Optimization over k-set polytopes and efficient k-set enumeration

Artur Andrzejak*
Komei Fukuda†

January 12, 1999

Abstract

We present two versions of an algorithm based on the reverse search technique for enumerating all k-sets of a point set in $\mathbb{R}^d$. The key elements include the notion of a k-set polytope and the optimization of a linear function over a k-set polytope.

1 Introduction

Let S be a set of n points in $\mathbb{R}^d$. A *k-set of S* is a set P of k points in S that can be separated from $S \backslash P$ by a hyperplane. The problem of enumerating the k-sets has many applications in computational geometry ([AW97a]), among others in computation of higher-order Voronoi diagrams ([Aur91, Mul93]), in orthogonal L_1 hyperplane fitting ([KM93]) and in halfspace range searching ([CP86, AM95]). The first output-sensitive algorithm for enumerating k-sets was given in [EW86] (for $\mathbb{R}^2$), and other such algorithms appeared in [Mul91, AM95, AMdS94].

While the above algorithms concentrate on time-efficiency and require sophisticated data structures, we present here two output-sensitive algorithms which are highly memory-efficient. They are based on the reverse search technique introduced in [AF92, AF96]. Except for being memory-efficient, the reverse search algorithms are very easy to implement since they do not depend on complicated data structures. In addition, reverse search allows parallel computation with high speed-up factors ([BMFN97]). These advantages make it possible to handle problem instances where other methods fail, as exhaustive search computations are frequently limited by memory requirements, and less by time resources.

While developing these algorithms we obtain several theoretical results related to the k-set polytopes introduced in [EVW97, AW97b]. For S as above and $k \in \{1, \dots, n-1\}$, the *k-set polytope* $\mathcal{Q}_k(S)$ is the convex hull of the set

$$X_k(S) = \{\sum_{p \in T} p | T \in \binom{S}{k}\}.$$

*Institute of Theoretical Computer Science, ETH Zürich, CH-8092 Zürich, Switzerland, artur@inf.ethz.ch

†Institute of Operations Research, ETH Zürich, CH-8092 Zürich, Switzerland, fukuda@ifor.math.ethz.ch

The most important property of these polytopes is the fact that there is a bijection between the k-sets of S and the vertices of $Q_k(S)$. In Section 2 we describe some basic properties of $Q_k(S)$ and prove that the diameter of $Q_k(S)$ is at most $\binom{n}{2}$.

In order to set up the first algorithm for enumerating k-sets we show in Section 3 how to optimize a linear function over $Q_k(S)$ and how to find a unique path from a vertex of $Q_k(S)$ to a unique optimal vertex. The key notion for this task is that of *polytopal neighbors* in $Q_k(S)$, i.e. two vertices of $Q_k(S)$ connected by an edge. We discuss how to find all polytopal neighbors of a given vertex of $Q_k(S)$ in an output-sensitive fashion by applying geometric duality and linear programming.

The second algorithm for enumerating k-sets based on reverse search is obtained in Section 4 by considering combinatorial neighbors of $Q_k(S)$. Two k-sets T_1, T_2 (or corresponding vertices of $Q_k(S)$) are called *combinatorial neighbors* if $|T_1 \cap T_2| = k-1$. While two polytopal neighbors are also combinatorial neighbors, we present an example for which the converse is not true. We give a method to determine all combinatorial neighbors of a given k-set which is simpler than the algorithm for polytopal neighbors.

In Section 5 we discuss Euclidean k-sets. An *Euclidean k-set of S* is a set P of k points in S such that there is a hyperplane which has the points in P on one side and the points in $S \backslash P$ together with the origin on the other side. The Euclidean k-sets correspond to certain cells in an arrangement of hyperplanes obtained by geometric duality from S. They can be used to solve the problem of finding the maximum feasible subsystem of linear relations (MAX FLS, see [AK95]). We show that in $\mathbb{R}^2$ the 1-skeleton of $Q_k(S)$ restricted to vertices corresponding to the Euclidean k-sets is not always connected. As a consequence, obtaining an output-sensitive algorithm based on reverse search for enumerating the Euclidean k-sets remains an open question.

The following definitions are used throughout this paper. Let S denote a set of n points. We always assume that S is in *general position*, i.e. no $i+1$ points are on a common $(i-1)$-flat for $i = 1, \ldots, d$. If h is an oriented hyperplane in $\mathbb{R}^d$, then h^+ denotes the (open) halfspace on the positive side of h and h^- the (open) halfspace on the negative side of h. We say that a hyperplane h *separates* the sets of points A and B, if the points in A are contained in one side of h and the points in B are contained in the other side of h. (Thus our notion of separation is what is sometimes called strong separation by a hyperplane). For two vectors $\mathbf{v}$ and $\mathbf{w}$, their inner product is denoted by $\mathbf{v} \cdot \mathbf{w}$. We will denote by $\mathrm{lp}(l, m)$ the time needed for solving a linear inequality system with l variables and m inequalities. For the complexity of solving a linear inequality system, see [GJ97].

2 The k-set polytope

2.1 Basic properties

Assume that $k \in \{1, \ldots, n-1\}$. For a $T \in \binom{S}{k}$ we denote by $\mathbf{v}(T)$ the sum $\sum_{p \in T} p$ (an element of $X_k(S)$) and for $v \in X_k(S)$ we denote by $\mathbf{T}(v)$ the unique set $T \in \binom{S}{k}$ such that $\sum_{p \in T} p = v$. For $T^* \in \binom{S}{k}$ and a vector $\mathbf{c}$ of $\mathbb{R}^d$ we say that T^* *maximizes* $\mathbf{c} \cdot \mathbf{v}(T)$ if

$$\mathbf{c} \cdot \mathbf{v}(T^*) = \max_{T \in \binom{S}{|T^*|}} \mathbf{c} \cdot \mathbf{v}(T).$$

The following lemma and theorem are given in [AW97b] in a slightly different form.

Lemma 1 *Let* $\mathbf{c}$ *be a vector of* $\mathbb{R}^d$ *and* $k \in \{1, \ldots, n-1\}$.

(a) *A unique set* $T^* \in \binom{S}{k}$ *maximizes* $\mathbf{c} \cdot \mathbf{v}(T)$ *if and only if* T^* *is a* k*-set and* $\mathbf{c}$ *is a normal vector of a hyperplane which separates* T^* *and* $S \backslash T^*$.

(b) $T^* \in \binom{S}{k}$ *maximizes* $\mathbf{c} \cdot \mathbf{v}(T)$ *if and only if* $\mathbf{v}(T^*)$ *is a vertex of* $\mathcal{Q}_k(S)$ *supported by a hyperplane with normal vector* $-\mathbf{c}$.

(c) *Assume that* T^* *maximizes* $\mathbf{c} \cdot \mathbf{v}(T)$ *and let* h *be an oriented hyperplane with normal vector* $\mathbf{c}$ *such that* $T^* \subseteq h \cup h^+$, $S \backslash T^* \subseteq h \cup h^-$ *and* $T^* \cap h \neq \emptyset$ *(note that* h *may contain points of* $S \backslash T^*$*). Then the sets*

$$T_1 = (T^* \cap h^+) \cup T_2 \quad \textit{for} \quad T_2 \in \binom{S \cap h}{|T^* \cap h|}$$

are exactly the sets in $\binom{S}{k}$ *which maximize* $\mathbf{c} \cdot \mathbf{v}(T)$.

Proof: (a): Assume that $T^* \in \binom{S}{k}$ is a k-set and $\mathbf{c}$ is a normal vector of a hyperplane h separating T^* and $S \backslash T^*$, with $T^* \subseteq h^+$. Consider a hyperplane $h_1 : \mathbf{c} \cdot \mathbf{x} = \alpha$, $\alpha \in \mathbb{R}$, parallel to h and with $S \subseteq h_1^+$. Then T^* is the unique set of k points which are furthest to h_1 and so the sum

$$\sum_{p \in T^*} (\mathbf{c} \cdot \mathbf{p} - \alpha)$$

is maximal among all $T \in \binom{S}{k}$. Therefore, $\mathbf{c} \cdot \mathbf{v}(T^*)$ is maximal. The converse follows analogously.

(b): For a hyperplane h_1 with normal vector $\mathbf{c}$ which has $Q_k(S)$ on its positive side the distance between a point $v \in X_k(S)$ and h_1 is proportional to $\mathbf{c} \cdot v$. Therefore, if T^* maximizes $\mathbf{c} \cdot \mathbf{v}(T)$, there is a hyperplane h_1 with normal vector $\mathbf{c}$, $\mathbf{v}(T^*) \subseteq h_1^+$ and $\mathbf{v}(T^*)$ being a furthest point to h_1 in $X_k(S)$. It follows that $\mathbf{v}(T^*)$ is in the convex hull of $X_k(S)$ and that there is a hyperplane parallel to h_1 supporting $\mathbf{v}(T^*)$. The converse follows analogously.

(c): The statement follows from the fact that the value of $\mathbf{c} \cdot \mathbf{v}((T^* \cap h^+) \cup T_2)$ is the same for every $T_2 \in \binom{S \cap h}{|T^* \cap h|}$. □

An oriented hyperplane h is called an (i, j)*-plane of* S if h contains exactly i points of S and has exactly j points of S on its positive side. We have then

Theorem 2 **(a)** *For* $k \in \{1, \ldots, n-1\}$, $T \in \binom{S}{k}$ *is a* k*-set of* S *if and only if* $\mathbf{v}(T)$ *is a vertex of* $\mathcal{Q}_k(S)$.

(b) *Let* h *be an* (j_1, j_2)*-plane of* S *with normal vector* $\mathbf{c}$*, say, for some integers* $j_1 \geq 1$, $j_2 \geq 0$. *Then for each* $j_3 = 1, \ldots, j_1$ *there is a hyperplane* h_1 *with normal vector* $\mathbf{c}$ *which contains a face* F *of* $\mathcal{Q}_{j_2+j_3}(S)$ *with exactly* $\binom{j_1}{j_3}$ *vertices. (We say that* h *induces* F *in* $\mathcal{Q}_{j_2+j_3}(S)$*).*

Furthermore, the vertices of F *are exactly the points*

$$\mathbf{v}(T_1 \cup (S \cap h^+))$$

for all $T_1 \in \binom{S \cap h}{j_3}$.

(c) *For $k \in \{1, \dots, n-1\}$ and a face F of $\mathcal{Q}_k(S)$, there is a (j_1, j_2)-plane of S which induces F for some $j_1 \geq 1$ and $j_2 \geq 0$.*

Proof: (a): It follows by Lemma 1(a) and (b).
(b): Fix $j_3 \in \{1, \dots, j_1\}$. In a way similar to the proof of Lemma 1(a) one can show that there is $T^* \in \binom{S}{j_2+j_3}$ which maximizes $\mathbf{c}\cdot\mathbf{v}(T)$. By Lemma 1(c) exactly the elements of $\binom{S}{j_2+j_3}$ of the form

$$T_2 = (S \cap h^+) \cup T_1 \quad \text{for} \quad T_1 \in \binom{S \cap h}{j_3}$$

maximize $\mathbf{c}\cdot\mathbf{v}(T)$. Then by (b) of the same lemma and the general position assumption the points in the set

$$A = \{\mathbf{v}((S \cap h^+) \cup T_1) | T_1 \in \binom{S \cap h}{j_3}\}$$

are pairwise different vertices of $\mathcal{Q}_{j_2+j_3}(S)$. It is not hard to see that the vertices in A lie in a common hyperplane h_1 with normal vector $\mathbf{c}$ such that $h_1 \cap X_{j_2+j_3}(S) = A$ and $h_1^+ \cap X_{j_2+j_3}(S) = \emptyset$. Furthermore, each set $T_2 \in \binom{S}{j_2+j_3}$ such that $\mathbf{v}(T_2) \in A$ is a (j_2+j_3)-set of S because of the general position assumption. Then by (a) $\mathbf{v}(T_2)$ must be an extreme point in A. Obviously A is the set of vertices of a face of $\mathcal{Q}_{j_2+j_3}(S)$ and $|A| = \binom{j_1}{j_3}$.
(c): Let A be the set of vertices of a face F of $Q_k(S)$ and h_1 a hyperplane with normal vector $\mathbf{c}$ containing F. By Lemma 1(b) and arguments analogous to the proof of Lemma 1(a) there is a hyperplane h_2 parallel to h_1 such that for each T^* with $\mathbf{v}(T^*)$ we have $T^* \subseteq h \cup h^+$, $T^* \cap h^- = \emptyset$, $T^* \cap h \neq \emptyset$ and T^* maximizes $\mathbf{c}\cdot\mathbf{v}(T)$. Obviously h_2 is an (j_1, j_2)-plane for some integers $j_1 \geq 1$ and $j_2 \geq 0$. Applying Lemma 1(c) to a set T^* with $\mathbf{v}(T^*) \in A$ and to h_2 we see that the sets in $\binom{S}{k}$ maximizing $\mathbf{c}\cdot\mathbf{v}(T)$ are exactly the sets which correspond to the vertices of a face F' of $Q_k(S)$ induced by h_2 (by Theorem 2(b)). By Lemma 1(b) exactly the sets T^* with $\mathbf{v}(T^*) \in A$ maximize $\mathbf{c}\cdot\mathbf{v}(T)$ and so we have $F = F'$. □

We call two vertices of $Q_k(S)$ *polytopal neighbors* if they are connected by an edge in $Q_k(S)$.

Corollary 3 *Assume that $k \in \{1, \dots, n-1\}$ and $T_1, T_2 \in \binom{S}{k}$, $T_1 \neq T_2$.*

(a) *The points $\mathbf{v}(T_1)$ and $\mathbf{v}(T_2)$ are polytopal neighbors if and only if there is a $(2, k-1)$-plane h with $h^+ \cap S = T_1 \cap T_2$.*

(b) *If the points $\mathbf{v}(T_1)$ and $\mathbf{v}(T_2)$ are polytopal neighbors, then $|T_1 \cap T_2| = k-1$, and $T_1 \cup T_2$ is a $(k+1)$-set.*

(c) *A vertex v_1 of $Q_k(S)$ has at most $k(n-k)$ polytopal neighbors.*

Proof: (a): If $\mathbf{v}(T_1)$ and $\mathbf{v}(T_2)$ are connected by an edge F in $Q_k(S)$, then by Theorem 2(c) there is a (j_1, j_2)-plane h of S which induces F. By (b) of the same theorem we have $j_1 = 2$, $j_3 = 1$ and the vertices of the edge are exactly the points $\mathbf{v}(T_3 \cup \{p_1\})$, $\mathbf{v}(T_3 \cup \{p_2\})$, where $T_3 = S \cap h^+$, $T_1 \backslash T_3 = \{p_1\}$ and $T_2 \backslash T_3 = \{p_2\}$. The converse follows by Theorem 2(b).
(b) Follows from (a).
(c) To obtain a polytopal neighbor v_2 of v_1, we have to remove a point from $\mathbf{T}(v_1)$ and add a point from $S \backslash \mathbf{T}(v_1)$, since $|\mathbf{T}(v_1) \cap \mathbf{T}(v_2)| = k-1$ by (b). □

2.2 The diameter of the k-set polytope

If two vertices $\mathbf{v}(T_1)$, $\mathbf{v}(T_2)$ are polytopal neighbors in $Q_k(S)$, then by Corollary 3 we have

$$\mathbf{v}(T_1) - \mathbf{v}(T_2) = p_1 - p_2$$

for some $p_1, p_2 \in S$. It follows that the k-set polytope has at most $\binom{n}{2}$ edge directions. By the following result of parametric linear programming we can show that the diameter of $Q_k(S)$ is at most $\binom{n}{2}$:

Lemma 4 *For a polytope Q and two vertices v_1, v_2, there exists a path from v_1 to v_2 which uses each edge direction at most once.*

Proof: Let $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^d$, $\mathbf{c}_1 \neq \mathbf{c}_2$ be the normal vectors of supporting hyperplanes at v_1, v_2, respectively. Consider the parametric function

$$h(\theta, \mathbf{x}) = (\theta \mathbf{c}_2 + (1-\theta)\mathbf{c}_1)^T \mathbf{x}.$$

It is not hard to see that there is a hyperplane with normal vector $(\theta \mathbf{c}_2 + (1-\theta)\mathbf{c}_1)$ supporting a vertex or an edge of $Q_k(S)$, and that this hyperplane "follows" a path from v_1 to v_2, while θ goes from 0 to 1. Simultaneously the expression in parenthesis takes no two equal values (since it is a convex hull of $\mathbf{c}_1$ and $\mathbf{c}_2$). Therefore each edge on the path from v_1 to v_2 has a different direction. $\square$

3 Optimization over the k-set polytope

3.1 The k-cells

We introduce the concept of a k-cell, which is a notion dual to a k-set. Let us identify $\mathcal{U} := \mathbb{R}^d$ with a hyperplane in $\mathbb{R}^{d+1}$ with equation $x_{d+1} = 1$. Let S^d be the unit sphere with center in the origin. Each point q in S^d or in $\mathcal{U}$ determines an oriented hyperplane $h(q)$ which contains the origin $\mathbf{o}$ and has normal vector $(q - \mathbf{o})$. We denote the sphere which is an intersection of S^d with $h(q)$ by $h_{sp}(q)$. The intersection of $h(q)$ with $\mathcal{U}$ is denoted as $h_{\mathcal{U}}(q)$. Conversely, a hyperplane h of the central arrangement with normal vector $\mathbf{n}$ (or a corresponding sphere in $\mathcal{A}_{sp}$ or a corresponding hyperplane h in $\mathcal{U}$) determines an oriented line l parallel to $\mathbf{n}$ which goes through the origin. We denote the intersection of l with S^d by $p_{sp}(h)$ and the intersection of l with $\mathcal{U}$ by $p_{\mathcal{U}}(h)$. The *horizon of S^d* is the intersection of S^d with the hyperplane $x_{d+1} = 0$.

The introduced mappings give a well-known geometric duality between points and hyperplanes. It has the following property of incidence and order preservation (see [Ede87]).

- If the intersection of two spheres h_1, h_2 of S^d contain a point p, then $h_{sp}(p)$ contains $p_{sp}(h_1)$ and $p_{sp}(h_2)$. Conversely, if two points p_1, p_2 in S^d determine a sphere h, then $h_{sp}(p_1)$ and $h_{sp}(p_2)$ intersect in a point $p_{sp}(h)$.

- If a point p is on a positive side of a sphere h in S^d, then $p_{sp}(h)$ is on the positive side of $h_{sp}(p)$.

The stated duality is also valid for points and hyperplanes in $\mathcal{U}$ (and can be also "mixed" between S^d and $\mathcal{U}$).

The spheres $h_{sp}(p)$, $p \in S$ create a spherical arrangement $\mathcal{A}_{sp}$ on S^d. The cell of $\mathcal{A}_{sp}$ containing the "north pole", i.e. the point $(0, \ldots, 0, 1)$ is called the *base cell* B. It is not hard to see that B is contained on the positive side of $h_{sp}(q)$ for each $p \in S$. For a cell c of $\mathcal{A}_{sp}$, we call a ridge r (a $(d-1)$-dimensional face of $\mathcal{A}_{sp}$) of c a *horizon ridge*, if it is an intersection of spheres h_1, h_2 of $\mathcal{A}_{sp}$ such that h_1, h_2 bound c, c lies on the positive side of h_1 and simultaneously c lies on the negative side of h_2. We say that a hyperplane h bounding a cell c is *visible* (for c) if h has c on its negative side, otherwise h is called *invisible* (for c).

For $k \in \{0, \ldots, n-1\}$, a cell c is called a *k-cell* if the shortest path from the relative interior of the base cell to the relative interior of c traverses exactly k spheres of $\mathcal{A}_{sp}$. The duality gives us the following relation between k-cells and k-sets. If q is a point in the relative interior of a k-cell c for some $k \in \{1, \ldots, n-1\}$, then $h_{\mathcal{U}}(q)$ is a line separating a k-set T and $S \backslash T$ in $\mathcal{U}$. Furthermore, T lies on the positive side of $h_{\mathcal{U}}(q)$. The hyperplanes $h(p)$, $p \in S$ which are invisible for c correspond exactly to the hyperplanes $h_{\mathcal{U}}(p)$, $p \in T$. We will denote by $c(T)$ the k-cell corresponding to the k-set T. In this way we obtain a one-to-one mapping between the k-sets and the k-cells for $k \in \{1, \ldots, n-1\}$.

3.2 Determining polytopal neighbors

Recall that two vertices of $Q_k(S)$ are called polytopal neighbors if they are connected by an edge in $Q_k(S)$. The corresponding k-sets and the corresponding k-cells are also called *polytopal neighbors*. The following lemma suggests how to determine the polytopal neighbors of a given k-cell c if the horizon ridges of c are given. Using the one-to-one mapping between the k-sets and the k-cells of the last section, we can compute all neighbors of a given vertex of $Q_k(S)$.

Lemma 5 *Assume that $k \in \{2, \ldots, n-1\}$ and T_1, T_2 are k-sets.*

(a) *The vertices $\mathbf{v}(T_1)$ and $\mathbf{v}(T_2)$ of $Q_k(S)$ are polytopal neighbors if and only if the k-cells $c(T_1)$ and $c(T_2)$ of $\mathcal{A}_{sp}$ share a horizon ridge.*

(b) *Assume that the horizon ridge r is an intersection of spheres $h_{sp}(p_1)$, $h_{sp}(p_2)$, $p_1, p_2 \in S$ such that $c(T_1)$ is on the positive side of $h_{sp}(p_1)$ and simultaneously on the negative side of $h_{sp}(p_2)$. Then $c(T_2)$ is separated from the base cell B by the hyperplanes*

$$\{h_{sp}(p) | p \in T_1 \backslash \{p_2\}\} \cup \{h_{sp}(p_1)\}.$$

Proof: (a): By Corollary 3(a) we have to show that $c(T_1)$ and $c(T_2)$ share a horizon ridge in $\mathcal{A}_{sp}$ if and only if there is a $(2, k-1)$-facet h of S with $h^+ \cap S = T_1 \cap T_2$. Assume that later is the case, and denote by p_1, p_2 the both points in $S \cap h$ with $p_1 \in T_1$, $p_2 \in T_2$. Obviously we can move a hyperplane h' from a position where it separates T_1 from $S \backslash T_1$, passing a position where h' equals h, into a position where h' separates T_2 from $S \backslash T_2$. By duality, this movement corresponds to a movement of a point q dual to h' starting in the relative interior of $c(T_1)$, passing the intersection r of the both hyperplanes $h_{sp}(p_1)$ and $h_{sp}(p_2)$, and ending in the relative interior of $c(T_2)$.

As q does not pass any other hyperplanes of $\mathcal{A}_{sp}$, the cells $c(T_1)$ and $c(T_2)$ are neighbors separated by the ridge r. By the property of the duality we have also that $c(T_1)$ ($c(T_2)$) is on the negative side (positive side) of $h_{sp}(p_1)$ and that $c(T_1)$ ($c(T_2)$) is on the positive side (negative side) of $h_{sp}(h_2)$, and so r is a horizon ridge. The converse can be shown analogously.
(b): Trivial. □

For a $k \in \{2, \ldots, n-1\}$, an arrangement $\mathcal{A}_{sp}$ and its k-cell c, let $R(c)$ be the set of horizon ridges of c, and $G(R(c))$ the incidence graph of $R(c)$ (i.e. the vertices of $G(R(c))$ are horizon ridges, and there is an edge between them if both ridges meet). First note that for $d > 2$ the graph $G(R(c))$ is connected, since $R(c)$ is the set of facets of some $(d-1)$-dimensional polytope. Thus, if a ridge $r \in R(c)$ is given, we can enumerate $R(c)$ (starting at r) by tracing the graph $G(R(c))$. The problem to be solved here is to find the neighbors of r in $R(c)$. This can be done by finding all hyperplanes bounding r in the central arrangement (we consider r itself as an $(d-1)$-dimensional polytope). As r lies in the intersection of two central hyperplanes, we can find all (central) hyperplanes bounding r by finding the non-redundant hyperplanes in (3) of

(1)	$a_1 x$	$=$	b_1	(the visible hyperplane)
(2)	$a_2 x$	$=$	b_2	(the invisible hyperplane)
(3)	$\tilde{A}x$	$\leq$	$\tilde{b}$	(the remaining halfspaces $h^+(p)$, $p \in S$).

The later problem is the redundancy removal problem (for r), which can be solved in time $\mathrm{O}(n \, \mathrm{lp}(m, d))$, where m is number of non-redundant hyperplanes in (3) ([OSS95]). Assume that $\tilde{H}$ is the set of all hyperplanes bounding a horizon ridge r of a cell c. Then a horizon ridge r' incident to r in $G(R(c))$ can be obtained by interchanging a visible hyperplane (1) with a visible hyperplane in $\tilde{H}$, or by interchanging the invisible hyperplane (2) with an invisible hyperplane in $\tilde{H}$. This is due to the fact that if c' is a cell which shares r' with c, then c' is obviously also a k-cell, as the numbers of invisible hyperplanes for both c and c' are equal.

In this way we can traverse $G(R(c))$ and find all polytopal neighbors of c. If n_c is the number of horizon ridges of a cell c, then the time required to find all neighbors of c is $\mathrm{O}(n_c \, n \, \mathrm{lp}(n, d))$. Recall that $n_c \leq k(n-k)$ by Corollary 3(c).

In $\mathbb{R}^2$, the graph $G(R(c))$ is disconnected, since the horizon ridges are points. We can find all horizon ridges of cell c by computing all lines bounding the cell. This operation takes time $\mathrm{O}(n \log n)$, since it is equivalent to computing the convex hull of points dual to the lines of $\mathcal{A}_{\mathcal{U}}$. Then we traverse the sequence of these lines. A horizon ridge is detected if among two consecutive lines one is visible and the other not.

In $\mathbb{R}^3$, the set of visibility ridges form a planar polygon Q. Therefore we can apply a simplified approach for this case. We trace the visibility ridges with simplex pivoting rule. Since the arrangement is non-degenerate, there are three edges emanating from each vertex on the boundary of a cell c. At each such vertex, the pivot step can go over the next visibility edge (i.e. ridge), over an edge which is an intersection of two planes visible for c (or two planes invisible for c), or it can go back over an already visited edge. We can find the right edge by testing whether the number of visible hyperplanes remains constant and whether we do not go back. The time for finding all visibility edges of a cell c in $\mathbb{R}^3$ is dominated by the time for the pivot step, which is $\mathrm{O}(n)$ for each ridge.

3.3 Optimization on $Q_k(S)$

Let $k \in \{1, \ldots, n\}$ be fixed. Assume that the points in S are numbered according to their lexicographical sorting. For two sets $T_1, T_2 \in \binom{S}{k}$ we say that T_1 is *lexicographically smaller* than T_2 if and only if the set of indices of the points in T_1 is lexicographically smaller than the set of indices of the points in T_2. We write $T_1 <_{lex} T_2$. An analogous definition applies to the vertices $\mathbf{v}(T_1)$, $\mathbf{v}(T_2)$ of $Q_k(S)$ if T_1, T_2 are k-sets. Obviously $\{p_1, \ldots, p_k\}$ is a k-set (the lexicographically smallest one among all k-sets). We denote the corresponding vertex of $Q_k(S)$ by w_0.

In order to find a unique path from a vertex w of $Q_k(S)$ to w_0, we determine first all polytopal neighbors of w as explained in Section 3.2. Then the successor of w on the path to w_0 is the lexicographically smallest vertex of $Q_k(S)$ among these neighbors. We repeat this step until we have reached w_0.

Note that each path found in this way terminates in w_0 since the edge graph of $Q_k(S)$ is connected and since finding a lexicographically smallest set among the polytopal neighbors of w is equivalent to minimizing a linear function, which is shown as follows. If $\mathbf{c}$ is a row vector of length d such that its i-th entry equals 2^{d-i} for $i = 1, \ldots, d$, then obviously $\mathbf{c} \cdot \mathbf{v}(T_1) < \mathbf{c} \cdot \mathbf{v}(T_2)$ if and only if $T_1 <_{lex} T_2$ for $T_1, T_2 \in \binom{S}{k}$. The lexicographically smallest polytopal neighbor of $\mathbf{T}(w)$ is the neighbor with the smallest value $\mathbf{c} \cdot \mathbf{v}(T)$, so indeed the above method optimizes $\mathbf{c}$ over $Q_k(S)$.

3.4 Reverse Search using polytopal neighbors

Assume that k, $Q_k(S)$ and w_0 are the same as in previous section and that W is the set of vertices of $Q_k(S)$. Let us describe first the local search on $Q_k(S)$ (cf. [AF96]). Let $f : W \backslash \{w_0\} \longrightarrow W$ be the function which maps a vertex $w \in W \backslash \{w_0\}$ to its successor on the unique path to w_0 in the way described in the previous section. Then we define a graph G with vertex set W whose edges are $\{w, f(w)\}$ for each $w \in W \backslash \{w_0\}$. By considerations in the previous section G is connected. The local search as described in [AF96] is then given by the triple $(G, \{w_0\}, f)$.

The adjacency oracle (A-oracle) *Adj* (see [AF96]) for the graph G can be implemented in the following way.

(A1) The vertices of G can be represented by the sets $\mathbf{T}(w)$ for $w \in W$ or by integer encoding of k-tuples of indices of the elements in $\mathbf{T}(w)$.

(A2) The integer δ is at most $k(n-k)$ by Corollary 3(c).

(A3) The adjacency list oracle $Adj(w, m)$ is implemented as follows. We compute all polytopal neighbors of w and sort them lexicographically by the indices of the points in the corresponding k-sets. Then $Adj(w, m)$ returns the m-th vertex in this sequence.

As both the function f and the adjacency oracle *Adj* require the knowledge of all polytopal neighbors of a current vertex w, we will store this information for each vertex $w \in W$ the first time it is encountered during the Reverse Search (in a similar way as described in Section 4.2). The total time for computing the neighbors information over all vertices is then

$$(\text{total \# of horizon ridges of all cells})\, \mathrm{O}(n \, \mathrm{lp}(n, d)).$$

In addition, each call of $Adj(w, m)$ takes time $O(\log n)$ and each call of f takes time $O(1)$. As a consequence,

$$O((\text{total \# of horizon ridges of all cells})\, n\, \text{lp}(n, d) + |W|\, n^2 \log n)$$

is the total time for Reverse Search using polytopal neighbors.

4 Reverse search for k-sets using combinatorial neighbors

In the following let $k \in \{2, \ldots, n-1\}$ be fixed, and assume that the points in S are numbered as described in Section 3.3. We say that two k-sets T_1, T_2 of S are *combinatorial neighbors*, if $|T_1 \cap T_2| = k - 1$. The corresponding definitions apply to vertices of $Q_k(S)$ and cells in $\mathcal{A}_{sp}$. We also say that two sets T_1 and T_2 of $\binom{S}{k}$ differ by an *(i,j)-flip* if $T_2 = (T_1 \backslash \{p_i\}) \cup \{p_j\}$ for some integers $i, j \in \{1, \ldots, n\}$, $i \neq j$. If T_1, T_2 are combinatorial neighbors, then they differ by an (i,j)-flip. If two k-sets are polytopal neighbors, then by Corollary 3(b) they are also combinatorial neighbors, but the converse is not true. This shows the example in Figure 1: we can reach the 4-cell c_2 starting from c_1 by passing the hyperplane h_1 visible for c_1 and the hyperplane h_2 invisible for c_1. (This corresponds to a $(1,2)$-flip). Obviously c_1 and c_2 are combinatorial neighbors, but since they do not share a visibility ridge, they are not polytopal neighbors (Lemma 5(a)).

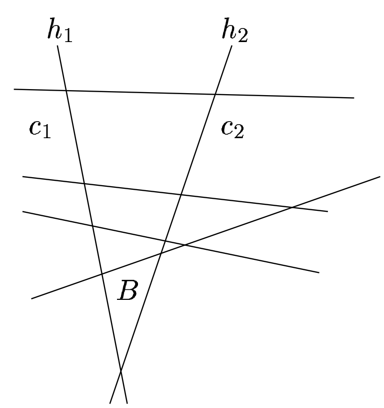

Figure 1: The 4-cells c_1, c_2 are combinatorial neighbors but not polytopal neighbors

We can test as follows whether $T' \in \binom{S}{k}$ is a k-set. The following linear inequality system is feasible if and only if T' is a k-set. The system tests whether there is a hyperplane $\mathbf{c}' \cdot \mathbf{x} = b$ which separates T' and $S \backslash T'$: find $\mathbf{c}' \in \mathbb{R}^d$ and $b \in \mathbb{R}$, such that

$$\begin{array}{lll} \mathbf{c}' \cdot p < b & \text{for all} & p \in T', \\ \mathbf{c}' \cdot p > b & \text{for all} & p \in S \backslash T'. \end{array}$$

The time for solving the above system is at most $\text{lp}(d+1, n)$.

4.1 The local search and the adjacency oracle

Using the definitions of the Section 3.3, we show first how to obtain a unique path from any vertex of $Q_k(S)$ to the unique optimal vertex w_0 of $Q_k(S)$. Let W be the set of vertices of $Q_k(S)$. We denote by f the function $f : W\backslash\{w_0\} \longrightarrow W$, which maps a vertex w to its successor w' in the unique path to w_0. The vertex $f(w)$ is computed in the following way. For each pair $(i,j), i,j \in \{1,\dots,n\}, i \neq j$, let $T_{i,j}$ be the set of $\binom{S}{k}$ which differs from $\mathbf{T}(w)$ by the (i,j)-flip, if $T_{i,j}$ exists. We denote by $\mathcal{Z}$ the family of these sets. Among the sets in $\mathcal{Z}$ we determine the lexicographically smallest one. The corresponding set $T_{i',j'}$ is then tested for being a k-set as described above. If this is the case, $\mathbf{v}(T_{i',j'})$ is the value of $f(w)$. Otherwise we discard $T_{i',j'}$ and search for the lexicographically smallest set among the remaining sets in $\mathcal{Z}$. This procedure is repeated until success. The time $t(f)$ for computing $f(w)$ is bounded by $\mathrm{O}(n^2 \mathrm{lp}(d+1,n))$, since we have to test at most n^2 elements of $\binom{S}{k}$ for being k-sets.

Since each combinatorial neighbor is also a polytopal neighbor, there exists a path from a vertex w to w_0. Furthermore, the above procedure yields a (unique) path ending in w_0, since it can be regarded as the optimization on the polytope $Q_k(S)$ by analogous arguments as in Section 3.3. The finite local search is then the triple $(G, \{w_0\}, f)$, where the graph G has W as a set of vertices and $\{\{w, f(w)\} | w \in W\backslash\{w_o\}\}$ as the set of edges.

The time $t(Adj)$ for evaluating the following adjacency oracle is clearly dominated by the time to test whether T' is a k-set, and so $t(Adj) = \mathrm{lp}(d+1,n)$.

(A1) The vertices of G can be represented by the sets $\mathbf{T}(w)$ for $w \in W$ or by integer encoding of k-tuples of indices of the elements in $\mathbf{T}(w)$.

(A2) The integer δ is at most $k(n-k)$, as each vertex $w \in W$ can have at most as many possible neighbors as possible flips.

(A3) The adjacency list oracle $Adj(w,m)$ is implemented as follows. First, we determine the lexicographically m-th pair (i,j) among the pairs $\{(i,j) | i,j \in \{1,\dots,n\}, i \neq j\}$. If the set $T' \in \binom{S}{k}$ which differs from $\mathbf{T}(w)$ by the (i,j)-flip exists, then it is tested for being a k-set. If this is the case, T' (or its integer encoding) is returned; otherwise the oracle returns 0.

The running time of the associated Reverse Search $(Adj, \delta, \{w_0\}, f)$ is $\mathrm{O}(\delta(t(Adj) + t(f))|W|) = \mathrm{O}(n^4 \mathrm{lp}(d+1,n)|W|) = \mathrm{o}(n^{4+d} \mathrm{lp}(d+1,n))$ by Corollary 2.3 of [AF96] and by the bounds on the number of k-sets for arbitrary dimensions (see [ABFK92]).

4.2 Acceleration by storing adjacency

The *trace* of a graph G is the directed subgraph $T = (W, E(f))$, where $E(f) = \{(w, f(w)) : w \in W\backslash\{w_0\}\}$. The *height* of T is the length of the longest path in T. (The height of the trace in G might exceed the diameter of $Q_k(S)$, see Section 2.2). We can reduce the total time for executing the adjacency oracle if we store the information about adjacent neighbors of a vertex w encountered for the first time during the Reverse Search. In the same step we compute and store the value of $f(w)$, which reduces the total time for computing f. This information is kept until all vertices of

G in the subtree below w are visited. In total, the number of vertices for which the neighbors information is stored simultaneously is bounded by the height of the trace.

For each vertex, we compute its set of neighbors in $Q_k(S)$ in the following way. For each pair (i, j), $i, j \in \{1, \ldots, n\}$, $i \neq j$ we test whether the set $T_{i,j} \in \binom{S}{k}$ which differs from $\mathbf{T}(w)$ by a (i, j)-flip exists and is a k-set. The time for this computation is at most $n^2 \mathrm{lp}(d + 1, n)$. Simultaneously, the value of $f(w)$ is computed. This information is then stored with each vertex in a data structure allowing access in time $\mathrm{O}(\log n)$ (e.g. binary tree). In the whole Reverse Search, the total time for computing the neighbor information for each vertex is at most $\mathrm{O}(|W| n^2 \mathrm{lp}(d + 1, n))$. Each call of $\mathrm{Adj}(w, k)$ takes time $\mathrm{O}(\log n)$, and each call of f takes time $\mathrm{O}(1)$. The total time for Reverse Search is then

$$\mathrm{O}(|W|\, n^2 (\mathrm{lp}(d + 1, n) + \log n)).$$

5 Euclidean k-sets

5.1 Basic properties

We will use the notation of Section 3.1. Given a set of points S and the corresponding spherical arrangement $\mathcal{A}_{sp}$, let us call as $\mathcal{A}_{\mathcal{U}}$ the arrangement of hyperplanes $h_{\mathcal{U}}(p)$, $p \in S$. We can think about $\mathcal{A}_{\mathcal{U}}$ as about the central projection of the part of $\mathcal{A}_{sp}$ lying in the 'northern' hemisphere of S^d onto $\mathcal{U}$. Let us call a cell of $\mathcal{A}_{sp}$ which is not completely below the horizon an *Euclidean k-cell.* Obviously there is a one-to-one mapping between the cells of $\mathcal{A}_{\mathcal{U}}$ and the Euclidean k-cells of $\mathcal{A}_{sp}$. We call a k-set of S an *Euclidean k-set* if it corresponds to an Euclidean k-cell of $\mathcal{A}_{sp}$. Under the assumption that the 0-cell of $\mathcal{A}_{sp}$ contains the point $(0, \ldots, 0, 1)$ of S^d, we can characterize the Euclidean k-sets in a following, more straightforward way.

Lemma 6 *Let $k \in \{1, \ldots, n\}$. Assume that the 0-cell of $\mathcal{A}_{sp}$ contains the point $(0, \ldots, 0, 1)$. Then a k-set $P \subseteq S$ is an Euclidean k-set if and only if P can be separated from $S \backslash P$ by a hyperplane such that the origin of $\mathcal{U}$ is on the same side as $S \backslash P$.*

Proof: First note that a central hyperplane h has the point $(0, \ldots, 0, 1)$ on its positive side if and only if the point $p_{sp}(h)$ is above the horizon of S^d. Assume that c is an Euclidean k-cell of $\mathcal{A}_{sp}$. Let q be a point of c above the horizon, and h_q the central hyperplane such that $p_{sp}(h_q) = q$. Then h_q has the origin of $\mathcal{U}$ (which coincides with the point $(0, \ldots, 0, 1)$ of $\mathbb{R}^{d+1}$) on its positive side. By the property of the duality, the k points of S corresponding to the hyperplanes separating the 0-cell and the cell c are on the negative side of h_q. The converse can be shown analogously. $\square$

If we allow a translation of the hyperplanes forming $\mathcal{A}_{\mathcal{U}}$ (or, equivalently, a translation of points in S), then the condition of Lemma 6 can be always satisfied: simply translate the hyperplanes forming $\mathcal{A}_{\mathcal{U}}$ in such a way that the origin of $\mathcal{U}$ is located in the 0-cell of $\mathcal{A}_{\mathcal{U}}$. Thus we can assume in the following, that the point $(0, \ldots, 0, 1)$ is contained in the 0-cell of $\mathcal{A}_{sp}$.

We can test whether a set $T' \in \binom{S}{k}$ is an Euclidean k-set in the analogous way as described in Section 4: T' is an Euclidean k-set if we can find $\mathbf{c}' \in \mathbb{R}^d$ and $b \in \mathbb{R}$, such that

$$\begin{array}{lll} \mathbf{c}' \cdot p < b & \text{for all} & p \in T' \\ \mathbf{c}' \cdot p > b & \text{for all} & p \in S \backslash T' \cup \{\mathbf{o}\}, \end{array}$$

i.e. if the corresponding inequality system is feasible. The running time for this test is $\text{lp}(d+1, n+1)$.

If c is a k-cell of $\mathcal{A}_{sp}$ and if we know the central hyperplanes bounding c (i.e. the central hyperplanes in $\mathbb{R}^{d+1}$ whose intersection with S^d is c), then we can test whether c is Euclidean in the following way. Let C be the cone induced by central hyperplanes bounding c. Then c is obviously an Euclidean k-cell if and only if the intersection of C and the set $\{\mathbf{x} | x_{d+1} \geq 0\}$ is not empty. This fact can be checked by a feasibility test of an inequality system (if the central hyperplanes of C or spheres bounding c in $\mathcal{A}_{sp}$ are known). The time necessary for this test is $\text{lp}(d+1, m)$, where m is the number of the central hyperplanes bounding c.

5.2 Reverse search for Euclidean k-sets

The *graph of Euclidean k-sets* of S has Euclidean k-sets as nodes, and two nodes share an edge if the corresponding Euclidean k-sets are polytopal neighbors. The Reverse Search algorithm can be applied to enumerating Euclidean k-sets only under the condition that the graph of Euclidean k-sets is connected, i.e. if one can reach each Euclidean k-set from any other Euclidean k-set by moving over Euclidean k-sets.

Unfortunately, this condition is not always fulfilled in $\mathbb{R}^2$ and it is not known to be fulfilled in higher dimensions. In Figure 2 the solid polygon represents the 4-set polytope $Q_4(S_1)$ of the points with coordinates $(-3.9, -6.8)$, $(-9.1, -4.8)$, $(3.1, -0.8)$, $(-11.1, -6.8)$, $(-1, -5)$, $(11, 9)$, $(10, 14)$. Except for the both (polytopal) neighbors of v_1 all vertices of $Q_4(S_1)$ are Euclidean. Therefore v_1 cannot be reached from any other Euclidean vertex of $Q_k(S_1)$ and so the graph of Euclidean 4-sets of S_1 is disconnected.

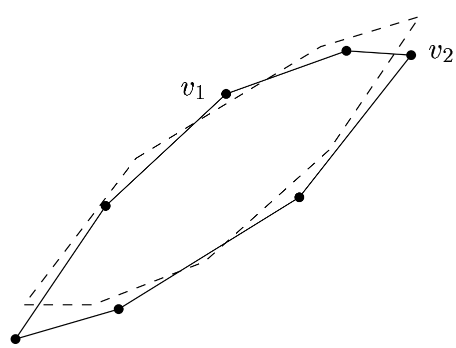

Figure 2: The graph of the Euclidean 4-sets of S_1 is not connected

We explain in the following how to obtain such counterexamples in $\mathbb{R}^2$ (and possibly also in higher dimensions). For a $k \in \{2, \ldots, n\}$, consider the polytope $Q_k(S \cup \{\mathbf{o}\})$ and the set $X_k(S \cup \{\mathbf{o}\})$ (where $\mathbf{o}$ is the origin). We have

$$X_k(S \cup \{\mathbf{o}\}) = X_k(S) \cup X_{k-1}(S),$$

since for a $T \in \binom{S\cup\{\mathbf{o}\}}{k}$, T contributes to $X_k(S)$ if $\mathbf{o} \notin T$, and T contributes to $X_{k-1}(S)$ if $\mathbf{o} \in T$. Therefore, $Q_k(S \cup \{\mathbf{o}\})$ is a convex hull of the polytopes $Q_k(S)$ and $Q_{k-1}(S)$.

Lemma 7 *A vertex v of $Q_k(S)$ is also a vertex of $Q_k(S \cup \{\mathbf{o}\})$ if and only if v corresponds to an Euclidean k-set.*

Proof: If v is Euclidean, then the set $\mathbf{T}(v)$ is a k-set in $S \cup \{\mathbf{o}\}$, too. If v is not Euclidean, then by definition the set $\mathbf{T}(v)$ is not a k-set in $S \cup \{\mathbf{o}\}$. □

Now we show that $Q_k(S)$ and $Q_{k-1}(S)$ can be translated relatively to each other by an arbitrary vector, thus allowing to 'hide' some vertices of $Q_k(S)$ by moving them into the interior of $Q_{k-1}(S)$. We translate the points in S by a vector $\mathbf{v} \in \mathbb{R}^d$ and obtain $S' = \{p + \mathbf{v} | p \in S\}$. Each point $x_1 \in X_k(S)$ becomes then $x_1 + k\mathbf{v}$, and each point $x_2 \in X_{k-1}(S)$ becomes $x_2 + (k-1)\mathbf{v}$. Therefore, $Q_k(S')$ is the polytope $Q_k(S)$ translated by $k\mathbf{v}$ and $Q_{k-1}(S')$ is the polytope $Q_{k-1}(S')$ translated by $(k-1)\mathbf{v}$. Relative to $Q_k(S)$, the polytope $Q_{k-1}(S)$ is translated by $-\mathbf{v}$.

By experimenting with some point sets and translation vectors $\mathbf{v}$, we could easily find the configuration shown in Figure 2. There, both neighbors of v_1 are inside the dashed polytope $Q_3(S_1)$, and we conclude that they are the only non-Euclidean vertices of $Q_4(S_1)$. It is also easily possible to enlarge S_1 and still keep its property of being a counterexample.

We conclude that the enumeration of Euclidean k-sets in a efficient and output-sensitive way remains an open problem. A possible yet not output-sensitive way to enumerate the Euclidean k-sets of S is to enumerate all k-sets of S and to check each output for being an Euclidean k-set by one of the tests described in Section 5.1. The enumeration of Euclidean k-sets can be used to solve the problem of finding the maximum feasible subsystem of linear relations Max FLS (see full version). It includes many interesting special cases such as the minimum feedback arc set ([AK95, GJ79]) .

Acknowledgments

The authors are grateful to Emo Welzl for helpful discussions and suggestions.

References

[ABFK92] Noga Alon, Imre Bárány, Zoltán Füredi, and Daniel J. Kleitman. Point selections and weak ϵ-nets for convex hulls. *Combinatorics, Probability and Computing*, 1992.

[AF92] David Avis and Komei Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Disc. Comput. Geom.*, 8:295–313, 1992.

[AF96] David Avis and Komei Fukuda. Reverse search for enumeration. *Disc. Applied Math.*, 65:21–46, 1996.

[AK95] Edoardo Amaldi and Viggo Kann. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoretical Computer Science*, 147:181–210, 1995.

[AM95] Pankaj K. Agarwal and Jirí Matousek. Dynamic half-space range reporting and its applications. *Algorithmica*, 13(4):325–345, April 1995.

[AMdS94] Pankaj K. Agarwal, Jiří Matoušek, Mark de Berg, and Otfried Schwarzkopf. Constructing levels in arrangements and higher order voronoi diagrams. In *Proceedings of the Tenth Annual Symposium on Computational Geometry*, pages 67–75, Stony Brook, New York, June 1994.

[Aur91] Franz Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys, Sept 1991*, 23(3), 1991.

[AW97a] Artur Andrzejak and Emo Welzl. k-sets and j-facets - A tour of discrete geometry. In preparation, 1997.

[AW97b] Artur Andrzejak and Emo Welzl. Relations between numbers of k-sets and numbers of j-facets. In preparation, 1997.

[BMFN97] Adrian Brüngger, Ambros Marzetta, Komei Fukuda, and Jurg Nievergelt. The parallel search bench zram and its applications. *Annals of Operations Research, to appear*, July 1997.

[CP86] Bernard Chazelle and Franco P. Preparata. Halfspace range search: an algorithmic application of k-sets. *Discrete Comput. Geom.*, 1:83–93, 1986.

[Ede87] Herbert Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag Berlin Heidelberg New York, 1987.

[EVW97] Herbert Edelsbrunner, Pavel Valtr, and Emo Welzl. Cutting dense point sets in half. *Discrete Comput. Geom.*, 17, 1997.

[EW86] Herbert Edelsbrunner and Emo Welzl. Constructing belts in two-dimensional arrangements with applications. *SIAM J. Comput. v.*, pages 271–284, 1986.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to NP-Completeness.* W.H. Freeman and Company, San Francisco, California, 1979.

[GJ97] Jacob E. Goodman and O'Rourke Joseph, editors. *Handbook of Discrete and Computational Geometry.* CRC Press, Boca Raton, New York, 1997.

[KM93] Nikolai M. Korneenko and Horst Martini. Hyperplane Approximations and Related Topics. Pach, János (ed.): New trends in discrete and computational geometry. Algorithms and Combinatorics v. 10, pages 135–161. Springer-Verlag, Berlin Heidelberg New York, 1993.

[Mul91] Ketan Mulmuley. On Levels in Arrangements and Voronoi Diagrams. *Discrete Comput. Geom.*, 6:307–338, 1991.

[Mul93] Ketan Mulmuley. Output sensitive and dynamic constructions of higher order Voronoi diagrams and levels in arrangements. *Journal of Computer and System Sciences*, 47(3):437–458, December 1993.

[OSS95] Thomas Ottmann, Sven Schuierer, and Subbiah Soundaralakshmi. Enumerating Extreme Points in Higher Dimensions. Proceedings of 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS 95), Munich, Germany, March 1995, LNCS 900, pages 562–570. Springer-Verlag Berlin Heidelberg New York, 1995.