Chapter 5

Cuboids

We have already seen that we can efficiently find the bounding box $Q(P)$ and an arbitrarily good approximation to the smallest enclosing ball $B(P)$ of a set $P \subseteq \mathbb{R}^d$. Unfortunately, both bounding volumes are bad when the task is to approximate the volume of $\mathrm{conv}(P)$. As you can see from Figures 4.1 and 4.2, the ratios

$$\frac{\mathrm{vol}(Q(P))}{\mathrm{vol}(\mathrm{conv}(P))} \text{ and } \frac{\mathrm{vol}(B(P))}{\mathrm{vol}(\mathrm{conv}(P))}$$

can get arbitrarily large even for $d = 2$, and if $\mathrm{conv}(P)$ has nonzero volume: as the points in P get closer and closer to some fixed—nonvertical and nonhorizontal—line segment, the volume of the convex hull becomes smaller and smaller, while the volumes of $Q(P)$ and $B(P)$ converge to nonzero constants.

In this chapter, we show that boxes of *arbitrary orientations* are better with respect to volume. To distinguish them from the (axis-parallel) boxes, we call them *cuboids*, see Figure 5.1. Formally, a cuboid is any set of the form

$$C = \{Mx \mid x \in Q_d(\underline{b}, \overline{b}), M^{-1} = M^T\}. \tag{5.1}$$

A matrix M with $M^{-1} = M^T$ is called *orthogonal*, and in the orthogonal coordinate system defined by the columns of M, C is an axis-parallel box.

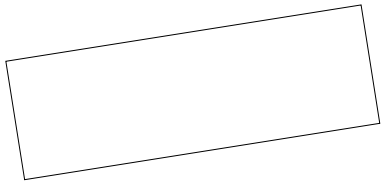

Figure 5.1: A cuboid in $\mathbb{R}^2$

5.1 Approximating the smallest enclosing cuboid

The exact smallest enclosing cuboid $C(P)$ of set $P \subseteq \mathbb{R}^d$ can be computed in time $O(n \log n)$ for $d = 2$ and $O(n^3)$ for $d = 3$. No better algorithms are known. In contrast, $Q(P)$ and $B(P)$ can be computed in optimal time $O(n)$ for $d = 2, 3$ [5]. This already shows that $C(P)$ is a more complicated object then $B(P)$, and that we should be more modest regarding the quality of approximation we can expect. Actually, $C(P)$ is not even well-defined, because there is not necessarily a unique smallest enclosing cuboid, see Figure 5.2. When we are writing $C(P)$, we therefore mean *some* smallest enclosing cuboid, and with a little care, this poses no problems. For example, the quantity $\mathrm{vol}(C(P))$ is well-defined, because it does not depend on the particular choice of $C(P)$.

Figure 5.2: A set may have more than one smallest enclosing cuboid

Here is the main result of this section.

Theorem 5.1.1 *For $P \subseteq \mathbb{R}^d$ with $|P| = n$, we can compute in $O(d^2 n)$ time a cuboid C that contains P and satisfies*

$$\mathrm{vol}(C) \leq 2^d d! \mathrm{vol}(C(P)).$$

This means, for any constant dimension d, we can approximate the volume of the smallest cuboid that contains P up to some constant factor; however, this factor is already pretty bad when the dimension is only moderately high. On the other hand, the result is not as bad as it might look like, and the exponential dependence on d seems very hard to avoid.

To see this, let's look at smallest enclosing balls again. In Chapter 4, we have shown that we can find an enclosing ball B of P whose radius is at most $(1 + \varepsilon)$ times the radius of $B(P)$, for any $\varepsilon > 0$. With respect to *volume*, this means that

$$\mathrm{vol}(B) \leq (1 + \varepsilon)^d \mathrm{vol}(B(P)),$$

which is exponential in d. In view of this, the bound in Theorem 5.1.1 starts to look somewhat more reasonable.

In order to prove the theorem, we first describe an algorithm for computing some bounding cuboid C and then argue about the quality of C. Actually, the algorithm

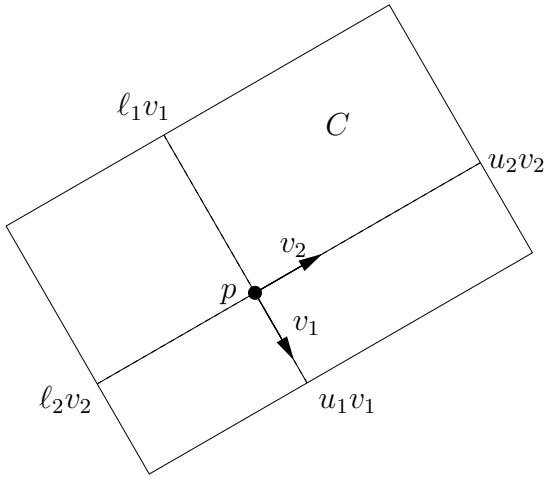

Figure 5.3: Cuboid defined by the algorithm, $d = 2$

computes a point $p \in \mathbb{R}^d$, a set of *pairwise orthogonal*[1] unit vectors $v_1, \ldots, v_d$, and numbers $\ell_k \leq u_k, k = 1, \ldots, d$ such that

$$C = \{p + \sum_{k=1}^{d} \lambda_k v_k \mid \ell_k \leq \lambda_k \leq u_k, k = 1, \ldots, d\}, \tag{5.2}$$

see Figure 5.3. In the exercises, we ask you to verify that this set C is indeed a cuboid.

We will assume that $\mathbf{0} \in P$. This can be achieved through a translation of P in time $O(dn)$. Now we call the following recursive algorithm with parameters (P, d), meaning that the global invariant holds in the beginning. The algorithm constructs v_k, l_k, u_k one after another, starting with $k = d$. The point p used in (5.2) will be $p = 0$.

`MinCuboid_Approx`(P, k):
  (* Global invariant: $p \cdot v_i = 0$ for $p \in P, i = k+1, \ldots, d$ *)
  choose $q \in P$ such that $\|q\|$ is maximum
  `IF` $q \neq \mathbf{0}$ `THEN`
    $v_k = q/\|q\|$
    $\ell_k = \min_{p \in P} p \cdot v_k$
    $u_k = \|q\|$
    $P' = \{p - (p \cdot v_k)v_k \mid p \in P\}$
    (* Local invariant: $p' \cdot v_k = 0$ for $p' \in P'$ *)
    `MinCuboid_Approx`$(P', k-1)$
  `END`

Before we analyze the algorithm, let us try to get a geometric intuition for the top-level call with $k = d$ (Figure 5.4). The vector v_d is a unit vector pointing in the direction

[1]Two vectors v, w are orthogonal if $v \cdot w = 0$.

of some point q farthest away from $\mathbf{0} \in P$. Because $p \cdot v_d$ is the (signed) length of p's projection onto the direction v_d, the value $u_d - \ell_d = \|q\| - \ell_d$ is the extent of P along direction v_d. P' is the projection of P onto the unique hyperplane h through the origin that is orthogonal to v_d. For P', the recursive call finds a $(d-1)$-dimensional bounding cuboid C' within h, which we then extend along direction v_d to a cuboid C containing P.

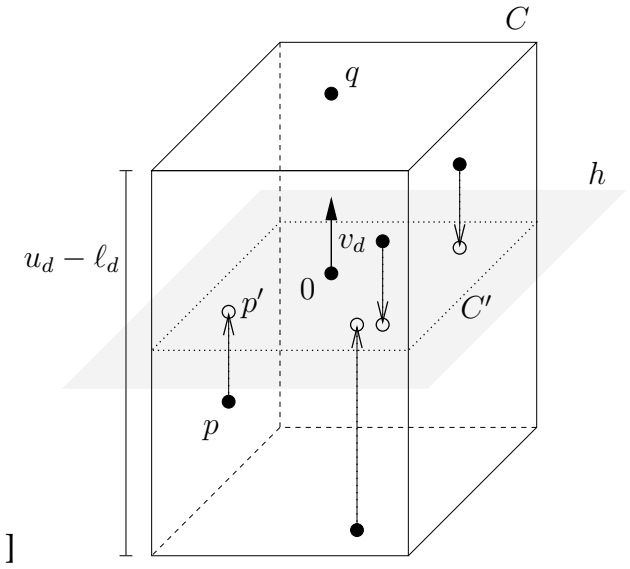

Figure 5.4: Geometric picture of the algorithm

5.1.1 Correctness and runtime of the algorithm.

Because $\|v_k\| = 1$ and $p' \cdot v_k = p \cdot v_k - (p \cdot v_k)\|v_k\|^2 = 0$, the local invariant holds for all $p' \in P'$. The global invariant also holds in the recursive call, because of the just established local invariant, and because the global invariant for $p, q \in P$ shows that for $i = k+1, \ldots, d$,

$$\begin{aligned} p' \cdot v_i &= (p - (p \cdot v_k)v_k) \cdot v_i \\ &= \underbrace{p \cdot v_i}_{=0} - (p \cdot v_k)(v_k \cdot v_i) = -(p \cdot v_k)\underbrace{(q \cdot v_i)}_{=0}/\|q\| = 0. \end{aligned}$$

The latter equation also shows that $v_k \cdot v_i = 0, i = k+1, \ldots, d$ which yields the pairwise orthogonality of all the v_k. Because there are only d pairwise orthogonal nonzero vectors, the recursion bottoms out for some value $\underline{k} \geq 0$. This implies the runtime of $O(d^2 n)$.

To prove that the cuboid

$$C = \{ \sum_{i=\underline{k}+1}^{d} \lambda_i v_i \mid \ell_i \leq \lambda_i \leq u_i, i = \underline{k}+1, \ldots, d\} \tag{5.3}$$

contains P, we proceed by induction on d. For $d = 0$, or if the condition of the `IF` clause fails, we have $P = C = \{\mathbf{0}\}$ and $\underline{k} = d$, so the claim holds. Now assume $d > 0$, and we have already proved the claim for $d - 1$. Inductively, we then know that all points $p' \in P'$ can be written in the form

$$p' = \sum_{i=k_0+1}^{d-1} \lambda_i v_i, \quad \ell_i \leq \lambda_i \leq u_i, i = \underline{k} + 1, \ldots, d-1. \tag{5.4}$$

Furthermore, the definition of p' yields

$$p = p' + (p \cdot v_d)v_d, \tag{5.5}$$

where

$$\ell_d \leq p \cdot v_d \leq |p \cdot v_d| \leq \|p\|\|v_d\| = \|p\| \leq \|q\| = u_d,$$

by the Cauchy-Schwarz inequality. Therefore, setting $\lambda_d = (p \cdot v_d)$ and plugging (5.4) into (5.5) gives the desired conclusion $p \in C$.

5.1.2 Quality of the algorithm

It remains to bound the volume of the cuboid C resulting from the algorithm according to (5.3). If the value of $\underline{k}$ for which the recursion bottoms out satisfies $\underline{k} > 0$, we have $\mathrm{vol}(C) = 0$ and are done, so we will assume that $\underline{k} = 0$.

Let $P^{(k)}, k = 1, \ldots, d$, be the iterated projection of P that the algorithm considers in the recursive call with second parameter k. We have

$$P^{(d)} = P, \quad P^{(d-1)} = \{p - (p \cdot v_d)v_d \mid p \in P^{(d)}\},$$

and using the pairwise orthogonality of the v_k, we can inductively verify the general formula

$$P^{(k)} = \{p - \sum_{i=k+1}^{d} (p \cdot v_i)v_i \mid p \in P\}, \quad k = 1, \ldots, d. \tag{5.6}$$

Let $q_k \in P^{(k)}$ be the point chosen for second parameter k, and let $p_k \in P$ be the point of P whose (iterated) projection q_k is. As a consequence of the previous equation we have

$$q_k = p_k - \sum_{i=k+1}^{d} (p_k \cdot v_i)v_i, \quad k = 1, \ldots, d. \tag{5.7}$$

The approximation ratio of Theorem 5.1.1 is derived using two ingredients: a *lower* bound on the volume of the smallest enclosing cuboid $C(P)$, and an *upper* bound on the volume of the cuboid C computed by the algorithm.

A lower bound for $\mathrm{vol}(C(P))$**.** We consider the two sets

$$S_q = \mathrm{conv}(\{\mathbf{0}, q_d, \ldots, q_1\}), \quad S_p = \mathrm{conv}(\{\mathbf{0}, p_d, \ldots, p_1\}).$$

Because the q_i are pairwise orthogonal, they are in particular linearly independent. In this case, we can apply the well-known formula for the volume of a *simplex* in $\mathbb{R}^d$ to obtain

$$\mathrm{vol}(S_q) = \frac{1}{d!}|\det(q_d, \ldots, q_1)|.$$

On the other hand, (5.7) shows that q_k equals p_k plus some linear combination of the $q_i, i > k$ (note that v_i is a multiple of q_i). Recalling that the determinant of a set of vectors does not change when we add to some vector a linear combination of the other vectors, we consequently get (how?) that

$$\det(q_d, \ldots, q_1) = \det(p_d, \ldots, p_1),$$

and therefore

$$\mathrm{vol}(S_p) = \mathrm{vol}(S_q) = \frac{1}{d!}\prod_{k=1}^{d}\|q_k\|,$$

using orthogonality of the q_k. Geometrically, this shows that we can transform the simplex S_q into the simplex S_p, by always moving one of the points *parallel* to its opposite side. During such a movement, the volume stays constant, see Figure 5.5.

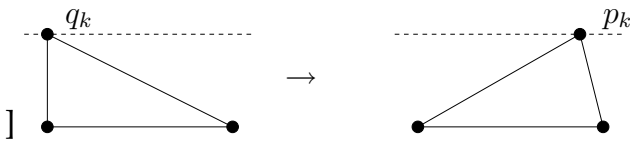

Figure 5.5: Moving a vertex parallel to the opposite side does not change the volume

Now we have a lower bound for the volume of the smallest cuboid $C(P)$. Because $S_p \subseteq \mathrm{conv}(P) \subseteq C(P)$, we get

$$\mathrm{vol}(C(P)) \geq \mathrm{vol}(\mathrm{conv}(P)) \geq \mathrm{vol}(S_p) = \frac{1}{d!}\prod_{k=1}^{d}\|q_k\|. \tag{5.8}$$

An upper bound for $\mathrm{vol}(C)$**.** By construction of C, we have

$$\mathrm{vol}(C) = \prod_{k=1}^{d}(u_k - \ell_k),$$

because $u_k - \ell_k$ is the extent of C in direction of the unit vector v_k. Let $\underline{p}$ be the point defining ℓ_k in the recursive call of the algorithm with second parameter k. We get

$$\begin{array}{rcll} u_k - \ell_k & \leq & |u_k| + |\ell_k| = \|q_k\| + |\underline{p} \cdot v_k| & \\ & \leq & \|q_k\| + \|\underline{p}\|\|v_k\| & \text{(Cauchy-Schwarz inequality)} \\ & = & \|q_k\| + \|\underline{p}\| & (\|v_k\| = 1) \\ & \leq & 2\|q_k\|. & \text{(choice of } q) \end{array}$$

It follows that

$$\mathrm{vol}(C) \leq 2^d \prod_{k=1}^{d} \|q_k\| \overset{(5.8)}{\leq} 2^d d! \mathrm{vol}(C(P)),$$

which is what we wanted to prove in Theorem 5.1.1.

Looking at (5.8), we see that we have actually proved a little more.

Corollary 5.1.2 *Let C be the cuboid computed by a call to* `MinCuboid_Approx`(P, d), $P \subseteq \mathbb{R}^d$. *Then*

$$\frac{\mathrm{vol}(C(P))}{\mathrm{vol}(\mathrm{conv}(P))} \leq \frac{\mathrm{vol}(C)}{\mathrm{vol}(\mathrm{conv}(P))} \leq 2^d d!.$$

We therefore have shown that $C(P)$ is strictly better than $Q(B)$ and $B(P)$ when it comes to approximating the volume: the volume of any smallest enclosing cuboid of P is only by a *constant factor* (depending on d) larger than the volume of $\mathrm{conv}(P)$. The same factor even holds for the cuboid C that we can easily compute in $O(d^2 n)$ time.

This cuboid satisfies an additional property that we mention without proof. This goes back to a paper by Barequet and Har-Peled [2] that also contains a sketch of the algorithm `MinCuboid_Approx`.

Theorem 5.1.3 *Let C be the cuboid computed by a call to* `MinCuboid_Approx`(P, d), $P \subseteq \mathbb{R}^d$. *There exists a constant $\alpha_d > 0$ (only depending on d) such that C, scaled with α_d and suitably translated, is contained in* $\mathrm{conv}(P)$, *see Figure 5.6.*

This means that we will find a not-too-small copy of C inside $\mathrm{conv}(P)$, and this is exploited in a number of approximation algorithms for other problems.

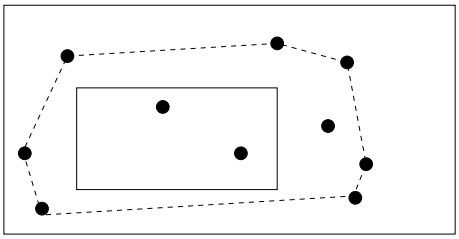

Figure 5.6: Bounding and *inscribed* cuboid of the same shape

The proof of Theorem 5.1.3 (and similar statements) involves properties of an even nicer family of bounding volumes that we introduce next.

5.2 Ellipsoids

Definition 5.2.1

(i) An ellipsoid *in $\mathbb{R}^d$ is any set of the form*

$$E = \{x \in \mathbb{R}^d \mid (x-c)^T A(x-c) \leq z,$$

where $c \in \mathbb{R}^d$ is the center *of the ellipsoid, A is a positive definite matrix,*[2] *and $z \in \mathbb{R}$.*

(ii) For $\lambda \geq 0$, the scaled *ellipsoid λE is the set*

$$\lambda E = \{c + \lambda(x-c) \mid x \in E\}.$$

This scales E relative to its center, and it is easy to prove (Exercises) that λE is an ellipsoid again.

If A is the identity matrix, for example, E is a ball with center c and squared radius z. In general, any ellipsoid is the affine image of a ball, and the image of any ellipsoid under a nonsingular affine transformation is again an ellipsoid.

Here is what makes ellipsoids attractive as bounding (and also as inscribed) volumes. This is classic material [4].

Theorem 5.2.2 *Let $K \subseteq \mathbb{R}^d$ be a convex body (this is a convex and compact set of positive volume).*

(i) There is a unique ellipsoid $\overline{E}(K)$ of smallest volume such that $K \subseteq \overline{E}(K)$ and a unique ellipsoid $\underline{E}(K)$ of largest volume such that $\underline{E}(K) \subseteq K$.

(ii) $1/d\, \overline{E}(K) \subseteq K$

(iii) $d\underline{E}(K) \supseteq K$.

(iv) If K is centrally symmetric *($p \in K \Rightarrow -p \in K$), we have $1/\sqrt{d}\, \overline{E}(K) \subseteq K$ and $\sqrt{d}\underline{E}(K) \supseteq K$.*

This is a theorem, similar in spirit to Theorem 5.1.3. The difference is that the scaling factors are explicit (and small), and that no additional translation is required. The theorem says that any convex body is 'wedged' between two concentric ellipsoids whose scale only differs by a factor of d at most.

[2] $x^T Ax > 0$ for $x \neq 0$.

Chapter 6

Directional Width

Any unit vector

$$v \in S_{d-1} := \{x \in \mathbb{R}^d \mid \|x\| = 1\}$$

defines a *direction* in $\mathbb{R}^d$. In this chapter, we are interested in approximating the "extent" of a point set in a given direction. Actually, we will approximate the extent in all directions simultaneously. You are well-aware of this concept of extent; for example, you may read in a travel guide that the extent of Switzerland in north-south direction is roughly 220 km, and its extent in east-west direction is roughly 348 km. Here is the formal definition.

Definition 6.0.3 *Let $K \subseteq \mathbb{R}^d$ be a compact set, $v \in S_{d-1}$. The* width *of K* in direction v *is defined as*

$$w_v(K) = \max_{p \in K} p \cdot v - \min_{p \in K} p \cdot v.$$

Bilinearity of the scalar product easily yields (think about it!) the following alternative definition.

$$w_v(K) = \max_{p,q \in K} (p - q) \cdot v.$$

Because v is a unit vector, $(p - q) \cdot v$ is the (signed) length of $(p - q)$'s projection onto v. The directional width $w_v(K)$ is therefore the longest such projection you can get from two points in K. Another way of thinking about the directional width is as the minimum distance between any two hyperplanes with normal vector v that have K between them, see Figure 6.1.

In case of the Switzerland example, the north-south extent is the minimum distance between two latitudes that have the country in between them, and the east-west extent is the same with longitudes.

We have the following intuitive

Fact 6.0.4 *Let $K \subseteq \mathbb{R}^d$ be a compact set. Then*

$$w_v(K) = w_v(\mathrm{conv}(K)), \quad \forall v \in S_{d-1}.$$

In the sequel, we restrict (as usual) to finite point sets.

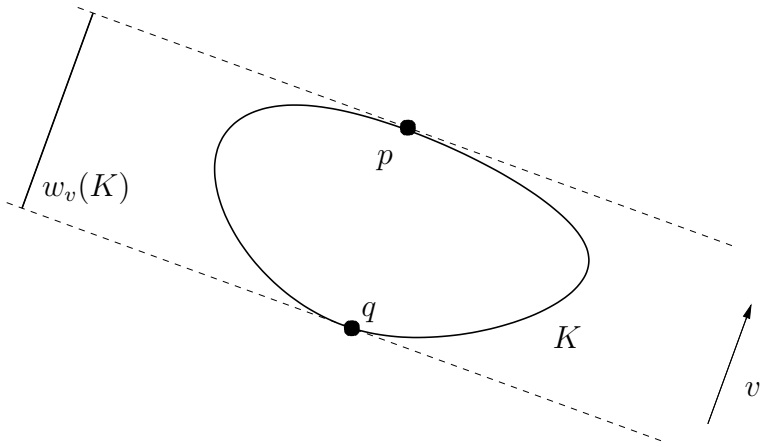

Figure 6.1: The directional width

6.1 A constant-factor approximation

Let $P \subseteq \mathbb{R}^d$ be a finite point set. We start off by showing that the bounding cuboid we have computed in Chapter 5 to approximate the volume of $\mathrm{conv}(P)$ also approximates all directional widths.

Theorem 6.1.1 *Assume that* $\mathbf{0} \in P \subseteq \mathbb{R}^d$*, and that* P *is* full-dimensional *(not contained in any hyperplane). Let*

$$C = \{\sum_{k=1}^{d} \lambda_k v_k \mid \ell_k \leq \lambda_k \leq u_k, k = 1, \ldots, d\} \tag{6.1}$$

be the bounding cuboid of P *computed by a call to the algorithm* `MinCuboid_Approx`(P, d)*. Then for all* $v \in S_{d-1}$*, the inequality*

$$w_v(C) \leq 2^{d+1} w_v(P)$$

holds.

We would like to remark that this bound is pretty bad. In approximating the volume of the convex hull, we have argued that an exponential bound in d is not that disturbing, because the volume is exponential in the scale of the point set. Any directional width, however, only grows *linearly* with the scale, and in that situation, an exponential upper bound is quite embarassing. It is not known whether the approximation factor in Theorem 6.1.1 can be improved to something polynomial in d. The trouble is that $\mathrm{conv}(P)$ might be quite long and skinny, even though its bounding cuboid is "fat".

In order to prove the theorem, we proceed as in the volume case. We first find a *lower* bound for $w_v(P)$, then an *upper* bound for $w_v(C)$. Comparing these bounds then yields the result.

Let us briefly recall how the algorithm `MinCuboid_Approx`(P, d) works, and how it sets the vales v_k, u_k, ℓ_k that define the cuboid in the statement of Theorem 6.1.1.

In round k (running from d to 1), the algorithm first chooses a point q_k of maximum norm in its current point set $P^{(k)}$ (where $P^{(d)} = P$). It then sets v_k to $q_k/\|q_k\|$ so that we get a unit vector; u_k is simply $\|q_k\|$, while ℓ_k is chosen as $\min_{p\in P^{(k)}} p \cdot v_k$. It follows that $|\ell_k| \leq u_k$.

If $P^{(k)} \neq \{\mathbf{0}\}$, the point set gets projected onto the hyperplane through the origin orthogonal to v_k, and the algorithm recursively proceeds with the resulting point set $P^{(k-1)}$ in round $k-1$. Our assumption that P is full-dimensional guarantees that the algorithm actually reaches round 1, and that the vectors $q_1, \ldots, q_d$ (or their scaled versions $v_1, \ldots, v_d$) form an orthogonal (or orthonormal) basis of $\mathbb{R}^d$.

Note that $u_k - \ell_k$ is exactly the width of $P^{(k)}$ in direction v_d, in formulas:

$$w_{v_k}(P^{(k)}) = u_k - \ell_k.$$

We also need to recall that according to (5.6), the point set $P^{(k)}$ can be expressed in terms of the original set P via

$$P^{(k)} = \{p^{(k)} \mid p \in P,\}, \quad p^{(k)} = p - \sum_{i=k+1}^{d} (p \cdot v_i)v_i. \tag{6.2}$$

A lower bound for $w_v(P)$. As in the Chapter 5, let $p_k \in P$ be the preimage of $q_k \in P^{(k)}$, meaning that $q_k = p_k^{(k)}$.

Because the directional width cannot increase when we remove points, we have that

$$w_v(P) \geq w_v(\{\mathbf{0}, p_k\}) = |p_k \cdot v|, \quad k = 1, \ldots, d,$$

and this implies

$$w_v(P) \geq \max_{k=1}^{d} |p_k \cdot v|. \tag{6.3}$$

This is our lower bound.

An upper bound for $w_v(C)$. Since we have $q_k = \|q_k\|v_k = u_k v_k$ and $|\ell_k| \leq u_k$ for all k, the cuboid (6.1) is contained in the cuboid

$$C' = \{\sum_{k=1}^{d} \lambda_k q_k \mid -1 \leq \lambda_k \leq 1\}, \tag{6.4}$$

so when we upper-bound the directional widths of the latter, we also have valid bounds for the original cuboid C.

We now want to express C' in terms of the p_k, in order to be able to compare this with our lower bound. Slightly rewriting the formula for $p^{(k)}$ in (6.2), for $p = p_k$ (observe that $q_k = p_k^{(k)}$) yields

$$p_k = q_k + \sum_{i=k+1}^{d} \frac{p_k \cdot v_i}{u_i} q_i, \quad k = 1, \ldots, d. \tag{6.5}$$

This shows that the p_k are linear combinations of the q_i, where i ranges from k to d; the equations can compactly be written in matrix form as

$$\begin{pmatrix} p_1^T \\ p_2^T \\ \vdots \\ p_{d-1}^T \\ p_d^T \end{pmatrix} = \begin{pmatrix} 1 & (p_1 \cdot v_2)/u_2 & \cdots & (p_1 \cdot v_{d-1})/u_{d-1} & (p_1 \cdot v_d)/u_d \\ 0 & 1 & \cdots & (p_2 \cdot v_{d-1})/u_{d-1} & (p_2 \cdot v_d)/u_d \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & (p_{d-1} \cdot v_d)/u_d \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_{d-1}^T \\ q_d^T \end{pmatrix}.$$

Let us denote the transformation matrix in the latter equation by A. Since A is upper-triangular with the diagonal entries being equal to 1, A is invertible, and we get

$$\begin{pmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_{d-1}^T \\ q_d^T \end{pmatrix} = A^{-1} \begin{pmatrix} p_1^T \\ p_2^T \\ \vdots \\ p_{d-1}^T \\ p_d^T \end{pmatrix}.$$

It follows that

$$\left(\sum_{k=1}^{d} \lambda_k q_k\right)^T = (\lambda_1, \ldots, \lambda_d) \begin{pmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_{d-1}^T \\ q_d^T \end{pmatrix} = (\lambda_1, \ldots, \lambda_d) A^{-1} \begin{pmatrix} p_1^T \\ p_2^T \\ \vdots \\ p_{d-1}^T \\ p_d^T \end{pmatrix} = \left(\sum_{k=1}^{d} \mu_k p_k\right)^T,$$

with

$$(\mu_1, \ldots, \mu_d) = (\lambda_1, \ldots, \lambda_d) A^{-1}.$$

Plugging this into (6.4) yields

$$C' = \{\sum_{k=1}^{d} \mu_k p_k \mid (\mu_1, \ldots, \mu_d) = (\lambda_1, \ldots, \lambda_d) A^{-1}, -1 \leq \lambda_k \leq 1\}. \tag{6.6}$$

It is important to understand that nothing fancy is going on here; the matrix A^{-1} is simply a basis transformation matrix: vectors whose coordinates $\lambda_1, \ldots, \lambda_d$ are given with respect to the basis $(q_1, \ldots, q_d)$ are expressed using coordinates $\mu_1, \ldots, \mu_d$ with respect to the basis $(p_1, \ldots, p_d)$.

In general, any invertible matrix may arise as a basis transformation matrix; but in our case, the matrix has a special structure, due to the way the p_k are constructed in the algorithm.

Observation 6.1.2 *All off-diagonal entries of A are* at most one *in absolute value.*

Proof. Consider any off-diagonal entry

$$\frac{p_k \cdot v_i}{u_i}, \quad i > k.$$

The fact that q_i was chosen in round i shows that

$$|p_k^{(i)} \cdot v_i| \leq \|p_k^{(i)}\| \leq \|q_i\| = u_i, \tag{6.7}$$

with $p_k^{(i)} \in P^{(i)}$ the iterated projection of p_k. On the other hand, computing $p_k^{(i)}$ according to (6.2) yields

$$p_k^{(i)} = p_k - \sum_{j=i+1}^{d} (p_k \cdot v_j) v_j,$$

and using pairwise orthogonality of the v_j gives

$$p_k^{(i)} \cdot v_i = p_k \cdot v_i.$$

Together with (6.7), we therefore get

$$\left| \frac{p_k \cdot v_i}{u_i} \right| = \left| \frac{p_k^{(i)} \cdot v_i}{u_i} \right| \leq \frac{u_i}{u_i} = 1.$$

□

This observation now implies bounds on the absolute values of the μ_k in (6.6). In an exercise, we ask you to show that in A^{-1}, the sum of absolute values in column k is at most 2^{k-1}. Because μ_k is a weighted sum of these values, with weights in $[-1, 1]$, we also get that $|\mu_k| \leq 2^{k-1}$ for all μ_k appearing in (6.6). This can be used to prove

Lemma 6.1.3 *The width of C' in any direction v satisfies*

$$w_v(C') \leq 2(2^d - 1) \max_{k=1}^{d} |p_k \cdot v|.$$

Together with $w_v(C) \leq w_v(C')$ and the lower bound (6.3), Theorem 6.1.1 follows. It remains to show the lemma.

Proof. Since C' is centrally symmetric ($x \in C'$ implies $-x \in C'$), we have

$$w_v(C') = 2 \max_{x \in C'} x \cdot v,$$

and it suffices to compute the latter maximum. For $x \in C'$ and $\mu_1, \ldots, \mu_d$ such that $x = \sum_{k=1}^{d} \mu_k p_k$ we get

$$x \cdot v \leq |x \cdot v| = |\sum_{k=1}^{d} \mu_k (p_k \cdot v)| \leq \sum_{k=1}^{d} |\mu_k| |p_k \cdot v| \leq \sum_{k=1}^{d} 2^{k-1} \max_{k=1}^{d} |p_k \cdot v| = (2^d - 1) \max_{k=1}^{d} |p_k \cdot v\|.$$

Because this holds for all x, we have proved a bound for $\max_{x \in C'} x \cdot v$, and the lemma follows. □

6.2 Core sets for directional width

Based on the constant-factor approximation from the previous section, we can now construct *core sets*. Let us state the theorem right away.

Theorem 6.2.1 *Let $P \subseteq \mathbb{R}^d$, $|P| = n$ and choose $\varepsilon > 0$. Furthermore, let M be such that*

$$w_v(C) \leq M w_v(P), \quad \forall v \in S_{d-1},$$

where C is the bounding cuboid of P obtained from `MinCuboid_Approx`(P, d).

(i) There exists a subset $S \subseteq P$ of size at most

$$2\left(\left\lceil \frac{2M}{\varepsilon} \right\rceil\right)^{d-1},$$

with the property that for all $v \in S_{d-1}$, $w_v(S) \geq (1-\varepsilon) w_v(P)$.

(ii) A set S as in (i)—an ε-core set of P with respect to directional widths—can be computed in time

$$O(d^2 n),$$

assuming that M is known.

With Theorem 6.1.1, we may choose $M = 2^{d+1}$. Note that even though the resulting bounds are horrible with respect to d, they are *independent* of n. This is what defines a core set: a subset whose size only depends on the dimension and the parameter ε, with the property that it approximates the quantity under consideration (in our case all directional widths) arbitrarily well as $\varepsilon \to 0$. We have seen core sets before: In Corollary 4.2.5 we have shown the existence of a subset of P whose size only depends on ε (and *not* even on d) such that its smallest enclosing ball is a $(1+\varepsilon)$-approximation for the smallest enclosing ball of P.

For directional widths, the size of a core set depends on d, and it is known that this must be the case. The bound we prove here is not best possible, though.

Proof. (i) We assume that $2M/\varepsilon$ is integer, otherwise we round up to the next larger integer (this explains the ceiling in the statement of the theorem). We tile the bounding cuboid C of P obtained from `MinCuboid_Approx`(P, d) with

$$\left(\frac{2M}{\varepsilon}\right)^d$$

scaled-down copies of C which we call *cells* (the scaling factor is $\varepsilon/2M$). A *column* in the resulting tiling is any set of $2M/\varepsilon$ cells that have the same projection onto the hyperplane orthogonal to the vector v_d of the cuboid C. There are $(2M/\varepsilon)^{d-1}$ columns, see Figure 6.2 (left).

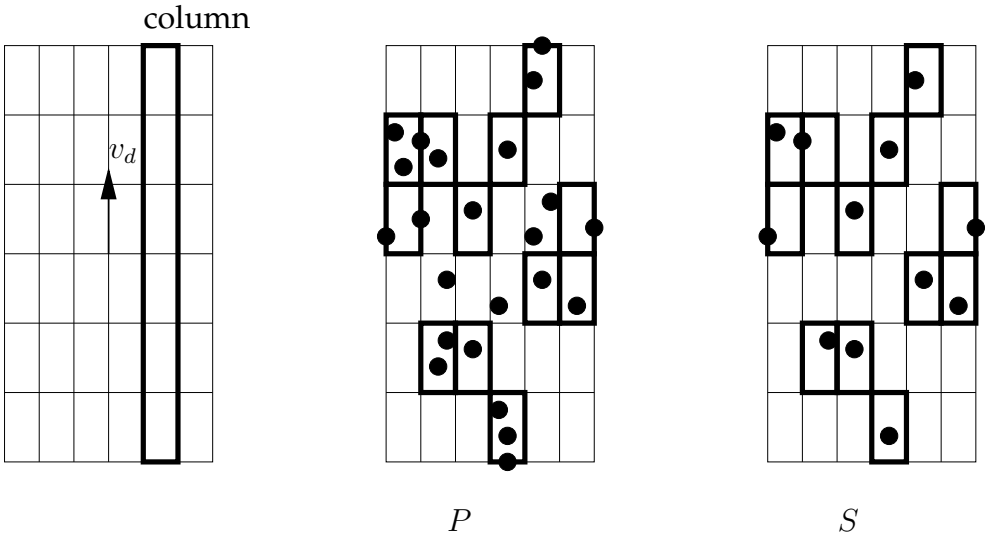

Figure 6.2: Constructing coresets for directional width

Within each column, we now find the highest and the lowest cell which contains points from P, see Figure 6.2 (middle).

Then we let S be a set of points obtained by choosing *exactly one* point from each such highest and lowest cell, see Figure 6.2 (right).

It is clear by construction that S has the required size, so we only need to show that it indeed approximates all directional widths according to the prescribed accuracy.

For this, let $\mathcal{C}$ be the union of all cells that contain points of S. Since every point $p \in P$ has a cell from $\mathcal{C}$ above it, and one below (both may coincide with the cell containing p), we know that $p \in \mathrm{conv}(\mathcal{C})$ for all $p \in P$, and this means

$$w_v(P) \leq w_v(\mathrm{conv}(\mathcal{C})) \overset{\text{Fact 6.0.4}}{=} w_v(\mathcal{C}), \quad \forall v \in S_{d-1}. \tag{6.8}$$

Now, because any cell c contributing to $\mathcal{C}$ contains a point of S, the width of $\mathcal{C}$ in any direction v cannot be larger than the width of S in direction v, plus *twice* the width of a single cell c in direction v, see Figure 6.3.

By definition of M, we have

$$w_v(c) = \frac{\varepsilon}{2M} w_v(C) \leq \frac{\varepsilon}{2} w_v(P),$$

and this gives

$$\begin{aligned} w_v(S) &\geq w_v(\mathcal{C}) - 2w_v(c) \\ &\overset{(6.8)}{\geq} w_v(P) - \varepsilon w_v(P) \\ &= (1-\varepsilon) w_v(P), \end{aligned}$$

as desired.

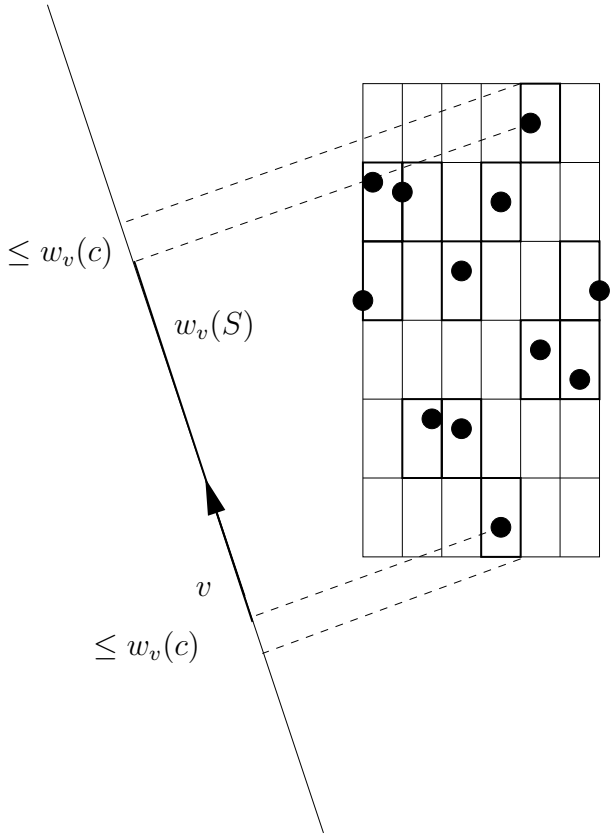

Figure 6.3: $w_v(\mathcal{C}) \leq w_v(S) + 2w_v(c)$

The construction for S we just gave can be turned into an algorithm of the required runtime. First compute in time $O(d^2n)$ the bounding cuboid of C, according to Theorem 5.1.1.

For any point $p \in P$, we have to compute a cell $c \subseteq C$ containing p.[1] Let us identify each cell with a unique d-tuple $(a_1, \ldots, a_d)$ of integers from $\{0, \ldots, 2M/\varepsilon - 1\}$, where $(a_1, \ldots, a_d)$ encodes the cell

$$\{x \in C \mid a_i \leq \frac{2M(x \cdot v_i - \ell_i)}{\varepsilon(u_i - \ell_i)} \leq a_{i+1}\}.$$

We also say that the cell has *index* $(a_1, \ldots, a_d)$. This may look complicated, but recall that by definition of u_i and ℓ_i, we have

$$\ell_i \leq x \cdot v_i \leq u_i, \quad \forall x \in C,$$

so the expression

$$\frac{2M(x \cdot v_i - \ell_i)}{\varepsilon(u_i - \ell_i)}$$

[1]Cells overlap along their boundaries, so a point may be in several cells.

runs between 0 and $2M/\varepsilon$. The possible values of a_i therefore subdivide this range into $2M/\varepsilon$ equal-sized intervals, just what we are doing with the cells.

In order to find the index of a cell containing p, we simply compute

$$a_i := \left\lfloor \frac{2M(p \cdot v_i - \ell_i)}{\varepsilon(u_i - \ell_i)} \right\rfloor, \quad i = 1, \ldots, d,$$

where we decrease this value to $2M/\varepsilon - 1$ in case the computation yields $2M/\varepsilon$. Assuming that the "floor" function can be evaluated in constant time, the index for p can be computed in $O(d^2 n)$ time, $O(dn)$ for each scalar product. (Actually, we have already computed these scalar products during the algorithm `MinCuboid_Approx`.)

We next create a hash table of size $O(n)$ for the columns; this simply means, for all n tuples $(a_1, \ldots, a_d)$ coming from the points, we insert the $d-1$-tuple $(a_1, \ldots, a_{d-1})$ into the hash table, at the same time maintaining the highest and lowest cell in each column (the ones with largest and smallest a_d), along with some point in this cell. Assuming that the hash value for each tuple can be computed in at most $O(d^2)$ time, we stay within the required time bound. From the hash table, we finally extract the core set S. □

6.3 The dual view: extent of hyperplanes

Let G be a set of nonvertical hyperplanes in $\mathbb{R}^d$, and let us assume that $g \in G$ is the set

$$g = \{x \in \mathbb{R}^d \mid x_d = \sum_{i=1}^{d-1} g_i x_i + g_d\}.$$

This is the notation of the introductory Chapter. For a point $u \in \mathbb{R}^{d-1}$ and $g \in G$, define

$$g(u) = \sum_{i=1}^{d-1} g_i u_i + g_d$$

to be g "evaluated" at u. The *extent* of G at u is defined as

$$e_u(G) = \max_{g \in G} g(u) - \min_{g \in G} g(u),$$

see Figure 6.4 for an illustration.

Definition 6.3.1 *For G a set of hyperplanes in $\mathbb{R}^d$ and $\varepsilon \geq 0$, $F \subseteq G$ is an* ε-core set *of G with respect to extents if*

$$e_u(F) \geq (1 - \varepsilon) e_u(G), \quad \forall u \in \mathbb{R}^{d-1}.$$

This is similar in spirit to ε-core sets for directional widths. We want a subset that approximates *all* possible extents up to a relative error of ε. In fact, as the section title already indicates, this is nothing but the directional width setup in disguise.

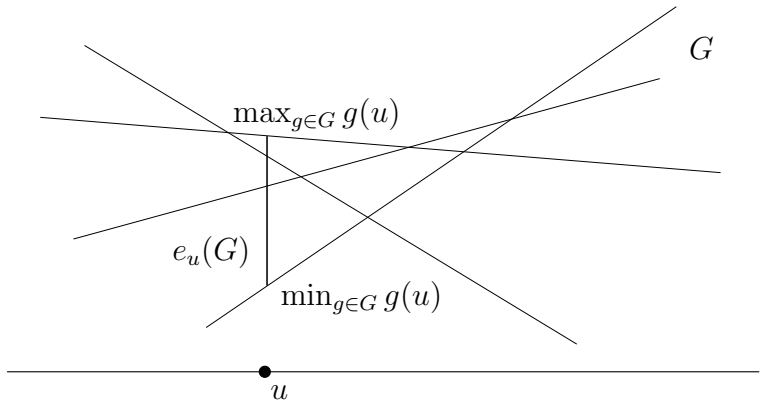

Figure 6.4: The extent of a set of nonvertical hyperplanes G at a point u

Theorem 6.3.2 *Let $P \subseteq \mathbb{R}^d$ be a finite point set and let $S \subseteq P$. Furthermore, let P^* be the set of nonvertical hyperplanes dual to P according to the definition in the introductory chapter. This means,*

$$P^* = \{p^* \mid p \in P\}, \quad p^* = \{x \in \mathbb{R}^d \mid x_d = \sum_{i=1}^{d-1} 2p_i x_i - p_d\},$$

where $p_1, \ldots, p_d$ are the coordinates of p. Then the following two statements are equivalent.

(i) S is an ε-core set of P w.r.t. directional widths, meaning that

$$w_v(S) \geq (1-\varepsilon) w_v(P), \quad \forall v \in S_{d-1}.$$

(ii) S^* is an ε-core set of P^* w.r.t. extents, meaning that

$$e_u(S^*) \geq (1-\varepsilon) e_u(P^*), \quad \forall u \in \mathbb{R}^{d-1}.$$

In particular, ε-core sets w.r.t. extents exist: simply take G, dualize it to a point set G^* and find a core set F^* with $F \subseteq G$ using Theorem 6.2.1. Then F is the desired core set of G w.r.t. extents (recall that the dual of the dual is the original object; this is needed in the argument).

Proof. Let us first assume that S is an ε-core set of P. Now fix $u \in \mathbb{R}^{d-1}$. Then there is a vector $v \in S_{d-1}$ such that

$$(u_1, \ldots, u_{d-1}, -1/2) = \lambda v$$

for suitable $\lambda > 0$. To get v, simply scale $(u_1, \ldots, u_{d-1}, -1/2)$ to unit length, see Figure 6.5.

For $p \in P$, let us evaluate the hyperplane p^* at u. We get

$$p^*(u) = \sum_{i=1}^{d-1} 2p_i u_i - p_d = 2p \cdot (u_1, \ldots, u_{d-1}, -1/2) = 2\lambda (p \cdot v). \tag{6.9}$$

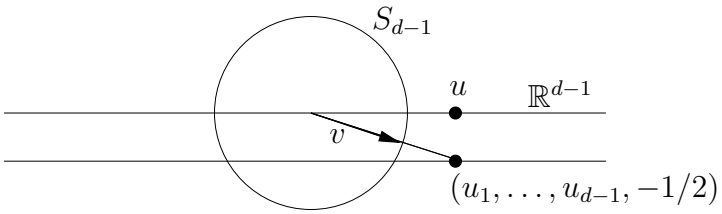

Figure 6.5: Constructing $v \in S_{d-1}$ from $u \in \mathbb{R}^{d-1}$

This is easily seen to imply (how?) that

$$\begin{aligned} e_u(P^*) &= 2\lambda w_v(P), \\ e_u(S^*) &= 2\lambda w_v(S). \end{aligned}$$

Using (i)—the fact that S is a core set—we get

$$e_u(S^*) = 2|\lambda| w_v(S) \geq 2|\lambda|(1-\varepsilon) w_v(P) = (1-\varepsilon) e_u(P^*).$$

This argument works for all u, so S^* is the desired core set and (ii) follows.

Our proof actually works in the other direction (ii)⇒(i) as well when we are able to construct for *any* given $v \in S_{d-1}$ a point $u \in \mathbb{R}^{d-1}$ such that the equation 6.9 holds for suitable $\lambda \neq 0$. Unfortunately, this cannot be done: vectors v such that $v_d = 0$ have no preimage u under the mapping illustrated in Figure 6.5.

But there is a simple trick. If $\tilde{v}$ is such an exceptional vector, we consider a sequence $v^{(i)}, i \in \mathbb{N}$ of vectors such that $v_d^{(i)} \neq 0$ and $\lim_{i\to\infty} v^{(i)} = v$. For any $v^{(i)}$ we get a preimage $u^{(i)}$ such that

$$p^*(u^{(i)}) = 2\lambda^{(i)}(p \cdot v^{(i)}), \lambda^{(i)} \neq 0.$$

To get $u^{(i)}$, choose the unique intersection point of the line through $\mathbf{0}$ and $v^{(i)}$ with the hyperplane $x_d = -1/2$. This intersection point exists if $v_d^{(i)} \neq 0$, but $\lambda^{(i)}$ might be negative.

Still, if condition (ii) of the theorem holds—S^* is a core set for P^*—we can argue as before that

$$w_{v^{(i)}}(S) = \left|\frac{1}{2\lambda^{(i)}}\right| e_{u^{(i)}}(S^*) \geq \left|\frac{1}{2\lambda^{(i)}}\right| (1-\varepsilon) e_{u^{(i)}}(P^*) = (1-\varepsilon) w_{v^{(i)}}(P).$$

Since w_v is a continuos function, this inequality also holds in the limit, for $\tilde{v}$. □

6.4 Linearization

Why did we develop the dual view of directional width, as explained in the previous section? So far it seems that we didn't get anything new. The new thing is this: the core set algorithm from Section 6.2 can also be used to find a coreset of a set of *polynomials* with respect to extents, see Figure 6.6.

Before we describe how this is done, let us consider an application.

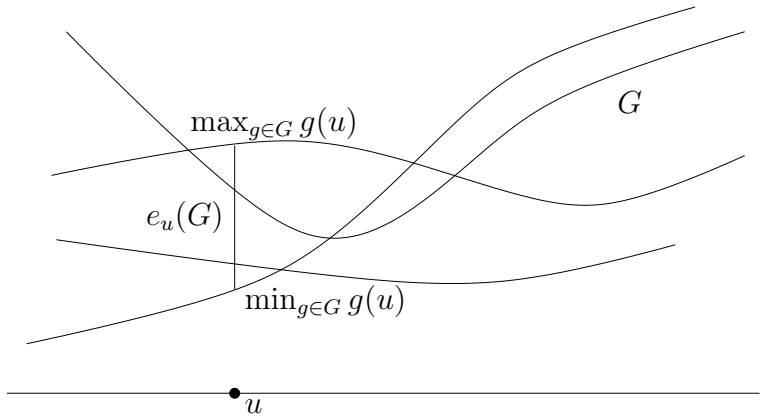

Figure 6.6: The extent of a set G of polynomials over $\mathbb{R}^{d-1}$ at a point u

6.4.1 Core sets for directional width of moving points

We have a set P of *moving* points. Formally, a moving point is a function $p : \mathbb{R} \to \mathbb{R}^d$ such that $p(t)$ is the position of p at time t. If each coordinate of $p(t)$ is a polynomial in t, we speak about *algebraic* motion.

For example, we may have $p(t) = a + tb$ for fixed $a, b \in \mathbb{R}^d$, in which case all the polynomials are linear, and we have *linear* motion.

Now consider the problem of finding an ε-core set for P with respect to directional widths. The width of P in direction v at time t is defined as

$$w_{v,t}(P) = \max_{p\in P} p(t) \cdot v - \min_{p\in P} p(t) \cdot v.$$

A set $S \subseteq P$ is an ε-core set if

$$w_{v,t}(S) \geq (1-\varepsilon) w_{v,t}(P), \quad \forall v \in S_{d-1}, t \in \mathbb{R}. \tag{6.10}$$

In order to obtain a picture as in Figure 6.6 for this problem, let us define functions $f_p : \mathbb{R}^{d+1} \to \mathbb{R}$ by

$$f_p(v,t) = p(t) \cdot v, \quad p \in P.$$

In case of algebraic motion, the coordinates of $p(t)$ are polynomials in t; the coordinates of v are (linear) polynomials in the variables $v_1, \ldots, v_d$, so all f_p are polynomials. Let G be the set of all f_p.

Assume we can find an ε-core set $F \subseteq G$ w.r.t. extents for the family of polynomials G, meaning that

$$e_{v,t}(F) \geq (1-\varepsilon) e_{v,t}(G), \quad \forall (v,t) \in \mathbb{R}^{d+1},$$

where we define the extent as

$$e_{v,t}(G) = \max_{f_p\in G} f_p(v,t) - \min_{f_p\in G} f_p(v,t) = w_{v,t}(P),$$

as we did it before for a set of hyperplanes. Then, by definition of the f_p, the set $S := \{p \mid f_p \in F\}$ satisfies (6.10) and is therefore the desired core set.

It's a pretty strong core set: it approximates *all* directional widths of the moving point set at *any* time.

Here is the theorem that lets us reduce the extent problem for polynomials to the extent problem for hyperplanes in some higherdimensional space.

Theorem 6.4.1 *Let G be a set of n polynomials in $d-1$ variables, and let k be the number of different nonconstant monomials over all members of G. Then the problem of finding an ε-core set for G w.r.t. extents can be reduced to the problem of finding an ε-core set of a set of n hyperplanes in $\mathbb{R}^{k+1}$, w.r.t. extents.*

Before we prove this, let us discuss what this means. Recall that a monomial is simply a product of variables. For example, the polynomial $x^2 + y^2 - 2xy$ is built from three monomials, x^2, y^2 and xy. If we are dealing with linear polynomials

$$\sum_{i=1}^{d-1} g_i x_i + g_d,$$

(this is the setup of hyperplane extent), there are d monomials $x_1, \ldots, x_{d-1}, 1$, but only $d-1$ of them are nonconstant. In this case, we get $k = d$,and the statement of the theorem is trivial.

Let us next consider our motivating example of core sets for moving points. Assuming linear motion, the polynomials are of the form

$$f_p(v,t) = (a + bt) \cdot v = \sum_{i=1}^{d} a_i v_i + \sum_{i=1}^{d} b_i t v_i,$$

so we have a total of $2d$ monomials $v_1, \ldots, v_d, tv_1, \ldots, tv_d$ in all the f_p. In this case, the theorem holds with $k = 2d$.

Proof. Let us denote the k nonconstant monomials by $\varphi_1, \ldots, \varphi_k$. Now we consider the mapping $\Phi : \mathbb{R}^{d-1} \to \mathbb{R}^k$, given by

$$\Phi(x_1, \ldots, x_{d-1}) = (\varphi_1(x_1, \ldots, x_{d-1}), \ldots, \varphi_k(x_1, \ldots, x_{d-1})).$$

Here, $\varphi_i(x_1, \ldots, x_{d-1})$ is the evaluation of monomial φ_i at $x_1, \ldots, x_{d-1}$. For example, evaluating x^2 at $(x, y) = (2, 3)$ gives 4.

Now let $f \in G$ be any polynomial, written in the form

$$f = \sum_{i=1}^{k} \beta_i \phi_i + \beta_{k+1}.$$

Here is the important observation: evaluating the polynomial f at $(x_1, \ldots, x_{d-1}) \in \mathbb{R}^{d-1}$ yields the same result as evaluating the *hyperplane*

$$\{y \in \mathbb{R}^{k+1} \mid y_{k+1} = \sum_{i=1}^{k} \beta_i y_i + \beta_{k+1}\}$$

at $y = \Phi(x_1, \ldots, x_{d-1})$.

Having a core set of the latter set of hyperplanes w.r.t. extents then yields a core set of the same quality for the original set of polynomials. The core set for the hyperplanes typically provides much more than we need: it works for the extents at *all* points $y \in \mathbb{R}^k$, while we only use it for points of the form $y = \Phi(x_1, \ldots, x_{d-1})$. □

This technique is called *linearization*. We have already seen an example for it in the chapter on VC-dimension: when you want to know whether a point $x \in \mathbb{R}^d$ is inside, on, or outside a certain ball in $\mathbb{R}^d$, the *lifting map* has allowed us to reduce this to the question whether the *lifted point*

$$\Phi(x) = \left(x_1, \ldots, x_d, \sum_{i=1}^{d} x_i^2 \right)$$

is below, on, or above a certain hyperplane in $\mathbb{R}^{d+1}$. In this case, the linearization is more economical, because it has one coordinate for the *sum* of all monomials x_i^2. Such savings are typical, and in that respect, the bound of Theorem 6.4.1 can be improved in many cases.

Bibliography

[1] M. Bădoiu and K. L. Clarkson. Optimal core-sets for balls. submitted, 2002.

[2] G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *Journal of Algorithms*, 38:91–109, 2001.

[3] 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.

[4] F. John. Extremum problems with inequalities as subsidary conditions. *Courant Anniversary*, ??:187–204, 1948.

[5] 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.