A Simple Streaming Algorithm for Minimum Enclosing Balls

Hamid Zarrabi-Zadeh* Timothy M. Chan†

Abstract

We analyze an extremely simple approximation algorithm for computing the minimum enclosing ball (or the 1-center) of a set of points in high dimensions. We prove that this algorithm computes a 3/2-factor approximation in any dimension using minimum space in just one pass over the data points.

1 Introduction

Given a set P of n points in d dimensions, we consider the problem of finding the smallest ball containing P. This is one of the most fundamental and well-known problems in computational geometry, having many applications.

The problem under consideration is an LP-type problem with combinatorial dimension $d+1$ [11] and can thus be solved exactly in $O(d^{O(d)}n)$ deterministic time [5], or $O(d^2n+2^{O(\sqrt{d\log d})})$ expected time [6, 9]. In particular, when d is fixed, the running time is linear.

In this paper, we focus on the case when d may be large. We are interested in faster *approximation* algorithms that avoid the exponential (or superpolynomial) dependency on the dimension. Furthermore, we are interested in algorithms that operate under the *data stream* model, where only one pass over the input is allowed and the algorithm has only a limited amount of working storage. (We assume here that one unit of space can hold one coordinate of a point.) This one-pass streaming model is attractive both in theory and in practice due to emerging applications involving massive data sets, since the entire input need not be stored and can be processed as elements arrive one at a time. For example, see Muthukrishnan's survey [10] on the growing literature on streaming algorithms.

In high dimensions, Bădoiu and Clarkson [2] (following Bădoiu *et al.* [3]) have given an elegant algorithm that can compute a $(1+\varepsilon)$-approximation to the minimum enclosing ball in $O(nd/\varepsilon+(1/\varepsilon)^5)$ time. Although this algorithm requires only $O(1/\varepsilon)$ working space, when viewed in the streaming model, it requires more than one pass—specifically, $\lceil 2/\varepsilon \rceil$ passes. In fixed dimensions, there is a simple (one-pass) streaming algorithm that computes a $(1+\varepsilon)$-approximation to the minimum enclosing ball using $O((1/\varepsilon)^{\lfloor d/2 \rfloor})$ space (in $O((1/\varepsilon)^{\lfloor d/2 \rfloor}n)$ time). Namely, the algorithm just keeps track of the extreme points along $O((1/\varepsilon)^{\lfloor d/2 \rfloor})$ directions.

There are also known streaming results for other geometric optimization problems, in both low and high dimensions. For instance, in fixed dimensions, Agarwal *et al.* [1] have given streaming algorithms that compute $(1+\varepsilon)$-approximations to the width, the minimum enclosing box, the minimum-width enclosing annulus, and related measures, all using $O((1/\varepsilon)^{O(d)}\log^{O(d)} n)$ space. Chan [4] has improved the space complexity to $O((1/\varepsilon)^{O(d)})$. In high dimensions, there is a trivial 2-approximation streaming algorithm for the simpler diameter problem, using $O(d)$ space. For this problem, Indyk [8] has improved the approximation factor to any constant $c>\sqrt{2}$, using $O(dn^{1/(c^2-1)}\log n)$ space with high probability. His technique does not seem to yield a result for minimum enclosing ball though; in any case, a "constant" space bound independent of n would be much more desirable than sublinear space. For the minimum enclosing cylinder problem in high dimensions, Chan [4] has given a streaming algorithm with any fixed approximation factor $c>5$ using $O(d)$ space.

For minimum enclosing ball in high-dimensional data streams, the only known result we are aware of is the trivial 2-approximation algorithm that picks an arbitrary input point c as the center of the ball and sets the radius of the ball to be the distance of the farthest point from c. In this paper, we show how to improve this constant factor. We analyze a simple one-pass streaming algorithm for minimum enclosing ball and prove that it achieves approximation factor 3/2. Our algorithm uses $O(d)$ time per point and just $O(d)$ space. In fact, it uses the "minimum" amount of space possible—at any time, it maintains a single ball, and nothing else. Our algorithm is arguably the simplest algorithm that uses the minimum amount of space.

2 The Algorithm

Without further ado, here is our algorithm in its entirety, which returns a ball B enclosing P:

*School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, hzarrabi@uwaterloo.ca.

†School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, tmchan@uwaterloo.ca. Research supported in part by NSERC.

```
1:  $B \leftarrow \emptyset$
2:  **for** each point $p$ in the input stream $P$ **do**
3:      **if** $p$ is outside $B$ **then**
4:          $B \leftarrow$ the smallest ball enclosing $B$ and $p$
```

Despite the utter simplicity of this algorithm, the analysis of its approximation factor seems nonobvious and, to our knowledge, has not been studied before.

An example of the execution of the algorithm on a point set in two dimensions is illustrated in Figure 1.

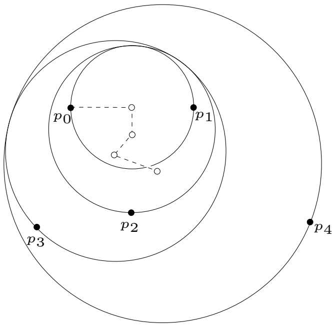

Figure 1: An example.

To aid in the analysis (and implementation), we mention exactly how the coordinates of the ball B can be calculated in line 4. Let p_i be the input point causing the i-th update to B, and let B_i be the value of B after its i-th update. We denote the center point and the radius of B_i by c_i and r_i, respectively.

Initially, we set $r_0 = 0$ and $c_0 = p_0$, where $p_0 \in P$ is the first input point. Letting $\delta_i = \frac{1}{2}(\|p_i - c_{i-1}\| - r_{i-1})$ (i.e., half the distance of p_i to B_{i-1}), we have

$$r_i = r_{i-1} + \delta_i \quad \text{and} \quad \|c_i - c_{i-1}\| = \delta_i.$$

As c_i lies on the line segment from c_{i-1} to p_i, the second equation implies that $c_i = c_{i-1} + \frac{\delta_i}{\|p_i - c_{i-1}\|}(p_i - c_{i-1})$.

Remark. The above equations imply the following interesting property concerning the trajectory of the center points $c_1, \ldots, c_i$ (the path shown in dashed lines in Figure 1): the total length of this trajectory is exactly equal to the last radius r_i.

3 The Analysis

In this section, we prove an upper bound of 3/2 on the approximation factor of our algorithm. The proof is not long but is tricky and involves a clever invariant. We first recall one known geometric fact.

Lemma 1 *If two chords intersect inside a circle, then the product of the segments of one chord equals the product of the segments of the other chord.*

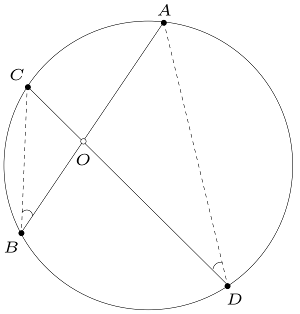

Figure 2: On Lemma 1.

Proof. Let O be the intersection point of the two chords AB and CD. The two triangles $\triangle AOD$ and $\triangle BOC$ are similar. Therefore $\frac{|AO|}{|CO|} = \frac{|DO|}{|BO|}$, and hence $|AO||BO| = |CO||DO|$. □

Theorem 2 *Given a set P of n points in d dimensions, the algorithm in Section 2 computes a 3/2-approximation to the minimum enclosing ball of P in $O(dn)$ time and $O(d)$ space.*

Proof. Let B^* be the minimum enclosing ball of P, of radius r^*. Let p_i, c_i, and r_i be as defined in Section 2. The p_i's lie inside B^*, and so are the c_i's (as one can see easily by induction, due to the convexity of B^*). For each $i > 0$, consider the chord of B^* which passes through c_{i-1} and p_i (which passes through c_i as well). The point c_i splits this chord into two segments, one containing c_{i-1} and the other containing p_i. Let a_i be the length of the former segment and b_i be the length of the latter segment. The key to the whole proof lies in finding the right invariant, which turns out to be the following:

Claim. $r_i^2 < 3a_ib_i$ *for all* $i > 0$.

Proof. We prove by induction on i. The base case follows immediately because $a_1, b_1 \geq r_1$ (since c_1 is the midpoint of $p_0, p_1 \in B^*$).

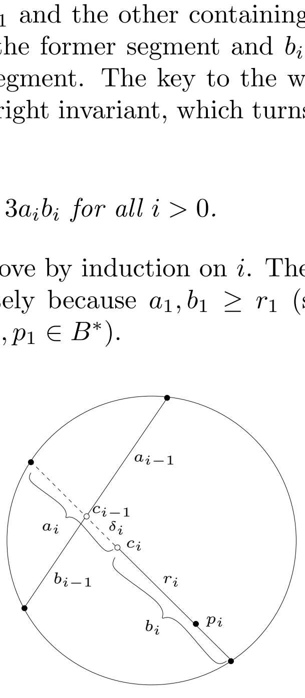

Figure 3: From step $i-1$ to step i.

Now, suppose that $r_{i-1}^2 < 3a_{i-1}b_{i-1}$. By applying Lemma 1 (to the intersection of B^* with the plane through c_{i-1}, p_{i-1}, and p_i), we have

$$a_{i-1}b_{i-1} = (a_i - \delta_i)(b_i + \delta_i). \qquad (1)$$

A chain of algebraic manipulations then (magically) yields the claim:

$$\begin{aligned}
3a_ib_i &= 3\left(\frac{a_{i-1}b_{i-1}}{b_i+\delta_i}+\delta_i\right)b_i \qquad \text{(by (1))}\\
&> \left(\frac{r_{i-1}^2}{b_i+\delta_i}+3\delta_i\right)b_i \qquad \text{(by hypothesis)}\\
&\geq \left(\frac{r_{i-1}^2}{r_i+\delta_i}+3\delta_i\right)r_i\\
&\quad \text{(as } b_i \geq r_i \text{ and } x/(x+\delta_i) \text{ is increas. for } x \geq 0)\\
&= \left(\frac{(r_i-\delta_i)^2}{r_i+\delta_i}+3\delta_i\right)r_i\\
&= \left(\frac{r_i^2+\delta_ir_i+4\delta_i^2}{r_i+\delta_i}\right)r_i\\
&> r_i^2. \qquad \square
\end{aligned}$$

The proof of the theorem is now straightforward. W.l.o.g., assume that $r_i \geq r^*$. Clearly, $a_i + b_i \leq 2r^*$. So,

$$r_i^2 < 3a_ib_i \leq 3(2r^* - b_i)b_i \leq 3(2r^* - r_i)r_i,$$

as $b_i \geq r_i$ and $(2r^* - x)x$ is decreasing for $x \geq r^*$. Thus $r_i < 3(2r^* - r_i)$, which means that $r_i < \frac{3}{2}r^*$ for all $i > 0$. □

4 Lower Bound

In this section we show that the analysis from Section 3 is essentially tight by providing a lower bound example for which our algorithm produces an enclosing ball with a radius close to 3/2 times the radius of the minimum enclosing ball.

Our example composed of n points equally spaced on the boundary of a unit circle. Obviously, the radius of the minimum enclosing ball for this point set is 1. As n goes to infinity, the trajectory path produced by the algorithm approaches a curve (like what is shown in Figure 4 for $n = 6$ and $n = 12$), whose length can be described using some differentiate integral equation. This curve seems interesting in its own right, though we are unable to find any previous work or information about it. The resulting function is quite complicated and solving it seems to be very hard. We have numerically calculated the length of this curve (as a discrete path) for different values of n. The numerical results, presented in Table 1, shows that for large values of n, the length of the path becomes very close to 1.5. (It is a routine exercise to confirm formally that the limiting curve exists, with length converging to precisely 3/2.)

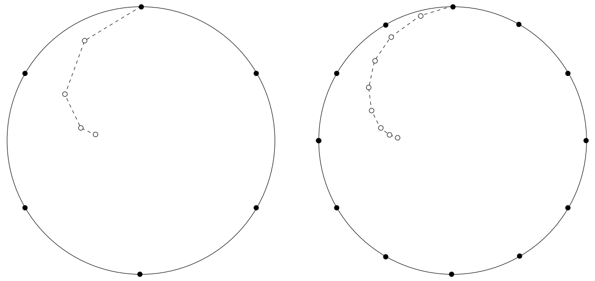

Figure 4: A lower bound example and an interesting curve.

point set size	trajectory length
10	1.39426
100	1.48955
1000	1.49895
10000	1.49989
100000	1.49998

Table 1: The lower bound example.

We can also prove the following lower bound on the approximation factor of any algorithm that uses the same amount of space as our algorithm does.

Theorem 3 *There is a lower bound of* $(1+\sqrt{2})/2 \approx 1.207$ *on the approximation factor of any deterministic algorithm for the minimum enclosing ball problem that at any time stores only one enclosing ball and nothing more.*

Proof. Let $\mathcal{A}$ be such an approximation algorithm. The adversary gives a sequence of points in the plane. Let $p_1 = (0, 1)$ and $p_2 = (0, -1)$ be the first two points provided by the adversary, and let B be the ball produced by $\mathcal{A}$ to enclose $\{p_1, p_2\}$. It is clear that B contains at least one of the points $q_1 = (-1, 0)$ and $q_2 = (1, 0)$. Suppose w.l.o.g. that B contains q_1. The adversary then gives the point $p_3 = (1 + \sqrt{2}, 0)$. Now, $\mathcal{A}$ knows that all points given so far are enclosed by B, but for all points enclosed by B, it cannot remember which ones (in particular, $q_1 \in B$) have been parts of the input. Therefore, the updated ball after receiving p_3 must also enclose q_1. So, the radius of the ball must be at least $\frac{1}{2}||p_3 - q_1|| = \frac{1}{2}(2 + \sqrt{2})$. However, the point set $\{p_1, p_2, p_3\}$ can be optimally enclosed by a ball of radius $\sqrt{2}$. Hence, the approximation factor of $\mathcal{A}$ is at least $\frac{1}{2\sqrt{2}}(2 + \sqrt{2}) = \frac{1}{2}(1 + \sqrt{2})$. □

5 Open Problems

In this paper, we have provided a very simple (and thus highly practical) one-pass algorithm that computes a 3/2-approximation to the minimum enclosing ball of a set of points in any dimension. Our algorithm is simple enough to easily replace the well-known naïve 2-approximation algorithm, which is the only previously known one-pass algorithm working in high dimensions. Our algorithm can be regarded as a solution to a restricted *on-line* version of the minimum enclosing ball problem; Section 3 can be viewed as a *competitive analysis* of an on-line algorithm.

Many open problems remain. For example, is 3/2 the best possible approximation factor among one-pass algorithms that use the "minimum" amount of space (i.e., can Theorem 3 be strengthened)? What is the best approximation factor among one-pass algorithms that use $O(d)$ space, or more generally, $d^{O(1)}$ space? Can we get still better factors for minimum enclosing ball using sublinear ($o(n)$) space, perhaps by adapting Indyk's technique for the diameter problem [8]?

The same questions can also be asked for algorithms with two passes, three passes, and so on. As our work shows, Bădoiu and Clarkson's $\lceil 2/\varepsilon \rceil$-pass algorithm [2] is not necessarily the best possible result. For example, we can consider the following two-pass strategy that invokes our algorithm twice. Let c_f and r_f be the center and the radius of the enclosing ball after terminating the first pass. We then use the same algorithm for the second pass, except that we start with a ball centered at c_f with radius $\frac{3}{5}r_f$. Experimental results seem to suggest that the approximation factor obtained by this two-pass algorithm is at most 1.4, at least for the bad example from Section 4.

Similar questions are open even for the diameter problem. For example, there is a simple two-pass diameter algorithm [7] that gives $\sqrt{3}$ factor with $O(d)$ space. We do not know if this is the best two-pass algorithm with $O(d)$ space, or what is the best three-pass algorithm.

References

[1] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 51(4):606–635, 2004.

[2] M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *Proc. 14th ACM-SIAM Sympos. Discrete Algorithms*, pages 801–802, 2003.

[3] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proc. 34th ACM Sympos. Theory Comput.*, pages 250–257, 2002.

[4] T. M. Chan. Faster core-set constructions and data stream algorithms in fixed dimensions. In *Proc. 20th ACM Sympos. Comput. Geom.*, pages 152–159, 2004. *Comput. Geom. Theory Appl.*, to appear.

[5] B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. *J. Algorithms*, 21(3):579–597, 1996.

[6] K. L. Clarkson. Las Vegas algorithms for linear and integer programming. *J. ACM*, 42(2):488–499, 1995.

[7] O. Eğecioğlu and B. Kalantari. Approximating the diameter of a set of points in the Euclidean space. *Inform. Process. Lett.*, 32:205–211, 1989.

[8] P. Indyk. Better algorithms for high-dimensional proximity problems via asymmetric embeddings. In *Proc. 14th ACM-SIAM Sympos. Discrete Algorithms*, pages 539–545, 2003.

[9] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4-5):498–516, 1996.

[10] S. M. Muthukrishnan. Data streams: Algorithms and applications. `http://athos.rutgers.edu/~muthu/stream-1-1.ps`, 2003.

[11] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In *New Results and New Trends in Computer Science*, LNCS 555, pages 359–370, 1991.