

Chapter 5

Approximate Smallest Enclosing Balls

5.1 Bounding Volumes

A *bounding volume* for a set $S \subseteq \mathbb{R}^d$ is a superset of S with a simple shape, for example a box, a ball, or an ellipsoid.

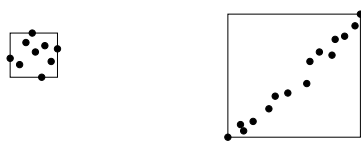


Figure 5.1: Bounding boxes $Q(P)$ of point sets $P \subseteq \mathbb{R}^2$

Bounding volumes which are smallest among the ones of a given shape (with respect to volume, diameter, or some other criteria) are very useful, because they ‘approximate’ the possibly complicated set S by a simple superset. Then, whenever you want to know something about S , you first try to use the bounding volume to answer the question (this is usually cheap), and only if this fails, you analyze S in more detail. Here, we focus on the situation in which S is a *point cloud* (finite point set) which we usually denote by P .

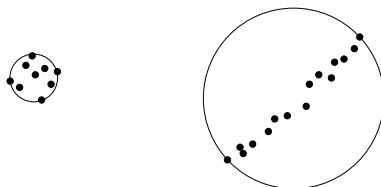


Figure 5.2: Smallest enclosing balls $B(P)$ of point sets $P \subseteq \mathbb{R}^2$

What shapes to use depends on the questions you want to ask, of course. If you

want to know, for example, how the set $P \subseteq \mathbb{R}^2$ must be translated and scaled such that a printout of it fills a sheet of paper, you want to use *axes-parallel bounding boxes*. Finding the smallest axes-parallel bounding box (or simply the bounding box) is easy. For every $i \in \{1, \dots, d\}$, iterate over the i -th coordinate of all points to find the smallest and largest one. This results in an algorithm of total runtime $O(dn)$ for $|P| = n$. See Figure 5.1 for two examples. The resulting box $Q(P)$ is smallest with respect to volume and diameter.

Alternatively, you may consider other bounding volumes. A popular one is the *smallest enclosing ball*, see Figure 5.2.

Here, *smallest* refers to the radius (or equivalently, the volume) of the ball. In the exercises, we ask you to prove that the diameter of the optimal ball $B(P)$ is never larger than the diameter of the bounding box $Q(P)$. This means, if you consider the diameter as a measure of how well the bounding volume approximates the point set, balls are better than boxes. On the other hand, if you consider the *volume* as the measure of quality, there is no clear winner. Superimposing Figures 5.1 and 5.2, you see that the ball has smaller volume than the box in the left situation, but larger volume in the right situation.

There is another aspect where the ball is the winner: rotating P changes the shape and diameter of the bounding box, but the smallest enclosing ball stays the same (up to translation). This is a desirable property in many geometric applications, because it means that the smallest enclosing ball does not have to be recomputed when we apply an *isometry* (any affine transformation¹ that preserves lengths of difference vectors) to P .

Other popular bounding volumes are boxes of arbitrary orientation (images of some $Q_d(\underline{b}, \bar{b})$ under an isometry) and ellipsoids, mostly because they approximate the volume of $\text{conv}(P)$ well. They are more difficult to compute than (axes-parallel) boxes and balls, though.

5.2 Finding an almost optimal ball

For $P \subseteq \mathbb{R}^d, |P| = n > 0$, we let $B(P)$ denote the ball of smallest radius that contains P . It is well-known that $B(P)$ exists and is unique [3], but how do we (efficiently) compute it? Recall that the bounding box was easy to find in time $O(dn)$. Can we also compute $B(P)$ in time $O(dn)$? There is no rigorous argument that this is not possible, but there are good reasons to believe that one cannot do it. At least, all known algorithms for computing $B(P)$ are much slower—so slow in fact that they will not be able to solve the problem for $n = d = 1,000$, say.

The goal of this chapter is to prove that time $O(dn)$ suffices to find a ball containing P whose radius is only 1% larger than the radius of $B(P)$. If this is not good enough, the same algorithm can produce a ball whose radius is only 0.1% larger, at the expense of running (essentially) ten times as long. In fact, any desired percentage can be

¹a mapping $x \rightarrow Ax + b$, with $A \in \mathbb{R}^{d \times d}$ a matrix and $b \in \mathbb{R}^d$ a translation vector

achieved in time $O(dn)$, but with the constant behind the big O depending on the percentage. The algorithm (actually, a combination of two algorithms) is due to Bădoiu and Clarkson [1].

5.2.1 Basics

Here is what we mean by (the center of) an almost optimal ball, with respect to a given constant $\varepsilon \geq 0$.

Definition 5.2.1 Let R_P be the radius of $B(P)$, $\varepsilon \geq 0$. A point $c \in \mathbb{R}^d$ is called $(1 + \varepsilon)$ -approximation of $B(P)$, if

$$\max_{q \in P} \|q - c\| \leq (1 + \varepsilon)R_P.$$

Note that the center c_P of $B(P)$ is a 1-approximation of $B(P)$. We will need the following statement about c_P . The origin of it is unknown to me, and often it is conceived as a more or less obvious fact. Bădoiu and Clarkson remark that the statement is proved in a paper by Goel et al. [2] which is not true. But anyway, it is a simple exercise to prove the following lemma.

Lemma 5.2.2 For any $c \in \mathbb{R}^d$, there exists a point $p \in P$ such that

- (i) $\|p - c_P\| = R_P$, and
- (ii) $\|p - c\|^2 \geq \|c - c_P\|^2 + \|p - c_P\|^2$,

see Figure 5.3 (left).

If $c \neq c_P$ and $|P| > 1$ (which is the case we are interested in), this lemma says that we can find a point on the boundary of $B(P)$ such that the nonzero vectors $p - c_P$ and $c - c_P$ span an obtuse angle (an angle of at least 90°). This interpretation of the inequality in Lemma 5.2.2 (ii) is a consequence of cosine theorem.

In more geometric terms, the inequality can also be interpreted as follows: if $c \neq c_P$, the unique hyperplane

$$h = \{x \in \mathbb{R}^d \mid (x - c_P) \cdot (c - c_P) = 0\}$$

through c_P with normal vector $c - c_P$ must have a boundary point of $B(P)$ in the halfspace

$$h^c := \{x \in \mathbb{R}^d \mid (x - c_P) \cdot (c - c_P) \leq 0\}$$

not containing c , see Figure 5.3 (left). The right part of the figure shows on an intuitive level why this condition is necessary: if no point p satisfies the criteria of the Lemma, we can shrink $B(P)$ and get a still smaller ball containing P , which is a contradiction to $B(P)$ being smallest.

Using Lemma 5.2.2, we can easily prove that a $(1 + \varepsilon)$ -approximation cannot be too far away from c_P , just like we would expect it.

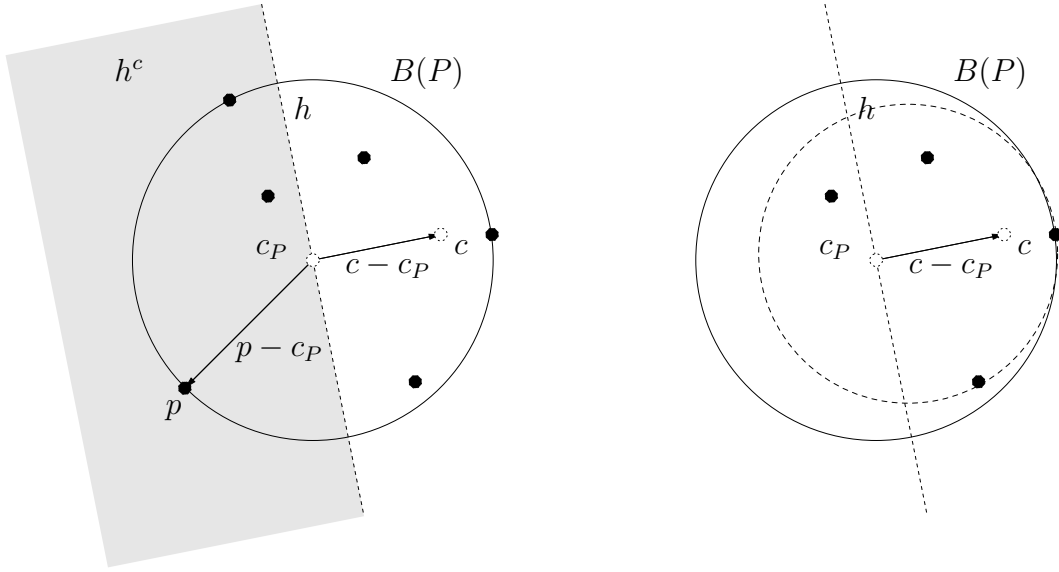


Figure 5.3: Illustration of Lemma 5.2.2

Lemma 5.2.3 *Let $c \in \mathbb{R}^d$ be a $(1 + \varepsilon)$ -approximation of $B(P)$. Then*

$$\|c - c_P\| \leq \sqrt{2\varepsilon + \varepsilon^2} R_P.$$

Proof. Choose p according to Lemma 5.2.2. Because p is on the boundary of $B(P)$ and c is a $(1 + \varepsilon)$ -approximation, we get

$$\|c - c_P\|^2 \leq \|p - c\|^2 - \|p - c_P\|^2 \leq (1 + \varepsilon)^2 R_P^2 - R_P^2 = (2\varepsilon + \varepsilon^2) R_P^2.$$

□

5.2.2 Algorithm 1

Here is our first algorithm for approximating $B(P)$. For given P and $\varepsilon > 0$, this algorithm computes a sequence of centers $c_i, i = 1, \dots, \lceil 1/\varepsilon^2 \rceil$, with the property that the last one is a $(1 + \varepsilon)$ -approximation of c_P .

MiniballApprox1(P, ε):

(* computes $(1 + \varepsilon)$ -approximation of $B(P)$ *)
 choose $p \in P$ arbitrarily and set $c_1 := p$
 FOR $i = 2$ TO $\lceil 1/\varepsilon^2 \rceil$ DO
 choose $q \in P$ such that $\|q - c_{i-1}\|$ is maximum
 $c_i := c_{i-1} + \frac{1}{i}(q - c_{i-1})$
 END
 RETURN $c_{\lceil 1/\varepsilon^2 \rceil}$

Theorem 5.2.4 For $i = 1, \dots, \lceil 1/\varepsilon^2 \rceil$,

$$\|c_i - c_P\| \leq \frac{R_P}{\sqrt{i}}.$$

This invariant immediately implies the approximation factor of the algorithm: using the triangle inequality, we get that for all $s \in P$,

$$\|s - c_i\| \leq \|s - c_P\| + \|c_i - c_P\| \leq R_P + \frac{R_P}{\sqrt{i}} = \left(1 + \frac{1}{\sqrt{i}}\right) R_P. \quad (5.1)$$

It follows that c_i is a $(1 + 1/\sqrt{i})$ -approximation; setting $i = \lceil 1/\varepsilon^2 \rceil$ gives the desired $(1 + \varepsilon)$ -approximation.

Proof. If $|P| = 1$, we have $c_i = c_P$ in any iteration and the result follows. Otherwise, we proceed by induction. The statement holds for $i = 1$ because of $c_1 \in P$. Now assume that $i > 1$ and that we have already verified the theorem for $i - 1$. Let us apply Lemma 5.2.2 with $c = c_{i-1}$, providing us with a point p . Let q be the point chosen in iteration i .

Claim.

$$\|q - c_{i-1}\|^2 \geq \|c_{i-1} - c_P\|^2 + \|q - c_P\|^2. \quad (5.2)$$

To see this, we use the way q was chosen, together with p 's properties (i) and (ii) from Lemma 5.2.2 to conclude that

$$\|q - c_{i-1}\|^2 \geq \|p - c_{i-1}\|^2 \stackrel{(ii)}{\geq} \|c_{i-1} - c_P\|^2 + \|p - c_P\|^2 \stackrel{(i)}{\geq} \|c_{i-1} - c_P\|^2 + \|q - c_P\|^2.$$

Note that for $c_{i-1} \neq c_P$, equation (5.2) is equivalent to $q \in h^{c_{i-1}}$, see Figure 5.4 and the discussion after Lemma 5.2.2.

The claimed bound for $\|c_i - c_P\|$ is obvious if $c_i = c_P$, so we will assume that $c_i \neq c_P$. Moreover, $|P| > 1$ implies $c_i \neq c_{i-1}$ (why?), and $i > 1$ implies $q \neq c_i$, see the situation in Figure 5.4.

We now restrict attention to the triangle spanned by c_P , c_{i-1} and q (this triangle also contains c_i); assume $c_P - c_i$ and $c_{i-1} - c_i$ span some angle β , so that $c_P - c_i$ and $q - c_i$ span angle $180^\circ - \beta$.² Let us introduce the following shortcuts.

$$\begin{aligned} x &:= \|c_i - c_P\|, \\ a &:= \|c_{i-1} - c_P\|, \\ b &:= \|q - c_P\|, \\ \kappa &:= \|c_{i-1} - c_i\|, \\ \mu &:= \|q - c_i\|, \\ \lambda &:= \|q - c_{i-1}\| = \kappa + \mu. \end{aligned}$$

²Because all involved vectors are nonzero, these angles are well-defined, even if the triangle spanned by c_P , c_{i-1} and q is flat.

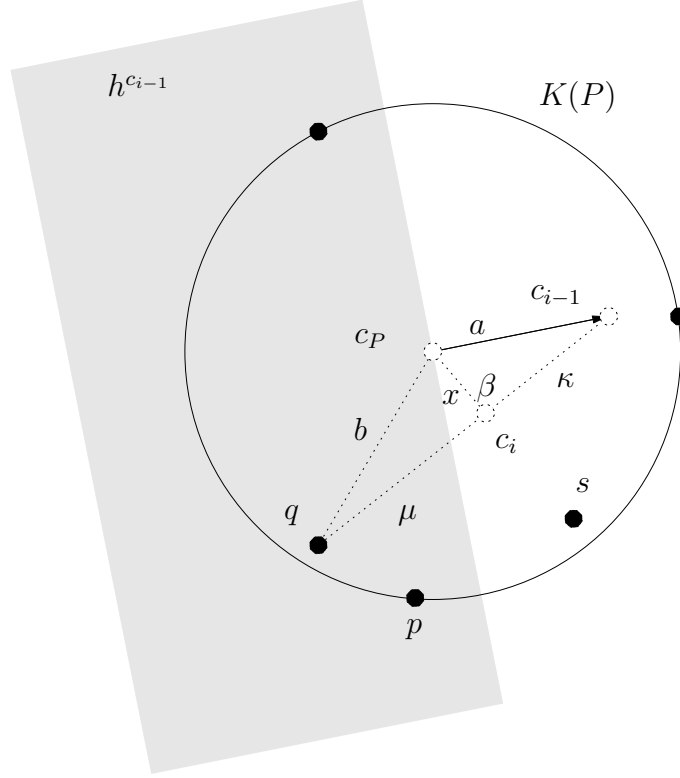


Figure 5.4: Proof of Theorem 5.2.4

Recalling that $\cos(180^\circ - \beta) = -\cos(\beta)$, the cosine theorem (applied to the two subtriangles involving $c_i - c_P$) yields

$$x^2 = a^2 - \kappa^2 + 2x\kappa \cos \beta, \quad (5.3)$$

$$x^2 = b^2 - \mu^2 - 2x\mu \cos \beta. \quad (5.4)$$

By definition of c_i , we have

$$\kappa = \frac{1}{i}\lambda, \quad \mu = \frac{i-1}{i}\lambda.$$

Therefore, (5.3) and (5.4) can be rewritten as follows.

$$x^2 = a^2 - \left(\frac{1}{i}\right)^2 \lambda^2 + 2x\frac{1}{i}\lambda \cos \beta, \quad (5.5)$$

$$x^2 = b^2 - \left(\frac{i-1}{i}\right)^2 \lambda^2 - 2x\frac{i-1}{i}\lambda \cos \beta. \quad (5.6)$$

Multiplying (5.5) with $i-1$ and adding up the two equations removes the contribution of β and x on the right hand side, and we get

$$ix^2 = (i-1)a^2 + b^2 - \frac{i-1}{i}\lambda^2. \quad (5.7)$$

Inequality (5.2) exactly says that $\lambda^2 \geq a^2 + b^2$, so that (5.7) further yields

$$\begin{aligned} x^2 &\leq \frac{1}{i} \left((i-1)a^2 + b^2 - \frac{i-1}{i}(a^2 + b^2) \right) \\ &= \left(\frac{i-1}{i} \right)^2 a^2 + \left(\frac{1}{i} \right)^2 b^2. \end{aligned}$$

Inductively, we know that $a^2 = \|c_{i-1} - c_P\|^2 \leq R_P^2/(i-1)$, and because $b^2 = \|q - c_P\|^2 \leq R_P^2$, the desired inequality

$$x^2 \leq \frac{R_P^2}{i}$$

follows. □

For $|P| = n$, the runtime of Algorithm `Miniball_Approx1` is $O(dn)$ for one iteration (we need to find the largest among n scalar products), meaning that the total runtime is

$$O\left(\frac{dn}{\varepsilon^2}\right).$$

5.2.3 Algorithm 2

The denominator of ε^2 in the runtime of Algorithm `Miniball_Approx1` is not very satisfactory. Even a moderate error bound of $\varepsilon = 0.01$ (1%) leads to 10,000 iterations in the algorithm; that way, we cannot compete with the bounding box.

Here is a second algorithm that achieves a denominator of ε . The prize to pay is that this algorithm requires as a black box the computation of the *exact* smallest enclosing ball of a small point set (the call to this black box can be replaced with a call to `Miniball_Approx1`, though).

`Miniball_Approx2`(P, ε):

```
(* computes  $(1 + \varepsilon)$ -approximation of  $B(P)$  *)
choose  $p \in P$  arbitrarily and set  $c_1 := p, S_1 := \{p\}, j := 0, \Delta := \infty$ 
FOR  $i = 2$  TO  $\lceil 2/\varepsilon \rceil$  DO
  choose  $q \in P$  such that  $\|q - c_{i-1}\|$  is maximum
  IF  $\|q - c_{i-1}\| < \Delta$  THEN
     $j := i - 1$ 
     $\Delta := \|q - c_{i-1}\|$ 
  END
   $S_i := S_{i-1} \cup \{q\}$ 
  compute the smallest enclosing ball  $B(S_i) = B_d(c_i, R_i)$ 
END
IF  $\max_{s \in P} \|s - c_{\lceil 2/\varepsilon \rceil}\| < \Delta$  THEN
   $j = \lceil 2/\varepsilon \rceil$ 
END
RETURN  $c_j$ 
```

In the analysis of the algorithm, we work with the following symbols.

$$\begin{aligned}
R &:= \text{radius } R_P \text{ of } B(P), \\
\bar{R} &:= (1 + \varepsilon)R, \\
c_i &:= \text{center of } B(S_i), \\
R_i &:= \text{radius of } B(S_i) \text{ (note that } R_1 = 0), \\
\lambda_i &:= R_i/\bar{R}, \\
k_i &:= \|c_i - c_{i-1}\|.
\end{aligned}$$

Observe that for all i ,

$$\lambda_i \leq R/\bar{R} = 1/(1 + \varepsilon), \quad (5.8)$$

because $S_i \subseteq P$, so $B(S_i)$ cannot have larger radius than $B(P)$. Let us assume that no c_i is a $(1 + \varepsilon)$ -approximation of $B(P)$. We will show that under this assumption, $\lambda_{\lceil 2/\varepsilon \rceil + 1}$ exceeds the bound in (5.8), a contradiction.³

To analyze the development of λ_i in iteration $i \geq 2$, we first apply Lemma 5.2.2 to the set S_{i-1} and the point $c = c_i$ to deduce the existence of $p \in S_{i-1}$ such that

$$\|p - c_i\|^2 \geq \|c_i - c_{i-1}\|^2 + \|p - c_{i-1}\|^2 = k_i^2 + R_{i-1}^2,$$

meaning that

$$\|p - c_i\| \geq \sqrt{\lambda_{i-1}^2 \bar{R}^2 + k_i^2}. \quad (5.9)$$

Furthermore, for the point q chosen in iteration i , the triangle inequality yields $\|q - c_{i-1}\| \leq \|q - c_i\| + \|c_i - c_{i-1}\|$, implying

$$\|q - c_i\| \geq \|q - c_{i-1}\| - \|c_i - c_{i-1}\| = \|q - c_{i-1}\| - k_i > \bar{R} - k_i. \quad (5.10)$$

In the last inequality (and only here), we use the assumption that c_{i-1} is not a $(1 + \varepsilon)$ -approximation, meaning that

$$\|q - c_{i-1}\| = \max_{s \in P} \|s - c_{i-1}\| > (1 + \varepsilon)R = \bar{R}.$$

We are now approaching a recursive lower bound for λ_i . Because both p and q are in S_i , we get—using (5.9) and (5.10)—that

$$\lambda_i \bar{R} = R_i \geq \max(\|p - c_i\|, \|q - c_i\|) \geq \max\left(\sqrt{\lambda_{i-1}^2 \bar{R}^2 + k_i^2}, \bar{R} - k_i\right). \quad (5.11)$$

The first term of the maximum increases with k_i , while the second term decreases. It follows that the maximum is minimized when both terms are equal, so when

$$k_i = \frac{1 - \lambda_{i-1}^2}{2} \bar{R}$$

³you may object that $\lambda_{\lceil 2/\varepsilon \rceil + 1}$ does not appear in the algorithm, but for this argument, we simply consider a hypothetical last iteration with $i = \lceil 2/\varepsilon \rceil + 1$.

and consequently

$$\sqrt{\lambda_{i-1}^2 \bar{R}^2 + k_i^2} = \bar{R} - k_i = \frac{1 + \lambda_{i-1}^2}{2} \bar{R}.$$

Substituting this into (5.11) yields

$$\lambda_i \geq \frac{1 + \lambda_{i-1}^2}{2}, \quad i \geq 2, \quad (5.12)$$

with $\lambda_1 = R_1/\bar{R} = 0$. Equation (5.12) is equivalently written as

$$1 - \lambda_i \leq \frac{1 - \lambda_{i-1}^2}{2}, \quad i \geq 2,$$

which in turn implies

$$\frac{1}{1 - \lambda_i} \geq \frac{2}{(1 - \lambda_{i-1})(1 + \lambda_{i-1})} = \frac{1}{1 - \lambda_{i-1}} + \frac{1}{1 + \lambda_{i-1}} > \frac{1}{1 - \lambda_{i-1}} + \frac{1}{2}, \quad i \geq 2,$$

because $\lambda_{i-1} < 1$. By expanding this, we get

$$\frac{1}{1 - \lambda_i} > \frac{i-1}{2} + \frac{1}{1 - \lambda_1} = 1 + \frac{i-1}{2}, \quad i \geq 1.$$

In other words,

$$\lambda_i > 1 - \frac{1}{1 + (i-1)/2}, \quad i \geq 1.$$

For $i = \lceil 2/\varepsilon \rceil + 1$, this gives

$$\lambda_i > 1 - \frac{1}{1 + 1/\varepsilon} = \frac{1}{1 + \varepsilon},$$

a contradiction to (5.8). Therefore, our initial assumption was wrong, and there must be some $c_{i_0}, i_0 \in \{1, \dots, \lceil 2/\varepsilon \rceil\}$, which yields a $(1 + \varepsilon)$ -approximation. By the choice of j , the point c_j returned by the algorithm satisfies

$$\max_{s \in P} \|s - c_j\| \leq \max_{s \in P} \|s - c_{i_0}\| \leq (1 + \varepsilon)R,$$

so c_j is itself a $(1 + \varepsilon)$ -approximation.

The runtime of the algorithm is bounded by

$$O\left(\frac{dn}{\varepsilon} + \frac{1}{\varepsilon} f_d(\lceil 2/\varepsilon \rceil - 1)\right),$$

with $f_d(n)$ the time necessary to compute the exact smallest enclosing ball of a set of n points in \mathbb{R}^d . The known bounds for $f_d(n)$ are exponential in d , but by using `MiniballApprox1` instead, we can replace this by a bound which is polynomial in d and $1/\varepsilon$ (we omit the details here).

5.2.4 Core sets

Our analysis of Algorithm `Miniball_Approx2` has a very interesting consequence which we want to state explicitly (choose $S = S_j$ to get it).

Corollary 5.2.5 *For any finite point set $P \subseteq \mathbb{R}^d$ and $\varepsilon > 0$, there exists a subset $S \subseteq P$, $|S| \leq \lceil 2/\varepsilon \rceil$, such that the center of $B(S)$ is a $(1 + \varepsilon)$ -approximation of $B(P)$.*

Such a set is called a *core set*, and we will encounter core sets for other bounding volumes later in the course.

According to the exercises, a subset with the *same* smallest enclosing ball as P may require up to $d + 1$ points (and this is tight). If you are willing to accept a small error of $\varepsilon = 0.01$, say, you can find a subset S of *constant* size 200 with ‘the same’ smallest enclosing ball as P , in *any* dimension. This is quite remarkable, and also exceptional. While other bounding volumes for P do have core sets whose sizes do not depend on $n = |P|$, there is usually an (exponential) dependence on d , on top of the dependence on ε .

5.3 Coresets and balls in other norms

Let us reformulate Corollary 5.2.5 somewhat differently: For any finite point set $P \subseteq \mathbb{R}^d$ and $\varepsilon > 0$, there exists a subset $S \subseteq P$, $|S| \leq \lceil 2/\varepsilon \rceil$, such that all points of P are within distance εR_S of $B(S)$. Here, $B(S)$ is the smallest enclosing ball of S , and R_P is the radius of the smallest enclosing ball $B(P)$ of P .

It turns out that a similar statement also holds if we generalize our concept of “balls”.

Bibliography

- [1] M. Bădoiu and K. L. Clarkson. Optimal core-sets for balls. submitted, 2002.
- [2] A. Goel, P. Indyk, and K. R. Varadarajan. Reductions among high-dimensional proximity problems. In *Proc. 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 769–778, 2001.
- [3] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In H. Maurer, editor, *New Results and New Trends in Computer Science*, volume 555 of *Lecture Notes in Computer Science*, pages 359–370. Springer-Verlag, 1991.