algorithm on a set of synthetic data, generated by uniformly sampling from a circle of radius $r = 1.00$, and then artificially introducing $k = 10$ extra outlier points. The

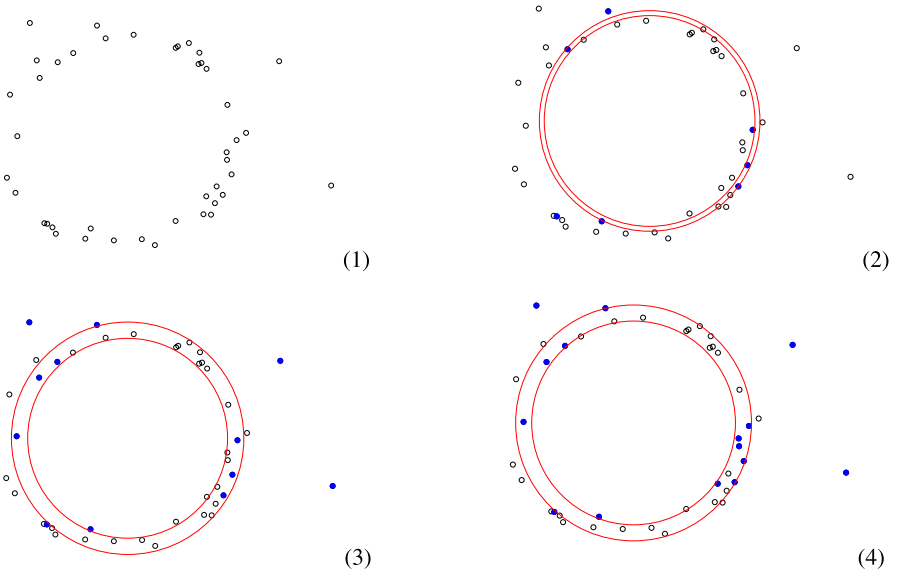

Fig. 4 Snapshots of the incremental algorithm for computing minimum-width annulus with $k = 3$ outliers on an input of size 40. *Black points* represent points in R at the beginning of each iteration

experimental results are shown in Table 3b. Similar to the annulus case, the number of iterations of the incremental algorithm is also small.

5 Conclusions

We have presented an iterative algorithm, with $O(n + k^2/\varepsilon^{d-1})$ running time, for computing a (k, ε)-kernel of size $O(k/\varepsilon^{(d-1)/2})$ for a set P of n points in $\mathbb{R}^d$. We also presented an incremental algorithm for fitting various shapes through a set of points with outliers, and exploited the (k, ε)-kernel algorithm to prove that the number of iterations of the incremental algorithm is independent of n. Both our algorithms are simple and work well in practice.

We conclude by mentioning two open problems: Can a (k, ε)-kernel of size $O(k/\varepsilon^{(d-1)/2})$ be computed in time $O(n + k/\varepsilon^{d-1})$? Can the number of iterations in the incremental algorithm for computing the minimum-width slab be improved to $O(1/\varepsilon^{(d-1)/2})$? For the first question, an anonymous referee pointed out that one can use the dynamic algorithm for ε-kernels [7] to obtain an algorithm with running time $O\big(n + k/\varepsilon^{3(d-1)/2} \cdot \text{polylog}(k, 1/\varepsilon)\big)$. This bound provides an improvement over the current running time for a sufficiently large k.

Acknowledgements The authors thank Yusu Wang for helpful discussions and two anonymous referees for constructive comments that greatly improved the presentation of the paper.

References

1. Agarwal, P.K., Sharir, M.: Arrangements and their applications. In: Sack, J.-R., Urrutia, J. (eds.) Handbook of Computational Geometry, pp. 49–119. Elsevier, Amsterdam (2000)
2. Agarwal, P.K., Yu, H.: A space-optimal data-stream algorithm for coresets in the plane. In: Proc. 23rd Annu. Sympos. Comput. Geom., pp. 1–10, 2007
3. Agarwal, P.K., Arge, L., Erickson, J., Franciosa, P., Vitter, J.S.: Efficient searching with linear constraints. J. Comput. Sys. Sci. **61**, 192–216 (2000)
4. Agarwal, P.K., Aronov, B., Har-Peled, S., Sharir, M.: Approximation and exact algorithms for minimum-width annuli and shells. Discrete Comput. Geom. **24**, 687–705 (2000)
5. Agarwal, P.K., Aronov, B., Sharir, M.: Exact and approximation algorithms for minimum-width cylindrical shells. Discrete Comput. Geom. **26**, 307–320 (2001)
6. Agarwal, P.K., Guibas, L.J., Hershberger, J., Veach, E.: Maintaining the extent of a moving point set. Discrete Comput. Geom. **26**, 353–374 (2001)
7. Agarwal, P.K., Har-Peled, S., Varadarajan, K.R.: Approximating extent measures of points. J. Assoc. Comput. Mach. **51**, 606–635 (2004)
8. Agarwal, P.K., Har-Peled, S., Varadarajan, K.: Geometric approximation via coresets. In: Goodman, J.E., Pach, J., Welzl, E. (eds.) Combinatorial and Computational Geometry. Math. Sci. Research Inst. Pub., Cambridge (2005)
9. Aronov, B., Har-Peled, S.: On approximating the depth and related problems. In: Proc. 16th ACM-SIAM Sympos. Discrete Algorithms, pp. 886–894, 2005
10. Barequet, G., Har-Peled, S.: Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. J. Algorithms **38**, 91–109 (2001)
11. Basch, J., Guibas, L.J., Hershberger, J.: Data structures for mobile data. J. Algorithms **31**, 1–28 (1999)
12. Chan, T.M.: Approximating the diameter, width, smallest enclosing cylinder and minimum-width annulus. Int. J. Comput. Geom. Appl. **12**, 67–85 (2002)
13. Chan, T.M.: Low-dimensional linear programming with violations. SIAM J. Comput. 879–893 (2005)
14. Chan, T.M.: Faster core-set constructions and data-stream algorithms in fixed dimensions. Comput. Geom. Theory Appl. **35**, 20–35 (2006)
15. Chazelle, B.: Cutting hyperplanes for divide-and-conquer. Discrete Comput. Geom. **9**, 145–158 (1993)
16. Chazelle, B., Preparata, F.P.: Halfspace range search: an algorithmic application of k-sets. Discrete Comput. Geom. **1**, 83–93 (1986)
17. Chazelle, B., Guibas, L.J., Lee, D.T.: The power of geometric duality. BIT **25**, 76–90 (1985)
18. Cole, R., Sharir, M., Yap, C.K.: On k-hulls and related problems. SIAM J. Comput. **16**, 61–77 (1987)
19. Har-Peled, S., Wang, Y.: Shape fitting with outliers. SIAM J. Comput. **33**, 269–285 (2004)
20. http://www.cc.gatech.edu/projects/large_models/. Large geometric models archive
21. Matoušek, J.: Approximate levels in line arrangements. SIAM J. Comput. **20**, 222–227 (1991)
22. Matoušek, J.: On geometric optimization with few violated constraints. Discrete Comput. Geom. **14**, 365–384 (1995)
23. Matoušek, J.: Lectures on Discrete Geometry. Springer, Heidelberg (2002)
24. Yu, H., Agarwal, P.K., Poreddy, R., Varadarajan, K.R.: Practical methods for shape fitting and kinetic data structures using coresets. Algorithmica (to appear)