Width of Points in the Streaming Model

ALEXANDR ANDONI, Microsoft Research
HUY L. NGUYỄN, Princeton University

We show how to compute the width of a dynamic set of low-dimensional points in the streaming model. In particular, we assume the stream contains both insertions of points and deletions of points to a set S, and the goal is to compute the width of the set S, namely the minimal distance between two parallel hyperplanes sandwiching the pointset S.

Our algorithm $(1+\epsilon)$-approximates the width of the set S using space polylogarithmic in the size of S and the aspect ratio of S. This is the first such algorithm that supports both insertions and deletions of points to the set S: previous algorithms for approximating the width of a pointset only supported additions [Agarwal et al. 2004; Chan 2006], or a sliding window [Chan and Sadjad 2006].

This solves an open question from the "2009 Kanpur list" of Open Problems in Data Streams, Property Testing, and Related Topics [Indyk et al. 2011].

Categories and Subject Descriptors: F.2.2 [**Nonnumerical Algorithms and Problems**]: Geometrical problems and computations

General Terms: Algorithms, Theory

Additional Key Words and Phrases: width, streaming, computational geometry, coreset

ACM Reference Format:
Andoni, A., and Nguyen, H. L. 2012. Width of points in the streaming model. ACM Trans. Algor. V, N, Article A (January YYYY), 10 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Two common geometric optimization problems are computing the diameter and the width of a set of points in the plane. These are just two out of an area of problems aimed at describing a set of points. A typical question may be: given a set of points in 2D, is it well-approximated by a line? Questions of this type — called shape fitting — are fundamental in computational geometry, computer vision, data mining, etc. While it would be difficult to summarize all previous work on the subject, we refer the reader to [Agarwal et al. 2005] who give a very good overview of the problem.

In the quest for very efficient algorithms for these problems, researchers developed efficient $(1 + \epsilon)$-approximation algorithms [Agarwal et al. 2004; Chan 2006] in the *streaming model* [Muthukrishnan 2005]. These algorithms process the stream of points — one point at a time in a sequential manner — while using only low (polylogarithmic) space. Streaming algorithms for these (and other related) problems are based on the *coreset technique*, which has been very powerful for obtaining such algo-

A preliminary version of this paper appeared in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2012. The work was done in part while A. Andoni was a postdoctoral researcher at Princeton University/CCI. A. Andoni was supported by NSF CCF 0832797. H. L. Nguyen was supported by NSF CCF 0832797 and a Gordon Wu fellowship.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

rithms for a range of geometric optimization problems (see the survey [Agarwal et al. 2005] and references therein, also [Chan 2006; Chan and Sadjad 2006]). In particular, the best algorithm for insertion-only width problem achieves space and amortized insertion time of $(1/\epsilon)^{O(d)}$ in dimension d [Chan 2006]. In the related model of "sliding window", where one has to report the width of most recent N points in the stream, the best algorithm achieves space and update time of $(1/\epsilon)^{O(d)} \log \Delta$ for points with integer coordinates bounded by Δ [Chan and Sadjad 2006].

It seems challenging though to adapt the coreset technique to the more general case of a *dynamic* set: when the stream contains both insertions of points to the set and *deletions* from the set (corresponding to the "strict turnstile model" in the streaming-speak). Without respect to the streaming model, there are "dynamic coresets" that yield a data structure that achieve, for example, $(1/\epsilon)^{O(d)} \log^{O(1)} n$ update time for the dynamic width problem [Chan 2009]. However such data structures use $\Omega(n)$ space, which is much more than the desired logarithmic-in-n space in the streaming model. [Indyk 2004] gave some of the first low-space dynamic algorithms for certain geometric problems. Presently we have efficient dynamic algorithms for some geometric problems, including the diameter [Feigenbaum et al. 2004; Indyk 2004], or the clustering and optimization problems [Frahling et al. 2005; Frahling and Sohler 2005]. Yet the dynamic width problem has so far remained open (see Question 17 in the open list [Indyk et al. 2011]).

Here we resolve this question by giving an efficient $(1+\epsilon)$-approximation algorithm for computing the width in the dynamic streaming model.

1.1. Techniques

Our algorithm relies on a certain "polynomial method". We show that it is possible to construct a (deterministic) oracle that, given a fixed line in the plane, returns an approximation to the maximal distance from the line to the points in set. Once we have such an oracle, it is possible to just enumerate over all possible lines and thus find the (approximately) best sandwiching lines. While such an enumeration would have a large runtime, we also show a randomized algorithm achieving a better runtime. The latter algorithm stores additional information about the pointset to reduce the number of candidate lines to a small (polylogarithmic) number only.

We now explain why such an oracle would be possible. The main idea is that the value of the width may be approximated by a polynomial (in the coordinates of the pointset and the parameters of the line), which has a sufficiently low degree. In particular, consider the width $W_{u,t}$ of a point set $\{p_i\}_i$ in the direction u (perpendicular to the sandwiching lines), where t is the inner product of u and a(ny) point on the mid-line (i.e., half-way between the two sandwiching lines):

$$W_{u,t} = 2 \max_i |up_i - t|.$$

Note that this may be also written as $W_{u,t} = 2\max_i |u_x x_i + u_y y_i - t|$, where $u = (u_x, u_y)$ and $p_i = (x_i, y_i)$.

Now, the polynomial arises from the following standard relation between norms (see, e.g., [Indyk et al. 2004]): one can approximate the max-norm $\|z\|_\infty = \max_i |z_i|$ by a sufficiently high p-norm $\|z\|_p = (\sum_i |z_i|^p)^{1/p}$. In our setting, z_i is the distance from the mid-line to a point i, namely $z_i = |u_x x_i + u_y y_i - t|$. In fact, the approximation is up to a factor of $n^{1/p}$ for n points (terms), and hence taking $p \approx O(\epsilon^{-1} \log n)$ yields a $(1+\epsilon)$-approximation. Writing out the resulting approximation $W_{u,t} \approx 2\left(\sum_i (u_x x_i + u_y y_i - t)^p\right)^{1/p}$, we observe that a small number of moments (up to degree p) of the pointset coordinates will suffice for evaluating this resulting polynomial for given sandwiching lines, specified by u, t.

Given the oracle, one can enumerate over all possible u, t and compute the best sandwiching lines as the width is equal to $W = \min_u \min_t W_{u,t}$. Next, we explain how we reduce the runtime to $(\epsilon^{-1} \log n)^{O(1)}$.

We start by showing how to efficiently minimize over t, for a given u. In particular, it suffices to sample a random point p from the (dynamic) set and compute $t' = up$. Then t' is a $O(W)$ additive approximation to the best t for the given u, and hence it suffices to try $O(1/\epsilon)$ discretizations of t (assuming a "guess" W). To sample a point p from the dynamic set, we use the algorithm of [Frahling et al. 2005].

The final ingredient is the algorithm for choosing the best direction u. Again, we reduce the number of candidate (near-)best u to a polylogarithmic number only. We distinguish two cases, depending on whether the instance pointset is *fat*, i.e., the width is not much smaller than the diameter. If the instance is fat, a small absolute error in the guess u, say, ϵ degrees, is sufficient to get an approximation for the width. Otherwise, if the instance is not fat, this is not sufficient. In this case, the direction minimizing width is approximately orthogonal to the direction maximizing width. Thus we can get a good guess for u by finding far points and using the direction orthogonal to the line connecting those points as a guess. This procedure only produces a reasonably good (but not $1+\epsilon$) approximation to the best direction u, but one can tweak this guess with steps of proportionate magnitude to get a $1+\epsilon$ approximation.

1.2. Preliminaries

We use the notation $[n] = \{1, 2, \ldots, n\}$. We now define the parameter width formally.

Definition 1.1. Define the *directional width* of S with respect to a unit vector (direction) u, denoted $W_u(S)$, to be

$$W_u(S) = \max_{a \in S} a \cdot u - \min_{b \in S} b \cdot u$$

The *width* of a set S, denoted by $W(S)$ is defined as the minimum directional width of S over all unit vectors u:

$$W(S) = \min_{||u||=1} W_u(S) = \min_{||u||=1} (\max_{a \in S} a \cdot u - \min_{b \in S} b \cdot u)$$

Note that the formula for computing the directional width $W_u(S)$ follows from the Hesse normal form. Also notice that even though the width and the directional width are defined over an infinite number of choices, it suffices to consider only directions orthogonal to lines going through two points in S and slabs centered around lines going through the mid-point of segments connecting two points in S. We will use this fact in the subsequent reasoning in the paper.

We assume that our pointset S comes from a discrete grid $\{1, 2, \ldots, \Delta\}^d$, and n is an upper bound on the size of S. We note that most of the paper describes the solution of the $d = 2$ case, though we address the $d > 2$ case in the last chapter of the paper. We will use the following lemma (which is similar to, say, the concommitant result of [Varadarajan and Xiao 2012, Lemma 5.1]):

LEMMA 1.2. *For any d, the minimum possible nonzero width of the point set S coming from $[\Delta]^d$ is at least $((\Delta + 3)\sqrt{d+1})^{-d-1}$.*

PROOF. Consider the optimal direction u and the two corresponding parallel hyperplanes forming the minimum width. There must exist $d+1$ points in S that are on the two hyperplanes and form a simplex. Otherwise, a smaller width can be obtained by rotating the hyperplanes. Assume $a_1, \ldots, a_t$ are on the first hyperplane and $a_{t+1}, \ldots, a_{d+1}$ are on the second hyperplane. Then $a_1 \cdot u = \cdots = a_t \cdot u = C$, $a_{t+1} \cdot u - W(S) = \cdots = a_{d+1} \cdot u - W(S) = C$ for some C. We can assume C is a real number satisfying $1 \leq |C| \leq \Delta + 3$ because if $|C| < 1$, we can consider a new set of points of the same width obtained from S by translating in every coordinate by 3 and get a new value of C different from the old value by at least $3\|u\|_1 \geq 3$. If we treat u and $W(S)$ as unknowns, and a_i and C as knowns, then we have a system of $d+1$ unknowns and $d+1$ equations. By Cramer's rule, $W(S)$ is the ratio of determinants of two $(d+1) \times (d+1)$ matrices with entries of absolute values at most $\Delta + 3$. The determinant in the numerator is C times the determinant of an integer matrix so if it is nonzero, its absolute value is at least 1. For any $(d+1) \times (d+1)$ matrix A with entries of absolute values at most $\Delta + 3$ and singular values $\lambda_1, \lambda_2, \ldots, \lambda_{d+1}$, by the AM-GM inequality applied to $\lambda_1^2, \ldots, \lambda_{d+1}^2$, we have

$$|\det(A)| = \prod_{i=1}^{d+1} \lambda_i \leq \left(\sum_{i=1}^{d+1} \lambda_i^2/(d+1)\right)^{(d+1)/2} = (\|A\|_F^2/(d+1))^{(d+1)/2} \leq ((\Delta+3)^2(d+1))^{(d+1)/2}$$

Therefore, if $W(S) > 0$ then $W(S) \geq ((\Delta + 3)\sqrt{d+1})^{-d-1}$. $\square$

2. THE ALGORITHMS

We present a low-space algorithm to process a stream of insertions and deletions of points to a dynamic set $S \subset [\Delta]^2$, and report an approximation of the width of the set S. Our algorithm has two parts:

- Maintain an oracle that, for a given vector u, can approximate the directional width $W_u(S)$.
- Approximate the width $W(S)$ by making a small number of queries u's for the above oracle.

The final algorithm is randomized. However, one can also obtain a *deterministic* algorithm for the problem in any small dimension using a powerful theorem for solving systems of polynomial equations by [Basu et al. 1996].

THEOREM 2.1 (MAIN). *Fix $\epsilon > 0$ and $n, \Delta > 1$. There exists a streaming algorithm that supports insertions and deletions of points to a set $S \subset [\Delta]^2$, $|S| \leq n$, and outputs the width of the set S, up to $1+\epsilon$ approximation, with 2/3 success probability. The algorithm uses $\mathrm{poly}(\log n\Delta, 1/\epsilon)$ space, and has $\mathrm{poly}(\log n\Delta, 1/\epsilon)$ update and evaluation time.*

2.1. An oracle for approximating the directional width

First we show how to approximate the *directional* width by maintaining a linear sketch of the point set. Let integer $k = \Theta\left(\frac{\log n}{\log(1+\epsilon)}\right)$ and k is even. The sketch simply consists of counters $T_{i,j} = \sum_{(x,y)\in S} x^i y^j$ for all $i, j \in \{0, \ldots, k\}$. Note that there are a total of $O(k^2) = O(\epsilon^{-2} \log^2 n)$ such counters, and it is trivial to maintain them in the strict turnstile streaming model.

LEMMA 2.2. *For any set $S \in [\Delta]^2$, given the counters $T_{i,j}$, $i, j \in \{0, \ldots k\}$, and any unit vector $u = (u_x, u_y)$, one can compute a $1+\epsilon$ approximation of the directional width $W_u(S)$ in time $O(\Delta^2 \cdot \mathrm{poly}(\log n, \log \Delta, 1/\epsilon))$. The algorithm is deterministic.*

PROOF. As mentioned above, the sketch consists of all counters $T_{i,j}$, for $i, j \in \{0, \ldots k\}$. The estimation algorithm outputs the following quantity, using the counters

$T_{i,j}$'s and u:

$$w = 2 \min_{t_x,t_y \in \{-2\Delta,\ldots,2\Delta\}} \left(\sum_{i=0}^{k} \sum_{j=0}^{k-i} u_x^i u_y^j T_{i,j} \binom{k}{i} \cdot \binom{k-i}{j} (-t_x u_x/2 - t_y u_y/2)^{k-i-j} \right)^{1/k}$$

We now prove that the above is a a good approximation to the directional width. First we note that the directional width $W_u(S)$ is equal to:

$$W_u = 2 \min_{t_x,t_y \in \{-2\Delta,\ldots,2\Delta\}} \max_{(x,y)\in S} (u_x(x - t_x/2) + u_y(y - t_y/2)).$$

Indeed, if $(t_x/2, t_y/2)$ is a point on the mid-line between the two sandwiching lines of S that are perpendicular to u, then $(u_x(x - t_x/2) + u_y(y - t_y/2))$ is the (directional) distance to point (x, y) from the midline.

We now approximate the max-norm in the above definition of W_u by a k-norm for some even k. Consider the approximation

$$\hat{w} = 2 \min_{t_x,t_y \in \{-2\Delta,\ldots,2\Delta\}} \left(\sum_{(x,y)\in S} (u_x(x - t_x/2) + u_y(y - t_y/2))^k \right)^{1/k}.$$

Since W_u has n terms, standard internorm relation implies that $W_u(S) \leq \hat{w} \leq n^{1/k} W_u(S) \leq (1+\epsilon) W_u(S)$. Finally, one can observe that the expansion of $\hat{w}$ gives precisely the same expression as w, i.e., $w = \hat{w}$ is also a $(1+\epsilon)$-approximation to W_u. □

We now show how to obtain a faster randomized algorithm. First of all, we augment the sketch by a sample point a from S at the end of the stream, by implementing the dynamic sampling data structure of [Frahling et al. 2005], in $\mathrm{poly}(\log n\Delta)$ space. We modify the width-estimation algorithm as follows. Intuitively, there are two steps. First, we obtain a 2 approximation of the directional width by observing that, for any point $a \in S$, the minimum slab containing S and whose central line goes through a is a 2 approximation of the directional width. Second, we achieve a $1+\epsilon$ approximation to the true width by trying all shifts of the central line, at steps proportional to the estimation obtained in the first step.

LEMMA 2.3. *There is a randomized algorithm using space* $\mathrm{poly}(\log n\Delta, 1/\epsilon)$ *that computes a* $1+\epsilon$ *approximation of the directional width* $W_u(S)$ *for any* $u = (u_x, u_y)$ *given at the end of the stream, with* $2/3$ *success probability. The runtime of the estimation algorithm is also* $\mathrm{poly}(\log n\Delta, 1/\epsilon)$.

PROOF. Our sketch maintains counters $T_{i,j}$, as well as a sample point a from S, using dynamic sampling algorithm of [Frahling et al. 2005]. Each component uses space $\mathrm{poly}(\log n\Delta, 1/\epsilon)$. The estimation algorithm computes estimates w and w' defined below. w' is the output of the estimation algorithm.

Let

$$w = \left(\sum_{i=0}^{k} \sum_{j=0}^{k-i} u_x^i u_y^j T_{i,j} \binom{k}{i} \binom{k-i}{j} (-a \cdot u)^{k-i-j} \right)^{1/k}$$

We now show w is a $2 + 2\epsilon$ approximation of $W_u(S)$. Notice that $w = \left(\sum_{b\in S} (b \cdot u - a \cdot u)^k\right)^{1/k}$. Thus,

$$\begin{aligned}W_u(S)/2 \leq& \max_{b\in S} |b\cdot u - a\cdot u|\\ \leq& w\\ \leq&(1+\epsilon)\max_{b\in S}|b\cdot u - a\cdot u|\\ \leq&(1+\epsilon)W_u(S)\end{aligned}$$

Next, define $w' = 2\min_{t\in\{-6/\epsilon,\ldots,6/\epsilon\}} f(t)$, where

$$f(t) = \left(\sum_{i=0}^{k}\sum_{j=0}^{k-i} u_x^i u_y^j T_{i,j}\binom{k}{i}\cdot\binom{k-i}{j}(t\epsilon w/3 - a\cdot u)^{k-i-j}\right)^{1/k}$$

We now argue that w' is a $1+\epsilon$ approximation to $W_u(S)$. Let $z^\star = \arg\min_z \max_{b\in S}|z - b\cdot u|$. Note that $\max_{b\in S}|z^\star - b\cdot u| = W_u(S)/2$. Therefore, $|z^\star - a\cdot u| \leq W_u(S)/2$. Thus, there exists $t^\star \in \{-6/\epsilon,\ldots,6/\epsilon\}$ such that $|z^\star + t^\star\epsilon w/3 - a\cdot u| \leq \epsilon w/6 \leq \epsilon W_u/3$. First it is clear that

$$w' \geq 2\min_t \max_{b\in S}|b\cdot u + t\epsilon w/3 - a\cdot u| \geq W_u(S)$$

Next we have

$$\begin{aligned}w' &\leq 2f(t^\star)\\ &= 2\left(\sum_{b\in S}(b\cdot u + t^\star\epsilon w/3 - a\cdot u)^k\right)^{1/k}\\ &\leq 2\left(\sum_{b\in S}(|b\cdot u - z^\star| + |z^\star + t^\star\epsilon w/3 - a\cdot u|)^k\right)^{1/k}\\ &\leq 2(n((1/2+\epsilon/3)W_u(S))^k)^{1/k}\\ &\leq (1+\epsilon)W_u(S).\end{aligned}$$

□

2.2. Width estimation algorithm

First we notice that we can already obtain a deterministic algorithm by running $O(\Delta^2)$ directional width queries (Lemma 2.2) for all possible directions in $[\Delta]^2$. In this section, we show a more efficient algorithm at the expense of randomization. The algorithm from this section calls the oracle from Lemma 2.3 for only $O(1/\epsilon^2)$ potential unit vectors u.

The algorithm uses the following subroutine that allows for sampling "sufficiently far" points in a specified direction. Intuitively, the algorithm first tries to guess the right scale of the width in the given direction. Given this guess, it divides the space into slabs of width equal to an ϵ fraction of the guess. Now, if the guess is correct, any points from the first slab and the last slab intersecting S can serve as a pair of approximately farthest points in the given direction. See Fig. 1 for a pictorial description of the lemma.

LEMMA 2.4. *Fix a unit vector u at the beginning of a stream of updates to a set S. There is a randomized algorithm using $\mathrm{poly}(\log n, \log\Delta, 1/\epsilon)$ space that, at the end of the stream, finds two points $a, b \in S$ such that $u\cdot(a-b) \geq (1-O(\epsilon))W_u(S)$ with 0.9 success probability.*

PROOF. The algorithm is as follows.

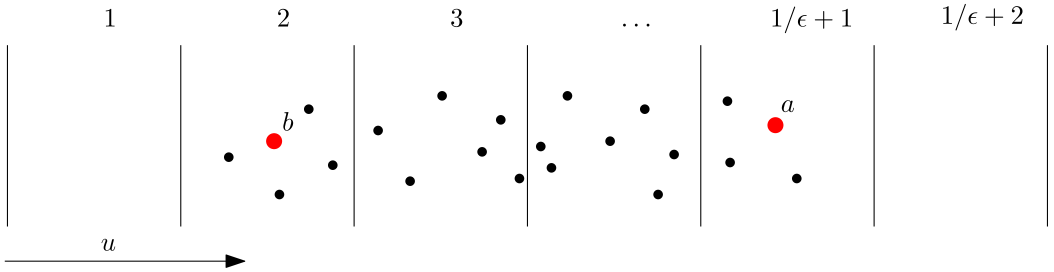

Fig. 1. Sampling “sufficiently far” points. At the right scale, the point set should occupy $1/\epsilon$ consecutive slabs (it is not necessarily the case that there is a point in every slab in that range) and any sample points from the first and last slabs are far from each other in the direction of u.

- Let $m = \lceil \log_{1+\epsilon}(\Delta + 3) \rceil$. For each $c \in \{0, (1+\epsilon)^{-3m-2/\epsilon}, (1+\epsilon)^{-3m-2/\epsilon+1}, \dots, (1+\epsilon)^{m+1/\epsilon}\}$ (except for 0, the other values form a sequence growing exponentially in $1+\epsilon$), perform the following steps.
  - Divide the space in the direction of u into slabs such that the ith slab consists of points p with $u \cdot p \in [i\epsilon c, (i+1)\epsilon c]$.
  - For each $j \in \{0, \dots, 3/\epsilon - 1\}$, let $S_j \subset S$ be the set of points in the slabs whose index $i = j \pmod{3/\epsilon}$. For each j, take a sample point from S_j (if it is not empty) using dynamic sampling of [Frahling et al. 2005].
- In the set T of sample points, choose $a = \arg\max_{p \in T} p \cdot u$ and $b = \arg\min_{p \in T} p \cdot u$.

Now we prove the correctness of the algorithm. By lemma 1.2, the width is at least $((\Delta + 3)\sqrt{3})^{-3}$ and at most 2Δ. Thus, there exists some value of c considered by the algorithm such that $W_u(S) \le c \le (1+\epsilon)W_u(S)$. Let a_1 be the sample point from the set of slabs containing $a^* = \arg\max_{p \in S} p \cdot u$, and b_1 be the sample point from the set of slabs containing $b^* = \arg\min_{p \in S} p \cdot u$. Because a_1 and a^* are in the same set of slabs, they are either in the same slab or in slabs of indices at least $3/\epsilon$ apart. Since $W_u(S) \le c \le (1+\epsilon)W_u(S)$, we have $|(a_1 - a^*) \cdot u| \le c < (3/\epsilon - 1)\epsilon c$. Thus, a_1 and a^* must be in the same slab. Similarly, b_1 and b^* must be in the same slab. We have $|(a_1 - a^*) \cdot u| \le \epsilon c$ and $|(b_1 - b^*) \cdot u| \le \epsilon c$ so $|(a_1 - b_1) \cdot u| \ge (1 - O(\epsilon))|(a^* - b^*) \cdot u| = (1 - O(\epsilon))W_u(S)$. □

(1) Let $\vec{i}, \vec{j}$ be the standard orthonormal basis of the plane. For each $l \in \{1, 2, \dots, \frac{4\pi}{\epsilon}\}$, let $u_l = \cos(l\epsilon/2)\vec{i} + \sin(l\epsilon/2)\vec{j}$. Find two points $a_l, b_l \in S$, such that $W_{u_l}(S) \le (1+\epsilon)(a_l - b_l) \cdot u_l$, using Lemma 2.4.

(2) Let $v_l^\perp = \frac{a_l - b_l}{||a_l - b_l||}$ and v_l be the unit vector orthogonal to $a_l - b_l$. Compute the approximation W_l of $W_{v_l}(S)$, using Lemma 2.3.

(3) For each integer $t \in \{-3/\epsilon, \dots, 3/\epsilon\}$, let $y_{l,t} = \frac{t \cdot \epsilon W_l}{3||a_l - b_l||}$, $y_{l,6/\epsilon+1+t} = y_{l,t}$, $x_{l,t} = \sqrt{1 - y_{l,t}^2}$, $x_{l,6/\epsilon+1+t} = -x_{l,t}$ and for each integer $t \in \{-3/\epsilon, \dots, 9/\epsilon + 1\}$, let $x_{l,12/\epsilon+2+t} = y_{l,t}$, $y_{l,12/\epsilon+2+t} = x_{l,t}$. For any l, t such that $x_{l,t}$ and $y_{l,t}$ are reals, let $v_{l,t} = x_{l,t}v_l + y_{l,t}v_l^\perp$. Compute an approximation of the directional width with respect to $v_{l,t}$ for all $l \in \{1, 2, \dots, \frac{4\pi}{\epsilon}\}, t \in \{-3/\epsilon, \dots 21/\epsilon + 3\}$, using Lemma 2.3. Return the minimum directional width over all l, t as an estimate for $W(S)$.

Fig. 2. The randomized algorithm for approximating width. It uses the algorithm for approximating the directional width from the previous section in a black-box fashion.

Next, we show the full algorithm described in Fig. 2 yields a good approximation to the width $W(S)$. The idea of algorithm is as follows. First it tries to guess the direction minimizing the width. If the instance is fat i.e. the width is not too small compared with the diameter, a small absolute error in the guess, say, ϵ degrees, is sufficient to get an approximation for the width. If the width is much smaller than the diameter, this is not sufficient. However, in this case, the direction minimizing width is approximately orthogonal to the direction maximizing width so we can get a good guess by finding far points and using the direction orthogonal to the line connecting those points as a guess. Next, once the algorithm has a reasonable approximation of the width, it can tweak the current guess with steps of proportionate magnitude to get a $1 + \epsilon$ approximation. See Fig. 3 for a pictorial description of the lemma.

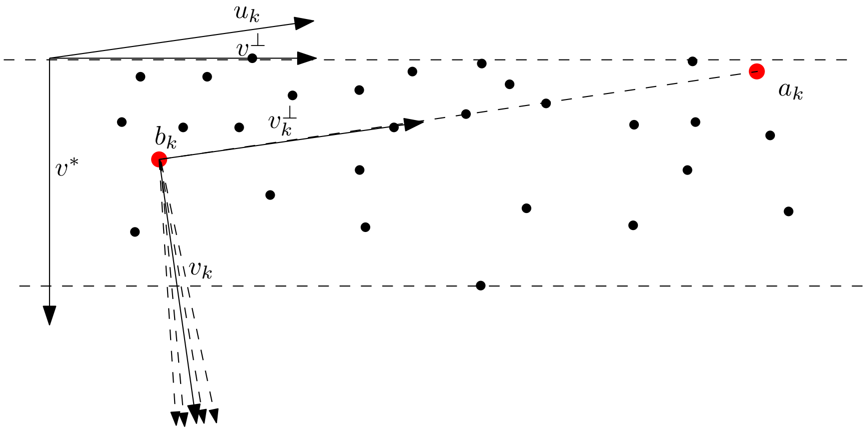

Fig. 3. Visualization of the algorithm for approximating width. v_k is a crude approximation for the optimal direction v^*. A finer approximation is obtained by searching around v_k.

LEMMA 2.5. *The minimum directional width over the directions $v_{l,t}$ is a $1 + O(\epsilon)$ approximation of $W(S)$, with $2/3$ success probability.*

PROOF. Let $v^* = \arg\min_{||v||=1} W_v(S)$. Let $v^\perp$ be the unit vector orthogonal to v^*. By the definition of u_l's, there exists some k such that the angle between u_k and $v^\perp$ is at most $\epsilon/2$. Let γ and δ be numbers satisfying $u_k = \gamma v^* + \delta v^\perp$. Notice that $|\gamma| = |u_k \cdot v^*| \le \epsilon$ and $|\delta| = |u_k \cdot v^\perp| \ge 1 - \epsilon^2$. Let α and β be numbers satisfying $v_k^\perp = \alpha v^* + \beta v^\perp$ (hence $v_k = \beta v^* - \alpha v^\perp$). Notice that $|\alpha|, |\beta| \le 1$ and $|(a_k - b_k) \cdot v^*| = |\alpha| \cdot ||a_k - b_k|| \le W(S)$.

First we show W_k is a $2 + O(\epsilon)$ approximation of the width $W(S)$. Consider two arbitrary points $p, q \in S$. Because $(a_k - b_k) \cdot u_k$ is a $1 \pm \epsilon$ approximation of $W_{u_k}(S)$, we have $|(p - q) \cdot u_k| \le (1 + \epsilon)|(a_k - b_k) \cdot u_k|$. Substituting u_k and $a_k - b_k$, we get

$$|\gamma(p - q) \cdot v^* + \delta(p - q) \cdot v^\perp| \le (1 + \epsilon)||a_k - b_k|| \cdot |\alpha\gamma + \beta\delta|$$

Thus,

$$|\delta(p - q) \cdot v^\perp| \le |\gamma(p - q) \cdot v^*| + (1 + \epsilon)||a_k - b_k|| \cdot |\alpha\gamma + \beta\delta|$$

We can now bound the distance between p and q in the direction v_k.

$$\begin{aligned} |(p-q)\cdot v_k| =& |(p-q)\cdot(\beta v^* - \alpha v^{\perp})| \\ \leq& |\beta(p-q)\cdot v^*| + |\alpha(p-q)\cdot v^{\perp}| \\ \leq& |\beta(p-q)\cdot v^*| + |\tfrac{\alpha}{\delta}|\left(|\gamma(p-q)\cdot v^*| + (1+\epsilon)||a_k - b_k|| \cdot |\alpha\gamma + \beta\delta|\right) \qquad (1) \\ \leq& |\beta| W(S) + |\tfrac{\alpha}{\delta}| \left(|\gamma| W(S) + (1+\epsilon)\frac{W(S)}{|\alpha|}\right) \\ \leq& (2+O(\epsilon))W(S) \end{aligned}$$

Thus, the directional width with respect to $v_k = \beta v^* - \alpha v^{\perp}$ is at most $(2+O(\epsilon))W(S)$.

Now we show that one of the directions $v_{l,t}$ gives a much finer $1+O(\epsilon)$ approximation to the width. We consider two cases.

— $|\alpha| \cdot ||a_k - b_k|| \leq \epsilon W(S)$. Intuitively, this is the case where the width is roughly the diameter i.e. the point set is "fat". Substituting the aforementioned condition into (1), we get that for any $p, q \in S$,

$$\begin{aligned} |(p-q)\cdot v_k| &\leq |\beta(p-q)\cdot v^*| + |\tfrac{\alpha}{\delta}| \left(|\gamma| W(S) + (1+\epsilon)\tfrac{\epsilon W(S)}{|\alpha|} \cdot |\alpha\gamma + \beta\delta| \right) \\ &\leq (1+O(\epsilon))W(S). \end{aligned}$$

Thus, W_k is already a $1+O(\epsilon)$ approximation of $W(S)$.

— $|\alpha| \cdot ||a_k - b_k|| > \epsilon W(S)$. Intuitively, this is the case where the width is much smaller than the diameter. By the definition of $y_{l,t}$'s, there exists some h such that $|y_{k,h} - \alpha| \leq \frac{\epsilon W(S)}{||a_k - b_k||}$ and $|x_{k,h} - \beta| \leq \frac{\epsilon W(S)}{||a_k - b_k||}$. Consider two arbitrary points $p, q \in S$. We have

$$\begin{aligned} |(p-q)\cdot v_{k,h}| &\leq |(\beta x_{k,h} + \alpha y_{k,h})(p-q)\cdot v^*| + |(\beta y_{k,h} - \alpha x_{k,h})(p-q)\cdot v^{\perp}| \\ &\leq |(p-q)\cdot v^*| + |(\beta y_{k,h} - \alpha x_{k,h})(p-q)\cdot v^{\perp}| \\ &\leq W(S) + |(\beta y_{k,h} - x_{k,h} y_{k,h})(p-q)\cdot v^{\perp}| + |(x_{k,h} y_{k,h} - x_{k,h}\alpha)(p-q)\cdot v^{\perp}| \\ &\leq W(S) + \frac{2\epsilon W(S)}{||a_k - b_k||} \cdot \tfrac{1}{\delta}\left(|\gamma(p-q)\cdot v^*| + (1+\epsilon)||a_k - b_k|| \cdot |\alpha\gamma + \beta\delta|\right) \\ &\leq W(S) + |\tfrac{2\alpha\gamma}{\delta}| \cdot W(S) + \frac{2\epsilon(1+\epsilon)W(S)}{|\delta|} \\ &\leq (1+O(\epsilon))W(S). \end{aligned}$$

Thus, the minimum directional width with respect to $v_{l,t}$'s is a $1+O(\epsilon)$ approximation of $W(S)$. Finally, if we perform the above algorithm with a ϵ replaced by ϵ/c for suitable large constant c, we obtain the desired $1+\epsilon$ approximation. □

In conclusion, our main Theorem 2.1 follows from Lemmas 2.3, 2.4, 2.5. We note that although the lemmas give constant probability of success, one can amplify the success rate in a standard way and hence apply them the desired number of times.

2.3. Deterministic algorithm for any dimension

In this section, we consider the generalization where points come from a d-dimensional grid $[\Delta]^d$. Our algorithm is also deterministic, albeit relies on the polynomial system solver of [Basu et al. 1996].

THEOREM 2.6. *There is a deterministic algorithm running in time $(\log\Delta)(\frac{d \log n}{\epsilon})^{O(d)}$ that computes a $1+\epsilon$ approximation of the width $W(S)$.*

PROOF. We use a generalization of the sketch used in Lemma 2.2. Again use $k = \Theta(\frac{\log n}{\log(1+\epsilon)})$ and k is even. The sketch consists of counters

$$T_{\{c_i\}_{i=1}^d} = \sum_{a\in S}\prod_{i=1}^{d}(a_i)^{c_i}$$

for all $c_i \in \{0, \ldots, k\}$. For $t, u \in \mathbb{R}^d$, define

$$f(t,u) = \sum_{a\in S}(u\cdot(a-t))^k$$

Similar to the proof of Lemma 2.2, since there are at most n points in S, we have $W_u(S) \le \min_t f(t,u)^{1/k} \le n^{1/k}W_u(S) \le (1+\epsilon)W_u(S)$. Furthermore, $f(t,u)$ can be computed from the counters $T_{\{c_i\}}$.

To distinguish between the case where the width is at least D and the case where the width is at most $D/(1+\epsilon)$, we have to determine if the system of two polynomial equations: $||u||_2^2 = 1$ and $f(t,u) \le D^k$ has any root. This system has degree $O(d\log n/\epsilon)$ and $O(d)$ variables. By the algorithm of [Basu et al. 1996], this task can be done in $(d\log n/\epsilon)^{O(d)}$ time. By binary search and Lemma 1.2, the algorithm for approximating width runs in time $(\log\Delta)(d\log n/\epsilon)^{O(d)}$. □

REFERENCES

AGARWAL, P., HAR-PELED, S., AND VARADARAJAN, K. 2004. Approximating extent measures of points. *J. of ACM 51,* 4.

AGARWAL, P. K., HAR-PELED, S., AND VARADARAJAN, K. R. 2005. Geometric approximation via coresets - survey. *Combinatorial and Computational Geometry (MSRI publication) 52*.

BASU, S., POLLACK, R., AND ROY, M.-F. 1996. On the combinatorial and algebraic complexity of quantifier elimination. *J. ACM 43*, 1002–1045.

CHAN, T. M. 2006. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom. 35,* 1-2, 20–35.

CHAN, T. M. 2009. Dynamic coresets. *Discrete Comput Geom 42*, 469–488. Previously in SoCG'08.

CHAN, T. M. AND SADJAD, B. S. 2006. Geometric optimization problems over sliding windows. *Int. J. Comput. Geometry Appl. 16,* 2-3, 145–158.

FEIGENBAUM, J., KANNAN, S., AND ZHANG, J. 2004. Computing diameter in the streaming and sliding-window models. *Algorithmica 41,* 1, 25–41.

FRAHLING, G., INDYK, P., AND SOHLER, C. 2005. Sampling in dynamic data streams and applications. In *Proceedings of the Symposium on Computational Geometry (SoCG)*. 142–149.

FRAHLING, G. AND SOHLER, C. 2005. Coresets in dynamic geometric data streams. In *Proceedings of the Symposium on Theory of Computing (STOC)*. 209–217.

INDYK, P. 2004. Algorithms for dynamic geometric problems over data streams. In *Proceedings of the Symposium on Theory of Computing (STOC)*.

INDYK, P., LEWENSTEIN, M., LIPSKY, O., AND PORAT, E. 2004. Closest pair problems in very high dimensions. In *Proceedings of International Colloquium on Automata, Languages and Programming (ICALP)*.

INDYK, P., MCGREGOR, A., NEWMAN, I., AND ONAK, K. 2011. Open problems in data streams, property testing, and related topics. Available at `http://www.cs.umass.edu/~mcgregor/papers/11-openproblems.pdf`.

MUTHUKRISHNAN, M. 2005. *Data Streams: Algorithms and Aplications*. Foundations and Trends in Theoretical Computer Science. Now Publishers Inc.

VARADARAJAN, K. AND XIAO, X. 2012. A near-linear algorithm for projective clustering integer points. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 1329–1342.

Received April 2012; revised December 2012; accepted MM YYYY