Width of Points in the Streaming Model

Alexandr Andoni* Huy L. Nguyen†

Abstract

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 lines 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 [AHPV04, Cha06], or a sliding window [CS06].

This solves an open question from the "2009 Kanpur list" of Open Problems in Data Streams, Property Testing, and Related Topics [IMNO11].

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, does it look like 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 [AHPV05] who give a good overview of the problem.

In the quest for very efficient algorithms for these problems, researchers developed efficient $1+\epsilon$ approximation algorithms [AHPV04, Cha06] in the *streaming model* [Mut05]. 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 algorithms for a range of geometric optimization problems (see the survey [AHPV05] and references therein, also [Cha06, CS06]).

At the same time, it seems challenging to adapt the 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). [Ind04] gave some of the first low-space dynamic algorithms for geometric problems. Presently we have efficient dynamic algorithms for some geometric problems, including the diameter [FKZ04, Ind04], or the clustering and optimization problems [FIS05, FS05]. However the width problem has so far remained open (see Question 17 in the open list [IMNO11]).

Here we give an efficient $1+\epsilon$ approximation algorithm for computing the width in the dynamic streaming model.

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. To construct this oracle, we show that this quantity may be approximated by a polynomial (in the coordinates of the pointset and the parameters of the line), which has a sufficiently low degree. The polynomial arises from the following standard relation between norms (see, e.g., [ILLP04]): one can approximate the max-norm $\|x\|_\infty = \max_i |x_i|$ by a sufficiently high p-norm $\|x\|_p = (\sum_i |x_i|^p)^{1/p}$. In our setting, x_i's are the distance from the line to a point i, in which case we can set $p \approx O(\epsilon^{-1} \log n)$. Furthermore, we observe that a small number of moments of the pointset will suffice for evaluating the resulting polynomial for a given input line. Finally, once we have such an oracle, it is possible to just enumerate over all possible lines and thus find the (approximately) best sandwiching lines. We also show a randomized algorithm achieving a better runtime.

1.1 Preliminaries

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

*Microsoft Research. Work done in part while the author was a postdoctoral researcher at Princeton University/CCI, supported by NSF CCF 0832797.

†Princeton University. Supported by NSF CCF 0832797 and a Gordon Wu fellowship.

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 $\{1, 2, \ldots \Delta\}^2$, and n is an upper bound on the size of S.

2 The Algorithm

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, with a caveat that the evaluation time (at the end of the stream) is polynomial in Δ.

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 2.1. *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. Let $w = 2 \min_{t_x, t_y \in \{-2\Delta, \ldots, 2\Delta\}} (\sum_{(x,y) \in S} (u_x(x - t_x/2) + u_y(y - t_y/2))^k)^{1/k}$. We argue that w is a good approximation of $W_u(S)$. Indeed since there are at most n points in S, we have that $W_u(S) \leq w \leq n^{1/k} W_u(S) \leq (1 + \epsilon) W_u(S)$.

We now observe that we can compute w from $T_{i,j}$'s and u:

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

We show how to obtain a faster randomized algorithm. To achieve this, 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 [FIS05], in $\mathrm{poly}(\log n\Delta)$ space. Intuitively, the algorithm works in two steps. First, observe 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. Next, the true width can be approximated up to a factor of $1 + \epsilon$ by trying all the shifts of the central line at steps proportional to the estimation obtained in the first step.

LEMMA 2.2. *There is a randomized algorithm running in time* $\mathrm{poly}(\log n\Delta, 1/\epsilon)$ *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.*

Proof. Our sketch maintains counters $T_{i,j}$, and sampling a point a from S using dynamic sampling by [FIS05]. The estimation algorithm computes estimates w and w' defined below.

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

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

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 Approximating the width of a point set First we notice that we can already obtain a deterministic algorithm by running $O(\Delta^2)$ directional width queries (Lemma 2.1) 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 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 2.3. *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.*

The full algorithm uses the above lemma and is described in Fig. 2.

Proof. [Proof of Lemma 2.3] The algorithm is as follows.

- Let $m = \lfloor \log_{1+\epsilon}\Delta \rfloor$. For each $c \in \{0, (1+\epsilon)^{-m}, (1+\epsilon)^{-m+1}, \ldots, (1+\epsilon)^m\}$, perform the following steps.
  - Divide the space 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 $j \in \{0,\ldots,3/\epsilon - 1\}$, 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 by [FIS05].
- 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. There exists some value of c considered by the algorithm such that $W_u(S) \leq c \leq (1+\epsilon)W_u(S)$. 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 $b^* = \arg\min_{p\in S} p\cdot u$. Since $W_u(S) \leq c \leq (1+\epsilon)W_u(S)$, a_1 and a^* must be in the same slab and b_1 and b^* must be in the same slab. We have $|(a_1 - a^*)\cdot u| \leq \epsilon c$ and $|(b_1 - b^*)\cdot u| \leq \epsilon c$ so $|(a_1 - b_1)\cdot u| \geq (1-O(\epsilon))|(a^* - b^*)\cdot u| = (1-O(\epsilon))W_u(S)$.

Now we show that the algorithm from Fig. 2 yields a good approximation to the width $W(S)$. Intuitively, the idea of algorithm is as follows. First the algorithm 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

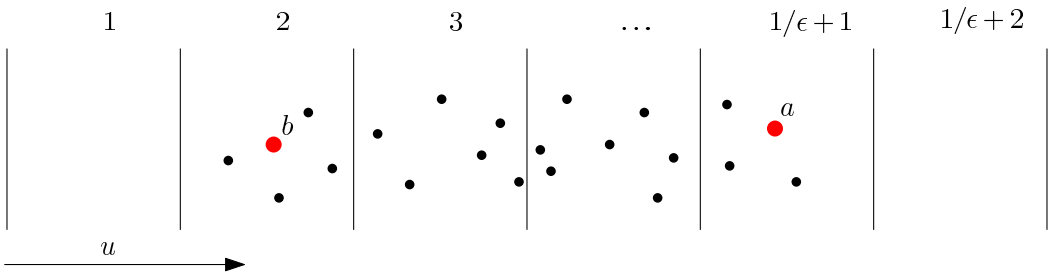

Figure 1: Sampling "sufficiently far" points. At the right scale, the point set should occupy $1/\epsilon$ consecutive slabs and any sample points from the first and last slabs are far from each other in the direction of u.

1. Let $\vec{i},\vec{j}$ be the standard orthonormal basis of the plane. For each $i \in \{1,\ldots,\frac{2\pi}{\epsilon}\}$, let $u_i = \cos(i\epsilon)\vec{i} + \sin(i\epsilon)\vec{j}$. Find two points $a_i, b_i \in S$, such that $W_{u_i}(S) \leq (1+\epsilon)(a_i - b_i)\cdot u_i$, using Lemma 2.3.
2. Let $v_i^\perp = \frac{a_i - b_i}{||a_i - b_i||}$ and v_i be the unit vector orthogonal to $a_i - b_i$. Compute the approximation W_i of $W_{v_i}(S)$, using Lemma 2.2.
3. For each integer $j \in \{-3/\epsilon,\ldots,3/\epsilon\}$, let $y_{i,j} = \frac{j\cdot\epsilon W_i}{3||a_i-b_i||}$, $y_{i,6/\epsilon+1+j} = y_{i,j}$, $x_{i,j} = \sqrt{1-y_{i,j}^2}$, $x_{i,6/\epsilon+1+j} = -x_{i,j}$ and for each integer $j \in \{-3/\epsilon,\ldots,9/\epsilon+1\}$, let $x_{i,12/\epsilon+2+j} = y_{i,j}$, $y_{i,12/\epsilon+1+j} = x_{i,j}$. For any i,j such that $x_{i,j}$ and $y_{i,j}$ are reals, let $v_{i,j} = x_{i,j}v_i + y_{i,j}v_i^\perp$. Compute an approximation of the directional width with respect to $v_{i,j}$ for all $i \in \{0,\ldots,\frac{2\pi}{\epsilon}\}, j \in \{-3/\epsilon,\ldots 21/\epsilon+2\}$, using Lemma 2.2. Return the minimum directional width over all i,j as an estimate for $W(S)$.

Figure 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.

guess with steps of proportionate magnitude to get a $1+\epsilon$ approximation. See Fig. 3 for a pictorial description of the lemma.

LEMMA 2.4. *The minimum directional width over the directions $v_{i,j}$ 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_i'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 $|\gamma| = |u_k \cdot v^*| \leq \epsilon$ and $|\delta| = |u_k \cdot v^\perp| \geq 1-\epsilon^2$. 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 $|\alpha|, |\beta| \leq 1$ and $|(a_k - b_k)\cdot v^*| = |\alpha|\cdot||a_k - b_k|| \leq W(S)$.

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 $|(p-q)\cdot u_k| \leq (1+\epsilon)|(a_k-b_k)\cdot u_k|$. Substituting u_k and $a_k - b_k$, we get

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

Thus,

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

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

$$\begin{aligned}
|(p-q)\cdot v_k| =& |(p-q)\cdot(\beta v^* - \alpha v^\perp)| \\
\leq& |\beta(p-q)\cdot v^*| + |\alpha(p-q)\cdot v^\perp| \\
\leq& |\beta(p-q)\cdot v^*| + |\tfrac{\alpha}{\delta}|\,(|\gamma(p-q)\cdot v^*| + \\
&+(1+\epsilon)||a_k - b_k||\cdot|\alpha\gamma + \beta\delta|) \qquad (2.1)\\
\leq& |\beta| W(S) + \\
&+ |\tfrac{\alpha}{\delta}|\left(|\gamma| W(S) + (1+\epsilon)\frac{W(S)}{|\alpha|}\right) \\
\leq& (2+O(\epsilon))W(S)
\end{aligned}$$

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

Now to show that the the directions $v_{i,j}$ give a much finer approximation, we consider two cases.

- $|\alpha|\cdot||a_k - b_k|| \leq \epsilon W(S)$. Substituting into (2.1),

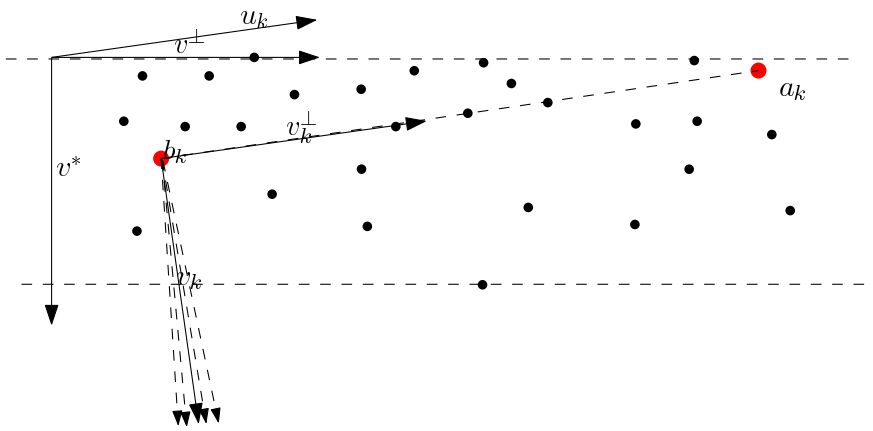

Figure 3: Visualization of the algorithm for approximating width. v_k is a crude approximation for the optimal direction v^*. A finer approximation is obtained by searching around v_k.

we get that for any $p, q \in S$,

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

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

- $|\alpha| \cdot ||a_k - b_k|| > \epsilon W(S)$. By the definition of $y_{i,j}$'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

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

Thus, the minimum directional width with respect to $v_{i,j}$'s is a $1+O(\epsilon)$ approximation of $W(S)$.

In conclusion, our main Theorem 2.1 follows from Lemmas 2.2, 2.3, 2.4. 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.

REMARK 2.1. *We remark that the algorithm can be generalized to higher dimensions, with parameters that depend exponentially on the number of dimensions.*

References

[AHPV04] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Approximating extent measures of points. *J. of ACM*, 51(4), 2004.

[AHPV05] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via coresets - survey. *Combinatorial and Computational Geometry (MSRI publication)*, 52, 2005.

[Cha06] Timothy M. Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom.*, 35(1-2):20–35, 2006.

[CS06] Timothy M. Chan and Bashir S. Sadjad. Geometric optimization problems over sliding windows. *Int. J. Comput. Geometry Appl.*, 16(2-3):145–158, 2006.

[FIS05] Gereon Frahling, Piotr Indyk, and Christian Sohler. Sampling in dynamic data streams and applications. In *Proceedings of the twenty-first annual symposium on Computational geometry (SoCG)*, pages 142–149, 2005.

[FKZ04] Joan Feigenbaum, Sampath Kannan, and Jian Zhang. Computing diameter in the streaming and sliding-window models. *Algorithmica*, 41(1):25–41, 2004.

[FS05] Gereon Frahling and Christian Sohler. Coresets in dynamic geometric data streams. In *STOC*, pages 209–217, 2005.

[ILLP04] Piotr Indyk, Moshe Lewenstein, Ohad Lipsky, and Ely Porat. Closest pair problems in very high dimensions. In *Proceedings of International Colloquium on Automata, Languages and Programming (ICALP)*, 2004.

[IMNO11] Piotr Indyk, Andrew McGregor, Ilan Newman, and Krzysztof Onak. Open problems in data streams, property testing, and related topics. Available at `http://www.cs.umass.edu/~mcgregor/papers/11-openproblems.pdf`, June 2011.

[Ind04] Piotr Indyk. Algorithms for dynamic geometric problems over data streams. In *Proceedings of the Symposium on Theory of Computing (STOC)*, 2004.

[Mut05] Muthu Muthukrishnan. *Data Streams: Algorithms and Aplications*. Foundations and Trends in Theoretical Computer Science. Now Publishers Inc, January 2005.