The Arrangement Method for Linear Programming

Vladlen Koltun*

November 18, 2005

Abstract

We present a new family of combinatorial algorithms for linear programming called the arrangement method. The method takes a walk on the graph of the hyperplane arrangement defined by the constraints of the linear program, improving the value of the current vertex with each step. This value is derived from a representation of the arrangement as a higher-dimensional polytope. We show that graphs of arrangements have small diameters, are good expanders, and have greater connectivity than graphs of polytopes. This characterizes the arrangement method as particularly well-suited to the design of efficient combinatorial algorithms for linear programming.

1 Introduction

Background. Linear programming is perhaps the most influential computational problem of the 20th century. It can model a wide variety of situations in which limited resources need to be allocated among competing activities so as to optimize some objective. Linear programming algorithms are then used to produce such an allocation, with innumerable applications that span the sciences and engineering.

Linear programming (briefly, LP) is the task of finding a point $\mathbf{x} \in \mathbb{R}^d$ that maximizes a linear objective function $\mathbf{c} \cdot \mathbf{x}$, for $\mathbf{c} \in \mathbb{R}^d$, subject to n linear inequality constraints of the form $\mathbf{a}_i \cdot \mathbf{x} \leq b_i$, for $\mathbf{a}_i \in \mathbb{R}^d$, $1 \leq i \leq n$. It could happen that no $\mathbf{x} \in \mathbb{R}^d$ satisfies all the constraints, in which case the linear program (LP) is said to be *infeasible*. d is called the *dimension* of the LP and n its *number of constraints*. See Section 2.1 for precise definitions and [11] for further background.

Each linear inequality constraint defines a halfspace in $\mathbb{R}^d$, and the set of points that satisfies all the constraints is the intersection of these halfspaces—a convex polytope with n facets [44]. LP can thus be viewed as the geometric problem of finding a vertex of this *feasible polytope* that is extremal in the direction $\mathbf{c}$. The decision version of this problem is *linear programming feasibility*, which calls for deciding whether the feasible polytope is nonempty; see Figure 1(a).

The vertices and edges of the feasible polytope naturally form an undirected graph. Dantzig's simplex method [10]—the historically first family of algorithms for linear programming—takes a walk in this graph, such that every step improves the objective function: that is, if the method is currently at vertex $\mathbf{v}$, it can take the edge to some neighbor $\mathbf{u}$ of $\mathbf{v}$ if $\mathbf{c} \cdot \mathbf{u} > \mathbf{c} \cdot \mathbf{v}$. Eventually the method either reaches an optimal vertex or discovers that none exist. At any point, the simplex method will generally need to choose between several neighboring vertices that improve the objective function. To this end, particular implementations employ various *pivot rules*, such as "choose an improving neighbor with the highest objective function value" or "choose a random improving neighbor". Unfortunately, most known pivot rules (including almost all natural deterministic ones) can lead the simplex method to take an exponential number of steps on certain linear programs; see [3] for a unified treatment of this issue.

To this day, no variant of the simplex method is known to converge in polynomial time. This lack of success prompted a search for alternative linear programming algorithms. Indeed, a number of algorithms that run in time that is polynomial in the size of the input have been discovered, see, e.g., [12, 25, 26]. However, none of the known LP algorithms are polynomial in the RAM model of computation that measures the number of arithmetic operations performed by the algorithm. In fact, as a function of d and n, the running time of the above algorithms is unbounded, since it depends on the numerical values of the LP coefficients.

*Computer Science Department, 353 Serra Mall, Gates 464, Stanford University, Stanford, CA 94305, USA; *vladlen@cs.stanford.edu*.

Finding a *strongly polynomial* algorithm for linear programming, i.e., one whose running time is polynomial as a function of d and n, is a central problem in the mathematical and computational sciences; see, e.g., Smale [37]. Aside from its great theoretical appeal, such an algorithm may translate into practical advances as it brings about a greater understanding of the structure of linear programming.

The search for a strongly polynomial LP algorithm has spurred a number of intensively pursued research directions. Strongly polynomial algorithms have been developed for restricted special cases of the problem [9, 40, 42]. Variants of the simplex method have been shown to run in polynomial time on "random" [1, 5, 20, 36] or "slightly perturbed" [38] linear programs. Linear time algorithms have been developed when the dimension d is constant [8, 13, 31, 35]. Finally, linear programming algorithms that are subexponential in the RAM model have been developed using randomized variants of the simplex method [23, 29]. The best known running time of any LP algorithm in the RAM model is $O(d^2n + e^{O(\sqrt{d \log d})})$, obtained by combining the simplex variants of [23, 29] with the algorithms of Clarkson [8].

Coupled with the related study of LP-inspired abstract frameworks, research into the performance of randomized variants of the simplex method is perhaps the most active exploration of potential approaches to the design a strongly polynomial linear programming algorithm [16, 17, 18, 22, 23, 28, 29, 30, 34, 39]. One of the difficulties facing this technique is our understanding of graphs of polytopes, which is often lacking or discouraging. To take one prominent example, the *diameter* of a graph is the maximal graph distance (i.e., length of shortest induced path) between two vertices in the graph. In 1957, Warren M. Hirsch conjectured that the diameter of the graph of a polytope with n facets in $\mathbb{R}^d$ is at most $n - d$ [11]. After almost 50 years, this conjecture remains unresolved, with the best known (super-polynomial) upper bound of $n^{\log_2(d)+2}$ being due to Kalai and Kleitman [24]. Significantly, the diameter of this graph is also a lower bound on the worst-case performance of any simplex variant. Substantial progress on the Hirsch conjecture is thus a prerequisite to the development of a strongly polynomial simplex algorithm. Further indication of the difficulty facing the design of a strongly polynomial implementation of the simplex method can be found in the works of Matoušek [28] and Matoušek and Szabó [30].

The arrangement method. In this paper we introduce a new family of combinatorial algorithms for linear programming called the arrangement method. The arrangement method allows the design of a strongly polynomial linear programming feasibility algorithm without making progress on the Hirsch conjecture, while possessing comparable or greater flexibility than the simplex method. This is accomplished by enabling the method to take a walk on a different graph from the graph of the feasible polytope. A key part of our work is showing that the graph that the arrangement method walks on is considerably better structured than a general polytope graph.

The arrangement method solves the decision version of LP, namely linear programming feasibility. This version plays a central role in areas like machine learning [33], and Smale's problem statement, for example, is in terms of the feasibility problem [37]. We now give a brief overview of the method.

The hyperplanes $\mathbf{a}_i \cdot \mathbf{x} = b_i$, for all $1 \leq i \leq n$, partition $\mathbb{R}^d$ into maximal connected components of various dimensions. This partition is called the *arrangement* of the hyperplanes [2]. The feasible polytope, if non-empty, is one of the cells (d-dimensional components) of this arrangement. The vertices and edges (0- and 1-dimensional components) of the arrangement naturally define an undirected graph that contains the graph of the feasible polytope as an induced subgraph.

This arrangement can be embedded in a polytope in $\mathbb{R}^{d+1}$ that we call an *arrangement polytope*; in traditional terms, it is the polar of a zonotope. The graph of the arrangement polytope is, roughly speaking, isomorphic to the graph of the arrangement. (More precisely, it is isomorphic to the graph of a related spherical arrangement.) We show that we can efficiently compute a linear optimization function such that a vertex that optimizes this function on the arrangement polytope corresponds to a feasible vertex of the given linear program if the LP is feasible. Furthermore, we show that, given a vertex of the above hyperplane arrangement, we can efficiently determine the coordinates of the corresponding vertex on the arrangement polytope. This allows us to simulate any variant of the simplex method on the arrangement polytope by effectively taking a walk on the graph of the hyperplane arrangement. The flexibility of the simplex method in using a wide variety of possible pivot rules is retained. In fact, as described immediately

below, further flexibility is gained in choosing a starting vertex.

The simplex method, as presented, needs to start its walk from some vertex of the feasible polytope. However, finding such a vertex itself requires solving the feasibility problem. A common technique, employed for instance by the self-dual parametric simplex method [11] is, in effect, to add artificial dimensions and constraints to the LP and thereby manufacture an artificial feasible vertex from which the simplex method can begin. The method, having to resort to such artificial devices, has little choice concerning the start of its walk. In contrast, the arrangement method can start the walk from an arbitrary vertex of the arrangement, since any such vertex corresponds to a feasible vertex of the arrangement polytope. In fact, the starting vertex can be chosen at random, providing a second gateway for randomness, in addition to the potential randomness in the walk itself.

The ability to decide linear programming feasibility by walking on graphs of arrangements without losing flexibility in the design of the walk implies that properties of arrangement graphs are highly relevant to the study of combinatorial algorithms for linear programming, to a comparable degree with the properties of polytope graphs. We thus undertake a preliminary study, reported in Section 4, of properties of graphs of arrangements that are particularly relevant to the design and analysis of walks on these graphs. We report three main results of this nature.

First, we show that a hyperplane arrangement graph as above has diameter at most $d(n-2d+1)$. Thus for any pair of vertices there exists a walk with a small polynomial number of steps that connects them. As described above, an analogous statement is not known to be true for graphs of polytopes. Second, we show that the arrangement graph is an expander, in the sense of having conductance $\Omega((n-d)/n^3 \log n)$, implying that a *random walk* on the graph mixes in polynomial time. (We provide an example that demonstrates that polytope graphs can have inverse exponential conductance. In fact it is not known to be possible to choose a random vertex of a given polytope in polynomial time, even given some vertex to start with.) Lastly, we show that graphs of arrangements have twice greater vertex connectivity than graphs of polytopes.

The analysis of the diameter of arrangement graphs implies that the status of the Hirsch conjecture can no longer be viewed as a fundamental stumbling block on the road to a strongly polynomial linear programming feasibility algorithm. We view the arrangement method as particularly well-suited to the design of such an algorithm, and intend to further analyze the behavior of its various implementations, particularly using randomized pivot rules. We also anticipate that the analysis of properties of arrangement graphs in Section 4 is not the final word on this subject; see Section 5 for some natural open problems.

Finally, we remark that the primary concern in the development of the arrangement method has been the discovery of new theoretical tools. However, given the simplicity and versatility of the method, it seems possible that efficient implementations of the arrangement method can also yield practical gains. We plan to investigate this possibility.

Historical precedents. We remarked above on the difficulty faced by the simplex method in finding a feasible starting vertex. In an attempt to address this, Zionts in 1969 proposed the criss-cross method [45], which starts from some, not necessarily feasible vertex of the above hyperplane arrangement, and performs one of the following two kinds of steps in every iteration:

- Move from the current arrangement vertex $\mathbf{v}$ to a vertex $\mathbf{u}$ that (i) lies on a hyperplane defined by a constraint that is violated by $\mathbf{v}$, (ii) shares the other $d-1$ hyperplanes with $\mathbf{v}$, and (iii) satisfies the constraint corresponding to the other hyperplane that $\mathbf{v}$ lies on.
- Move from the current arrangement vertex $\mathbf{v}$ to a vertex $\mathbf{u}$ that (i) shares $d-1$ hyperplanes with $\mathbf{v}$, such that (ii) $\mathbf{u}$ (resp., $\mathbf{v}$) satisfies the constraint corresponding to the other hyperplane that $\mathbf{v}$ (resp., $\mathbf{u}$) lies on, and (iii) $\mathbf{c} \cdot \mathbf{u} > \mathbf{c} \cdot \mathbf{v}$.

The criss-cross method generally traces a seemingly erratic sequence of arrangement vertices that is not directly related to the arrangement graph. The method in general is not guaranteed to terminate, and it wasn't until the early 1980's that Terlaky found a strategy for choosing between the two kinds of steps in every iteration with which the criss-cross method is guaranteed to reach the optimum of the linear

program in a finite number of steps [41]. However, this strategy can still visit almost all the vertices of the arrangement, resulting in considerably poorer performance than the simplex method in the worst case [15]. Aside from the criss-cross method, other linear programming strategies can be viewed as operating on the vertices of the hyperplane arrangement defined by the constraints, including for example Seidel's algorithm [35] and the Matoušek-Sharir-Welzl algorithm [29].

On another front, decades after the first investigations of the combinatorial properties of arrangements [19, 43], arrangement graphs have begun to emerge as objects of study in their own right. Felsner et al. [14] study the chromatic number, hamiltonicity, and connectivity properties of graphs of two-dimensional arrangements, and treat the special case of Theorem 4.5 for $d = 2$. (Their proof employs a case analysis technique that does not readily extend to arbitrary dimensions.) Bose et al. [6] concentrate on hamiltonicity and "inducing substructures" in planar arrangements of lines. It is our hope that the connection between arrangement graphs and linear programming will serve to encourage this budding field.

Finally, we mention the work of Brown and Diaconis [7] (see also the references therein), who analyzed a certain random walk on the cells of a hyperplane arrangement. Although their walk is motivated by group-theoretic considerations and is not directly related to the arrangement graph, it shares the theme of analyzing random walks on arrangements with Theorem 4.3.

Organization of this paper. In the next section we lay the foundation on which the arrangement method is built. After recounting some basic notions, we construct arrangement polytopes, explore their important properties, and show that linear programming feasibility can be reduced to linear programming on an arrangement polytope. After this we can easily describe the arrangement method, complete with pseudo-code, in Section 3. We then study the properties of arrangement graphs in Section 4 and conclude with open problems in Section 5.

2 Preliminaries

2.1 On Linear Programming, Polytopes, and Arrangements

For further background on much of the material of this subsection and the next the reader is encouraged to consult [44]. Throughout this paper bold lower case symbols denote column vectors, as in $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{R}^d$, and row vectors, as in $\mathbf{a}, \mathbf{b}, \mathbf{c} \in (\mathbb{R}^d)^*$, where $(\mathbb{R}^d)^*$ denotes the vector space of all row vectors of length d with real entries. For natural m, n, the space $\mathbb{R}^{m\times n}$ is also viewed as the space of all $m \times n$ matrices with real entries. Given a (row or column) vector $\mathbf{w}$, its i-th entry is denoted by w_i. As usual, the transpose of a matrix $M \in \mathbb{R}^{m\times n}$ is denoted by $M^T \in \mathbb{R}^{n\times m}$. For a convex set $X \in \mathbb{R}^m$, its relative boundary is denoted by ∂X. For any set $X \in \mathbb{R}^m$, its convex hull is denoted by $\mathrm{CH}(X)$. For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$, we let $[\mathbf{x}, \mathbf{y}]$ denote the straight line segment $[\mathbf{x}, \mathbf{y}] := \{\alpha\mathbf{x} + (1-\alpha)\mathbf{y} : 0 \le \alpha \le 1\}$.

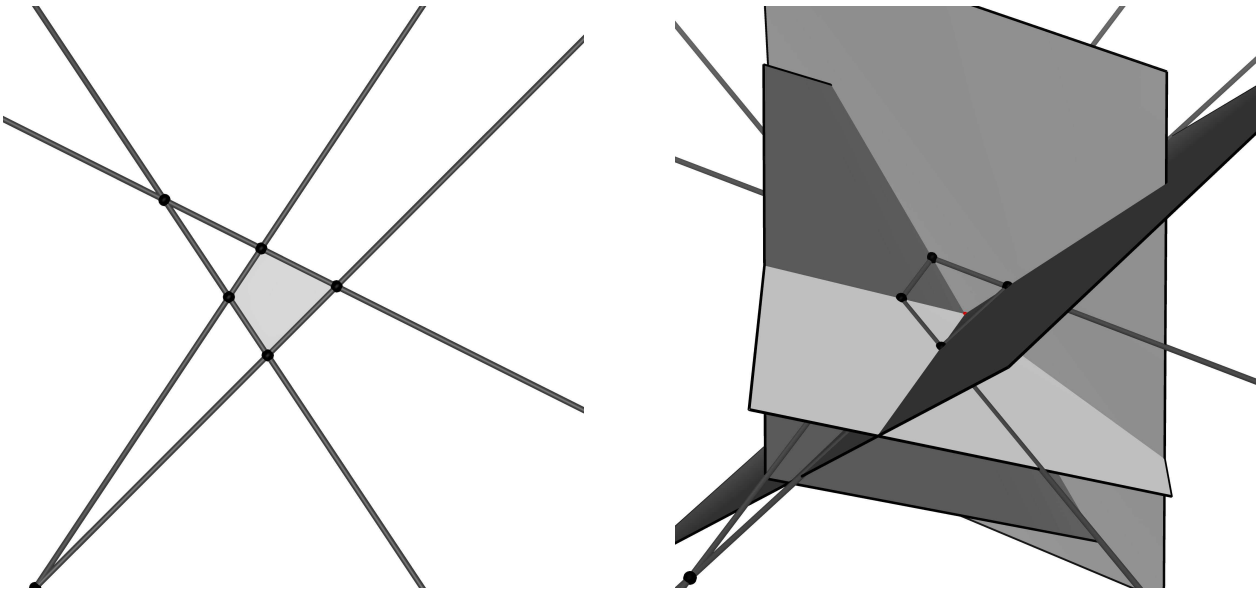
Given $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n \in (\mathbb{R}^d)^*$ and $\mathbf{b} \in \mathbb{R}^n$, the corresponding linear programming feasibility (in short, feasibility) problem is to find $\mathbf{x} \in \mathbb{R}^d$ subject to $\mathbf{a}_i\mathbf{x} \le b_i$ for all $1 \le i \le n$. Specifically, let $M = ((\mathbf{a}_1)^T, (\mathbf{a}_2)^T, \ldots, (\mathbf{a}_n)^T)^T \in \mathbb{R}^{n\times d}$ denote the matrix whose rows are $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$, and consider the *feasible polytope*

$$F := \left\{\mathbf{x} \in \mathbb{R}^d : M\mathbf{x} \le \mathbf{b}\right\}.$$

The feasibility problem is to either find some $\mathbf{x} \in F$ or correctly conclude that $F = \emptyset$ (in which case the problem is said to be *infeasible*). We call d the *dimension* of the problem, and n its *number of constraints*. For each $\mathbf{a}_i, b_i$ above we define the hyperplane

$$H_i := \left\{\mathbf{x} \in \mathbb{R}^d : \mathbf{a}_i\mathbf{x} = b_i\right\}$$

and let $\mathcal{H} := \{H_1, H_2, \ldots, H_n\}$. The *arrangement* $\mathcal{A}_\mathcal{H}$ of $\mathcal{H}$ is the subdivision of $\mathbb{R}^d$ into maximal connected open sets (called *faces*), each of which is contained in the relative interior of a fixed (possibly empty) subset of $\mathcal{H}$. The *dimension* $d(f)$ of a face f of $\mathcal{A}_\mathcal{H}$ is the dimension of the smallest affine subspace of $\mathbb{R}^d$ that contains f and its *cardinality* $c(f)$ is the number of hyperplanes of $\mathcal{H}$ that contain f. The d-dimensional



(a) (b)

Figure 1: (a) Geometric interpretation of linear programming: the feasible polytope F is a cell of the arrangement $\mathcal{A}_{\mathcal{H}}$, highlighted in the figure. (b) The arrangement $\mathcal{A}_{\mathcal{H}'}$ and the intersections $\Phi \cap H_i$, for $1 \leq i \leq n$.

(resp., 0-, 1-dimensional) faces of an arrangement are called cells (resp., vertices, edges). The *face poset* $\mathrm{PS}(\mathcal{A}_{\mathcal{H}})$ of the arrangement $\mathcal{A}_{\mathcal{H}}$ is the poset of all faces of $\mathcal{A}_{\mathcal{H}}$, partially ordered by inclusion of closures: $f \leq g$ if and only if the closure of g contains the closure of f.

To streamline the exposition we adopt the following general position assumption throughout this paper: for any face f of $\mathcal{A}_{\mathcal{H}}$, $d(f) + c(f) = d$. Most of the results below hold, with appropriate modifications, without a general position assumption. Note, however, that as Megiddo observed [32], without some general position assumption, solving any LP problem can be reduced to making a single pivot step of the simplex algorithm on a related one. In other words, when dealing with combinatorial algorithms for linear programming there are good reasons to make some general position assumption. If needed, its validity can be ensured by perturbation, for example random or symbolic.

For each $\mathbf{a}_i, b_i$, define $\mathbf{a}'_i := (\mathbf{a}_i, -b_i) \in (\mathbb{R}^{d+1})^*$. Let $M' \in \mathbb{R}^{n \times (d+1)}$ denote the matrix whose rows are $\mathbf{a}'_1, \mathbf{a}'_2, \ldots, \mathbf{a}'_n$. Let

$$H'_i := \left\{ \mathbf{x} \in \mathbb{R}^{d+1} : \mathbf{a}'_i \mathbf{x} = 0 \right\}$$

and $\mathcal{H}' := \{H'_1, H'_2, \ldots, H'_n\}$. Each H'_i is a linear subspace of $\mathbb{R}^{d+1}$, and $\mathcal{A}_{\mathcal{H}'}$, defined as above, is a *linear* hyperplane arrangement. By construction, if $\Phi := \{\mathbf{x} \in \mathbb{R}^{d+1} : x_{d+1} = 1\}$, the collection of intersections $\Phi \cap H_i$, for $1 \leq i \leq n$, is affinely isomorphic to $\mathcal{H}$; see Figure 1(b). We also identify $\mathbb{S}^d$ with the unit sphere

$$\mathcal{S} := \left\{ \mathbf{x} \in \mathbb{R}^{d+1} : \|\mathbf{x}\| = 1 \right\}$$

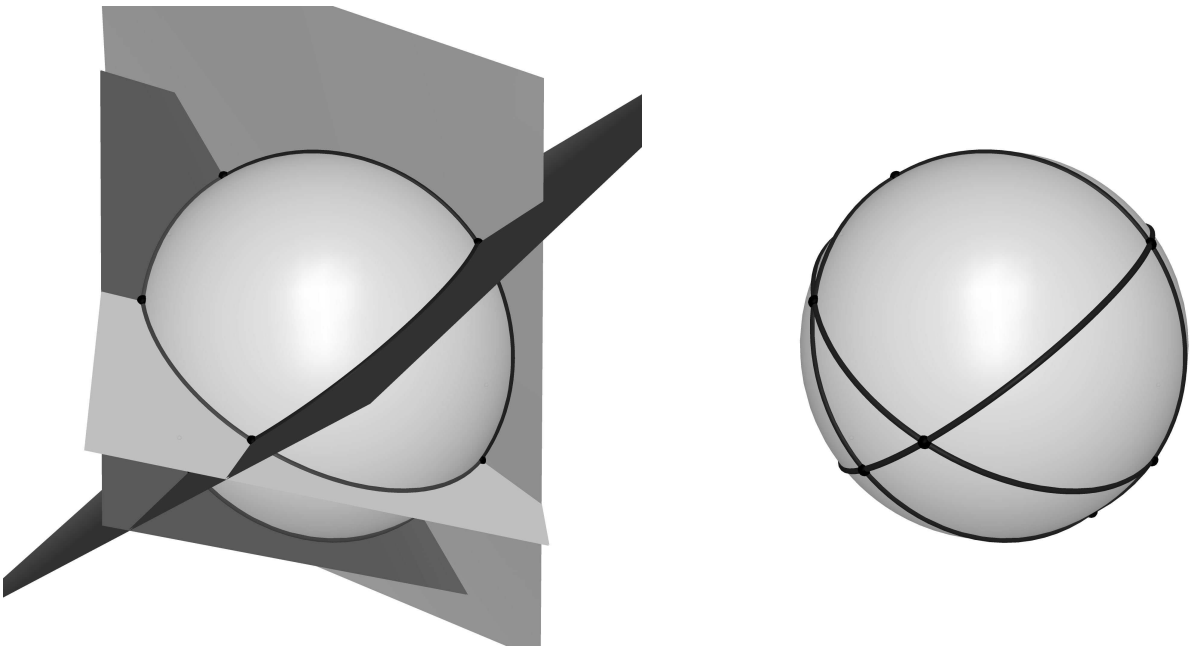
and consider $C_i := H'_i \cap \mathcal{S}$ for all i. The collection $\mathcal{C} := \{C_1, C_2, \ldots, C_n\}$ naturally yields a *spherical arrangement* $\mathcal{A}_{\mathcal{C}}$ of great spheres in $\mathbb{S}^d$; see Figure 2.

The *zonotope* generated by the vectors $\mathbf{a}'_1, \mathbf{a}'_2, \ldots, \mathbf{a}'_n$ is defined as

$$\mathcal{Z} := \left\{ \sum_{i=1}^{n} \lambda_i \mathbf{a}'_i : \lambda \in [-1, 1]^n \right\}.$$

Equivalently,

$$\mathcal{Z} = \left\{ \lambda^T M' : \lambda \in [-1, 1]^n \right\} = \mathrm{CH}\left(\left\{ \lambda^T M' : \lambda \in \{-1, 1\}^n \right\} \right).$$



(a) (b)

Figure 2: (a) The arrangement $\mathcal{A}_{\mathcal{H}'}$ and $\mathcal{S}$. (For illustration purposes the sphere has been scaled up relative to Figure 1(b).) (b) The spherical arrangement $\mathcal{A}_{\mathcal{C}}$.

$\mathcal{Z}$ is a centrally symmetric polytope around the origin $\mathbf{0}$ in $(\mathbb{R}^{d+1})^*$; see Figure 3. We define the *arrangement polytope* $\mathcal{AP}$ corresponding to $\mathcal{H}'$ as the *polar* of $\mathcal{Z}$, see Figure 4(a):

$$\mathcal{AP} := \left\{ \mathbf{x} \in \mathbb{R}^{d+1} : \lambda^T M' \mathbf{x} \leq 1 \text{ for all } \lambda \in [-1,1]^n \right\}.$$

For a polytope P in $\mathbb{R}^m$, its face poset $\mathrm{PS}(P)$, as in the case of arrangements, is the poset of all faces of P, partially ordered by inclusion of closures. The m-dimensional (resp., 0-, 1-dimensional) faces of P are called facets (resp., vertices, edges). Our convention throughout the paper is that the empty set is the face of any polytope, and so is the whole relative interior of the polytope itself; faces other than $\emptyset$ and the whole P are called *proper*. We consider faces as open sets, so the polytope is the disjoint union of its faces. It will be convenient to denote by $\mathrm{PS}'(P)$ the poset $\mathrm{PS}(P)$ minus the node that corresponds to the entire polytope.

For a convex set $S \in \mathbb{R}^m$ that does not contain the origin we define

$$\mathrm{cone}(S) := \{t\mathbf{x} : t \in \mathbb{R}, t > 0, \mathbf{x} \in S\}.$$

If $\mathbf{0} \in S$ then we define $\mathrm{cone}(S) := \mathbf{0}$. (This is somewhat nonstandard, but will be convenient in this paper.) For a polytope P, let

$$\mathrm{fan}(P) := \{\mathrm{cone}(f) : f \in \mathrm{PS}(P)\}.$$

2.2 Properties of the Arrangement Polytope

We are ready to discover the properties that make arrangement polytopes so useful.

Lemma 2.1. $\mathrm{fan}(\mathcal{AP}) = \mathcal{A}_{\mathcal{H}'}$.

Proof. See Figure 4(b). Consider any $\gamma \in \mathbb{R}^{d+1}$, $\gamma \neq \mathbf{0}$, and let e be the face of $\mathcal{A}_{\mathcal{H}'}$ that contains γ. Similarly, let g be the face of $\mathcal{AP}$ such that there exists $t > 0$ for which $t\gamma$ lies in the relative interior of g. To prove the lemma it is sufficient to show that $e = \mathrm{cone}(g)$.

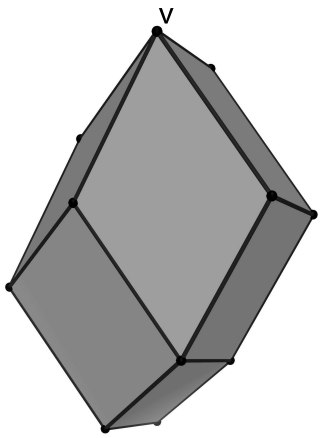


Figure 3: The zonotope $\mathcal{Z}$ constructed from the LP in Figure 1. The vertex $\mathbf{v}$ corresponds to the feasible polytope F.

Consider a face f of the zonotope $\mathcal{Z}$ that maximizes γ over $\mathcal{Z}$. That is, $\mathbf{v}\gamma \geq \mathbf{w}\gamma$ for all $\mathbf{v} \in f, \mathbf{w} \in Z$. For any $\mathbf{v} \in f$ recall that $\mathbf{v} = \sum_{i=1}^{n} \lambda_i \mathbf{a}'_i$ for some $\lambda \in [-1,1]^n$. Clearly, if $\mathbf{a}'_i\gamma < 0$ then $\lambda_i = -1$. Similarly, if $\mathbf{a}'_i\gamma > 0$ then $\lambda_i = 1$, and if $\mathbf{a}'_i\gamma = 0$ then λ_i can take any value in $[-1,1]$. Thus

$$f = \left\{ \sum_{i=1}^{n} \lambda_i \mathbf{a}'_i : \quad \lambda_i = -1 \text{ if } \mathbf{a}'_i\gamma < 0, \quad \lambda_i = 1 \text{ if } \mathbf{a}'_i\gamma > 0, \quad \lambda_i \in [-1,1] \text{ if } \mathbf{a}'_i\gamma = 0 \right\}. \tag{1}$$

Partition the indices $\{1, 2, \ldots, n\}$ into three disjoint groups $\mathcal{I}^{(1)}$, $\mathcal{I}^{(2)}$, and $\mathcal{I}^{(3)}$, such that for $i \in \mathcal{I}^{(1)}$ (resp., $i \in \mathcal{I}^{(2)}$, $i \in \mathcal{I}^{(3)}$) it holds that $\mathbf{a}'_i\gamma < 0$ (resp., $\mathbf{a}'_i\gamma > 0$, $\mathbf{a}'_i\gamma = 0$). Define

$$\begin{aligned} \Gamma_f := \big\{ \gamma \in \mathbb{R}^{d+1} : \quad & \mathbf{a}'_i\gamma < 0 \quad \text{for all } i \in \mathcal{I}^{(1)}, \\ & \mathbf{a}'_i\gamma > 0 \quad \text{for all } i \in \mathcal{I}^{(2)}, \\ & \mathbf{a}'_i\gamma = 0 \quad \text{for all } i \in \mathcal{I}^{(3)} \big\}. \end{aligned} \tag{2}$$

Γ_f is precisely the face of the arrangement $A_{\mathcal{H}'}$ that lies below the hyperplanes H'_i for $i \in \mathcal{I}^{(1)}$, above the hyperplanes H'_i for $i \in \mathcal{I}^{(2)}$, and in the relative interior of the hyperplanes H'_i for $i \in \mathcal{I}^{(3)}$. This is the face of $A_{\mathcal{H}'}$ that contains γ, and thus $\Gamma_f = e$.

On the other hand, observe that (1) and (2) together imply that Γ_f is the set of directions that are optimized on f:

$$\Gamma_f = \left\{ \gamma \in \mathbb{R}^{d+1} : \mathbf{v}\gamma \geq \mathbf{w}\gamma \text{ for all } \mathbf{v} \in f, \mathbf{w} \in \mathcal{Z} \right\}.$$

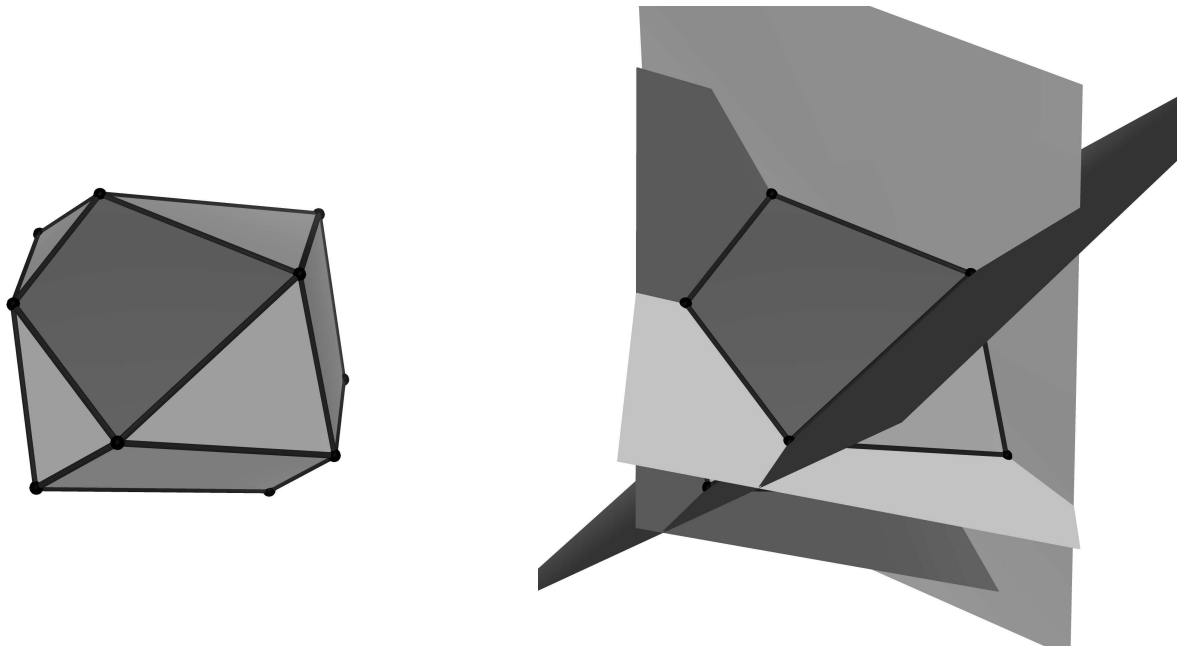
Let

$$h := \left\{ \mathbf{x} \in \mathbb{R}^{d+1} : \quad \mathbf{a}\mathbf{x} = 1 \text{ for all } \mathbf{a} \in f, \quad \mathbf{a}\mathbf{x} \leq 1 \text{ for all } \mathbf{a} \in \mathcal{Z} \right\}.$$

h is the face of $\mathcal{AP}$ that maximizes all the directions $\mathbf{a} \in f$. By definition, there is some value φ, such that for any $\mathbf{v} \in f$ and $\mathbf{w} \in \mathcal{Z}$, $\mathbf{v}\gamma = \varphi$ and $\mathbf{w}\gamma \leq \varphi$. Thus for $t = \frac{1}{\varphi}$ we have $t\gamma \in h$. Thus $h = g$. Furthermore, by the characterization of Γ_f above we have that for any $\gamma' \in \Gamma_f$ there is a φ' for which $\frac{1}{\varphi'}\gamma' \in h$. Hence $\Gamma_f \subseteq \text{cone}(h)$. Since it also holds that $h \subset \Gamma_f$, we have $e = \Gamma_f = \text{cone}(h) = \text{cone}(g)$. This concludes the proof. □

The following corollary follows immediately from Lemma 2.1.

Corollary 2.2. *The poset* $\text{PS}'(\mathcal{AP})$ *is isomorphic to the poset* $\text{PS}(\mathcal{A}_\mathcal{C})$.



(a) (b)

Figure 4: (a) The arrangement polytope $\mathcal{AP}$ constructed from the LP in Figure 1. The highlighted top face of $\mathcal{AP}$ is polar to the vertex $\mathbf{v}$ of $\mathcal{Z}$ and is projectively isomorphic to the feasible polytope F. (b) $\mathcal{AP}$ with the arrangement $\mathcal{A}_{\mathcal{H}'}$.

We can further characterize the correspondence between the faces of $\mathcal{AP}$ and of $\mathcal{A}_{\mathcal{H}}$. Given a point $\mathbf{x} \in \mathbb{R}^{d+1}$, we define the *signature* of $\mathbf{x}$ to be $\mathrm{str}(\mathbf{x}) \in \{-1, 0, 1\}^n$, such that $\mathrm{str}(\mathbf{x})_i = -1$ (resp., $\mathrm{str}(\mathbf{x})_i = 1$, $\mathrm{str}(\mathbf{x})_i = 0$) if and only if $\mathbf{a}_i'\mathbf{x} < 0$ (resp., $\mathbf{a}_i'\mathbf{x} > 0$, $\mathbf{a}_i'\mathbf{x} = 0$) for all $1 \leq i \leq n$. Abusing notation slightly we also define $\mathrm{str}(\mathbf{y}) \in \{-1, 0, 1\}^n$ for $\mathbf{y} \in \mathbb{R}^d$, such that $\mathrm{str}(\mathbf{y})_i = -1$ (resp., $\mathrm{str}(\mathbf{y})_i = 1$, $\mathrm{str}(\mathbf{y})_i = 0$) if and only if $\mathbf{a}_i\mathbf{y} < b_i$ (resp., $\mathbf{a}_i\mathbf{y} > b_i$, $\mathbf{a}_i\mathbf{y} = b_i$). Lastly, given a point $\mathbf{v} \in \partial\mathcal{Z}$, $\mathbf{v} = \sum_{i=1}^{n} \lambda_i \mathbf{a}_i'$, we define $\mathrm{str}(\mathbf{v}) \in \{-1, 0, 1\}^n$ so that $\mathrm{str}(\mathbf{v})_i = -1$ (resp., $\mathrm{str}(\mathbf{v})_i = 1$, $\mathrm{str}(\mathbf{v})_i = 0$) if and only if $\lambda_i = -1$ (resp., $\lambda_i = 1$, $\lambda_i \in (-1, 1)$). We define the signature $\mathrm{str}(f)$ of a (proper) face f of $\mathcal{A}_{\mathcal{H}}$ (resp., of $\mathcal{AP}$, of $\mathcal{Z}$) to be the common signature λ of all points in f. It is easy to see that f contains all points in $\mathbb{R}^d$ (resp., on $\partial\mathcal{AP}$, on $\partial\mathcal{Z}$) with signature λ, and the signatures of two distinct (proper) faces of $\mathcal{A}_{\mathcal{H}}$ (resp., of $\mathcal{AP}$, of $\mathcal{Z}$) are distinct.

Consider the map $\pi : \mathbb{R}^{d+1} \mapsto \mathbb{R}^d$,

$$\pi\left((x_1, x_2, \ldots, x_{d+1})\right) := \left(\frac{x_1}{x_{d+1}}, \frac{x_2}{x_{d+1}}, \ldots, \frac{x_d}{x_{d+1}}\right).$$

The map π is defined for $\mathbf{x}$ that have $x_{d+1} \neq 0$. We extend it to sets and for $\mathcal{X} \subset \mathbb{R}^{d+1}$ define $\pi(\mathcal{X}) := \{\pi(\mathbf{x}) : \mathbf{x} \in \mathcal{X}\}$. Abusing notation slightly, we also use π to denote the completely analogous mapping in the dual space $(\mathbb{R}^{d+1})^*$.

The following follows from Lemma 2.1:

Corollary 2.3. *Consider $\mathbf{x} \in \partial\mathcal{AP}$. If $x_{d+1} > 0$ then $\mathrm{str}(\pi(\mathbf{x})) = \mathrm{str}(\mathbf{x})$ and if $x_{d+1} < 0$ then $\mathrm{str}(\pi(\mathbf{x})) = -\mathrm{str}(\mathbf{x})$. Furthermore, for any $\mathbf{y} \in \mathbb{R}^d$, there are exactly two points $\mathbf{x}', \mathbf{x}'' \in \partial\mathcal{AP}$ with $\pi(\mathbf{x}') = \mathbf{y} = \pi(\mathbf{x}'')$. These points satisfy $\mathbf{x}' = -\mathbf{x}''$ and $\mathrm{str}(\mathbf{x}') = \mathrm{str}(\mathbf{y}) = -\mathrm{str}(\mathbf{x}'')$. Similarly, given a face f of $\mathcal{A}_{\mathcal{H}}$ there are precisely two proper faces g', g'' of $\mathcal{AP}$ that are (in whole or in part) mapped onto f by π, and it holds that $\mathrm{str}(g') = \mathrm{str}(f) = -\mathrm{str}(g'')$.*

We define a map $\chi : \mathbb{R}^d \mapsto \mathbb{R}^{d+1}$ so that $\chi(\mathbf{y})$ is the unique $\mathbf{x} \in \mathbb{R}^{d+1}$ for which $x_{d+1} > 0$ and $\pi(\mathbf{x}) = \mathbf{y}$. The correspondence between the faces of $\mathcal{Z}$ and of $\mathcal{AP}$ is formalized in the following lemma, whose validity can be established by similar algebraic manipulations to those employed in the proof of Lemma 2.1.

Lemma 2.4. *For every proper face f of $\mathcal{Z}$ (resp., g of $\mathcal{AP}$) there exists precisely one (proper) face g of $\mathcal{AP}$ (resp., f of $\mathcal{Z}$) that satisfies* $\mathrm{str}(g) = \mathrm{str}(f)$. *The face g (resp., f) is called the* polar *of f (resp., of g). It holds that*

(i) *the sum of the dimensions of f and of g is d.*

(ii) $\mathbf{v} \in \mathrm{cone}(f)$ *if and only if* $\mathbf{vx} \geq \mathbf{vy}$ *for all* $\mathbf{x} \in g$, $\mathbf{y} \in \mathcal{AP}$.

(iii) $\mathbf{x} \in \mathrm{cone}(g)$ *if and only if* $\mathbf{vx} \geq \mathbf{wx}$ *for all* $\mathbf{v} \in f$, $\mathbf{w} \in \mathcal{Z}$.

(iv) *for any* $\mathbf{x} \in g$, f *lies in the relative interior of the hyperplane* $\{\mathbf{v} \in (\mathbb{R}^{d+1})^* : \mathbf{vx} = 1\}$.

(v) *for any* $\mathbf{v} \in f$, g *lies in the relative interior of the hyperplane* $\{\mathbf{x} \in \mathbb{R}^{d+1} : \mathbf{vx} = 1\}$.

2.3 Reducing to LP on $\mathcal{AP}$

The arrangement method employs a reduction of the feasibility problem to the following linear program on the arrangement polytope:

$$\begin{aligned} \max \quad & \mathbf{vx}, \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{AP}, \end{aligned} \tag{3}$$

where $\mathbf{v} = -\sum_{i=1}^{n} \mathbf{a}'_i$. The following theorem describes the reduction.

Theorem 2.5. *Let* $\mathbf{x}$ *be a vertex of* $\mathcal{AP}$ *for which* $\mathbf{vx} \geq \mathbf{vy}$ *for all* $\mathbf{y} \in \mathcal{AP}$. *If* $F \neq \emptyset$ *then* $\pi(\mathbf{x}) \in F$.

Proof. Observe that the signature of any point within the relative interior of F, it such exist, is $\{-1\}^n$. By Corollary 2.3, if $F \neq \emptyset$ there exists a d-dimensional facet g of $\mathcal{AP}$ with $\mathrm{str}(g) = \{-1\}^n$. By Lemma 2.4 there is thus a vertex of $\mathcal{Z}$ whose string is $\{-1\}^n$, which implies that $\mathbf{v} = -\sum_{i=1}^{n} \mathbf{a}'_i$ is a vertex of $\mathcal{Z}$. Moreover, again by Lemma 2.4, g is the facet that maximizes $\mathbf{vy}$ over $\mathbf{y} \in \mathcal{AP}$ and lies on the hyperplane $\{\mathbf{x} \in \mathbb{R}^{d+1} : \mathbf{vx} = 1\}$. Thus, if $F \neq \emptyset$ then a vertex $\mathbf{x}$ as in the statement of the theorem lies in the closure of g and satisfies $\mathbf{vx} = 1$; in particular, $\pi(\mathbf{x}) \in F$. This proves the theorem. □

3 The Arrangement Method

Theorem 2.5 suggests that to solve linear programming feasibility problems efficiently it suffices to be able to solve linear programs on arrangement polytopes efficiently. This might sound like a mixed blessing at best. An arrangement polytope in general has an exponential number of facets, thus it does not possess an explicit description of polynomial size, and it is not clear how we can determine the coordinates of any of its vertices. How then should we go about optimizing a linear function over it?

Recall that the simplex method takes a walk on the graph of the feasible polytope. At each step of the walk it proceeds to a neighbor of the current vertex that improves the objective function. In general there is more than one such improving neighbor, and one is chosen using some pivot rule. The flexibility in choosing this rule qualifies the simplex as a "method", as opposed to a single algorithm.

The arrangement method simulates the simplex method on the arrangement polytope $\mathcal{AP}$. Corollary 2.2 implies that the graph of $\mathcal{AP}$ is isomorphic to the graph of the spherical arrangement $\mathcal{A}_{\mathcal{C}}$. In fact, the arrangement method will confine its walk to the half of $\mathcal{AP}$ that lies in the halfspace $\{\mathbf{x} \in \mathbb{R}^{d+1} : x_{d+1} > 0\}$. As the reader can verify, the graph of this part is isomorphic to the graph of the arrangement $\mathcal{A}_{\mathcal{H}}$, the hyperplane arrangement naturally defined by the constraints of the linear program. Thus the arrangement method can be viewed as taking a walk on the graph of $\mathcal{A}_{\mathcal{H}}$.

Recall from Section 1 that a thorny issue for the simplex method is where to start the walk. Since any vertex of $\mathcal{A}_{\mathcal{H}}$ corresponds to a feasible vertex of $\mathcal{AP}$, the arrangement method can start its walk from any vertex of $\mathcal{A}_{\mathcal{H}}$, for example one can be selected at random. (Note that a vertex of $\mathcal{A}_{\mathcal{H}}$ is completely specified by the d hyperplanes of $\mathcal{H}$ that contain it. In turn, any d-tuple of hyperplanes defines a unique vertex of $\mathcal{A}_{\mathcal{H}}$ by our general position assumption.)

Arrangement Method$(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n \in (\mathbb{R}^d)^*; \mathbf{b} \in \mathbb{R}^n)$

< Input the LP >

For $i := 1$ **to** n $\quad H_i := \{\mathbf{x} \in \mathbb{R}^d : \mathbf{a}_i\mathbf{x} = b_i\}$ $\quad$ **EndFor**

CurrentVertex := ChooseStartingVertex$(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n, \mathbf{b})$

< Choose a starting vertex on $\mathcal{AP}$ >

Repeat

ImprovingNeighbors := $\emptyset$

Current := π(CurrentVertex)

For $\{i :$ Current $\in H_i\}$

Line := $\bigcap_{j \neq i,\ \text{Current} \in H_j} H_j$

< Line is one the lines of $\mathcal{A}_\mathcal{H}$ incident to Current >

Vertices := $\{\text{Line} \cap H_j : \text{Line} \nsubseteq H_j\}$

< Vertices is a list of the $n - d + 1$ vertices of $\mathcal{A}_\mathcal{H}$ on Line, including Current >

SortedVertices := Sort(Vertices)

< SortedVertices is an array, numbered from 1 to $n - d + 1$, sorted by position along Line >

CurrentIndex := Index(Current, SortedVertices)

< Retrieve the index of Current in the array >

If CurrentIndex > 1 **Then**

LeftVertex := χ(SortedVertices[CurrentIndex-1])

If $\mathbf{v} \cdot$ LeftVertex $> \mathbf{v} \cdot$ CurrentVertex

Then ImprovingNeighbors := ImprovingNeighbors $\cup$ {LeftVertex} **EndIf**

EndIf

If CurrentIndex $< n - d + 1$ **Then**

RightVertex := χ(SortedVertices[CurrentIndex+1])

If $\mathbf{v} \cdot$ RightVertex $> \mathbf{v} \cdot$ CurrentVertex

Then ImprovingNeighbors := ImprovingNeighbors $\cup$ {RightVertex} **EndIf**

EndIf

< Compute the locations in $\mathbb{R}^{d+1}$ of the neighboring $\mathcal{AP}$ vertices along Line using Lemma 3.1, and if a neighbor improves the objective function $\mathbf{v}$ on $\mathcal{AP}$, add it to the list >

EndFor

If ImprovingVertices $\neq \emptyset$

Then CurrentVertex := Pivot(CurrentVertex, ImprovingVertices)

< Step to some neighbor that improves the objective function >

Else

If π(CurrentVertex) $\in F$

Then Output(π(CurrentVertex))

Else

Output("LP is infeasible")

EndIf

Exit

EndIf

EndRepeat

Figure 5: The arrangement method.

Given a current vertex $\mathbf{y}$ of $\mathcal{A}_\mathcal{H}$, the arrangement method considers all the neighbors of $\mathbf{y}$ in the graph of $\mathcal{A}_\mathcal{H}$. Each such neighbor lies on a common line with $\mathbf{y}$ defined by the intersection of $d-1$ hyperplanes of $\mathcal{H}$ that contain $\mathbf{y}$. There are d such lines, and on each there are one or two neighbors. Overall the neighbors of $\mathbf{y}$ can be found in time $O(nd)$. The method then determines the coordinates in $\mathbb{R}^{d+1}$ of the vertices of $\mathcal{AP}$ that correspond to $\mathbf{y}$ and its neighbors. This relies on the following lemma:

Lemma 3.1. *Given a vertex $\mathbf{y}$ of $\mathcal{A}_\mathcal{H}$, the vertex $\chi(\mathbf{y})$ of $\mathcal{AP}$ can be found in time $O(nd)$.*

Proof. We rely on the terminology and results of Section 2.2. Let $\mathbf{x} := \chi(\mathbf{y})$ and $\lambda := \mathrm{str}(\mathbf{y})$. λ can be easily computed in time $O(nd)$. We also know that for some $t > 0$,

$$\mathbf{x} = \chi((y_1, y_2, \ldots, y_d)) = (ty_1, ty_2, \ldots, ty_d, t).$$

The point $\mathbf{w} := \sum_{i=1}^{n} \lambda_i \mathbf{a}'_i$ satisfies $\mathrm{str}(\mathbf{w}) = \lambda$ and thus lies on the facet f of $\mathcal{Z}$ that is polar to $\mathbf{x}$. Therefore $\mathbf{wx} = 1$. The coordinates of $\mathbf{w}$ can be computed in time $O(nd)$. We have

$$\mathbf{w}(ty_1, ty_2, \ldots, ty_d, t) = t\mathbf{w}(y_1, y_2, \ldots, y_d, 1) = 1,$$

an equation from which t can be computed in time $O(d)$. □

The locations in $\mathbb{R}^{d+1}$ of the current vertex and its $O(d)$ neighbors on $\mathcal{AP}$ can thus be computed in time $O(nd^2)$. The arrangement method compares the objective function values of these vertices and determines which neighbors improve the value compared to the current vertex, using the linear objective function $\mathbf{v}$ from Theorem 2.5. Given the identities of improving neighbors, the method chooses one of them, and proceeds to the next iteration with the corresponding vertex of $\mathcal{A}_\mathcal{H}$. Any polynomial-time policy that relies on the metric and combinatorial properties of the vertices can be used to guide the choice, and any of the known pivot rules for the simplex method is a valid candidate. For consistency, we continue to refer to the policy used by the arrangement method as a pivot rule.

If no neighboring vertex improves the objective function on $\mathcal{AP}$ we use Theorem 2.5 to conclude that if $\pi(\mathbf{y}) \notin F$ then the LP is infeasible, otherwise we have found a feasible vertex $\pi(\mathbf{y})$. Pseudo-code for the arrangement method is given in Figure 5.

4 Graphs of Arrangements

4.1 Preliminaries

In this section we consider an arrangement $\mathcal{A}_\mathcal{H}$ of a set $\mathcal{H}$ of n hyperplanes in $\mathbb{R}^d$. The vertices and edges of $\mathcal{A}_\mathcal{H}$ naturally define an undirected graph $\mathcal{G}(\mathcal{A}_\mathcal{H})$. This is precisely the graph that the arrangement method takes a walk on. It is therefore important to understand its properties. Some basic ones are easy to discern: Every vertex $\mathbf{x}$ of $\mathcal{G}(\mathcal{A}_\mathcal{H})$ lies on d lines, each of which is an intersection of $d-1$ hyperplanes of $\mathcal{H}$. On each such line, $\mathbf{x}$ is adjacent to two edges of $\mathcal{A}_\mathcal{H}$. If $n > d$, at least d of these edges are finite and therefore correspond to edges of $\mathcal{G}(\mathcal{A}_\mathcal{H})$. The degree of $\mathbf{x}$ in $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is therefore between d and $2d$. The number of vertices of $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is $\binom{n}{d}$ and the number of edges is $(n-d)\binom{n}{d-1}$.

In what follows we derive strong bounds on properties of $\mathcal{G}(\mathcal{A}_\mathcal{H})$ that are relevant to the design and analysis of walks on this graph. We show that it has a small polynomial diameter and thus for any pair of vertices there exists a walk from one to the other with a polynomial number of steps. (An analogous statement is not known to be true for graphs of polytopes.) We also show that $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is an expander, in the sense of having inverse polynomial conductance, implying that a random walk on $\mathcal{G}(\mathcal{A}_\mathcal{H})$ mixes in polynomial time. We provide an example that demonstrates that polytope graphs can have inverse exponential conductance.

Finally, we analyze the connectivity of graphs of arrangements. A straightforward such analysis for $\mathcal{G}(\mathcal{A}_\mathcal{H})$ can only hope to show that the graph is at most d-connected. Indeed, since there are "boundary" vertices of $\mathcal{A}_\mathcal{H}$ that only have degree d, deleting the neighbors of such a vertex disconnects $\mathcal{G}(\mathcal{A}_\mathcal{H})$. However,

these "boundary effects" give a somewhat distorted picture of the connectedness in the "interior" of the arrangement graph. To avoid this distortion, we analyze the connectivity of the graph $\mathcal{G}(\mathcal{A}_\mathcal{C})$ of a spherical arrangement $\mathcal{A}_\mathcal{C}$ of a set $\mathcal{C}$ of n great $(d-1)$-spheres in $\mathbb{S}^d$. $\mathcal{G}(\mathcal{A}_\mathcal{C})$ is defined on the vertices and edges of $\mathcal{A}_\mathcal{C}$ as in the case of hyperplanes. By Corollary 2.2, $\mathcal{G}(\mathcal{A}_\mathcal{C})$ is isomorphic to the graph of the arrangement polytope $\mathcal{AP}$. We show that $\mathcal{G}(\mathcal{A}_\mathcal{C})$ is $2d$-connected; this result is tight. For comparison, Balinski's theorem [4] states that the graph of a polytope in $\mathbb{R}^d$ is only d-connected.

We note some basic facts without proof, keeping in mind the general position assumption from Section 2.1: A set $\Gamma \subset \mathcal{H}$ (resp., $\Gamma \subset \mathcal{C}$) of cardinality d intersects in a single point in $\mathbb{R}^d$ (resp., a pair of antipodal points in $\mathbb{S}^d$). If $\Gamma \subset \mathcal{H}$ (resp., $\Gamma \subset \mathcal{C}$) consists of $k < d$ hyperplanes (resp., spheres), the intersection $\sigma := \bigcap_{\gamma\in\Gamma} \gamma$ is a $(d-k)$-flat in $\mathbb{R}^d$ (resp., a great $(d-k)$-sphere in $\mathbb{S}^d$). The arrangement on σ of the intersections of $\mathcal{H}\backslash\Gamma$ (resp., $\mathcal{C}\backslash\Gamma$) with σ is called the *restriction* of $\mathcal{A}_\mathcal{H}$ (resp., of $\mathcal{A}_\mathcal{C}$) on σ and is isomorphic to a hyperplane arrangement of $n-k$ $(d-k-1)$-hyperplanes in $\mathbb{R}^{d-k}$ (resp., a spherical arrangement of $n-k$ great $(d-k-1)$-spheres in $\mathbb{S}^{d-k}$).

4.2 Diameter

Theorem 4.1. *The diameter of $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is at most $d(n-2d+1)$.*

Proof. Given two vertices u, v of $\mathcal{G}(\mathcal{A}_\mathcal{H})$, we use induction on d to show that v can be reached from u in at most $d(n-2d+1)$ steps. When $d=1$, $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is a path on n vertices. The distance from u to any other vertex is at most $n-1$, which is precisely the stated bound for $d=1$.

For an arbitrary $d \geq 2$, if u and v lie in a common hyperplane H_0 of $\mathcal{H}$ the bound follows immediately by induction—just consider the restriction of $\mathcal{A}_\mathcal{H}$ on H_0. Otherwise we show that u can reach a vertex u' that lies in a common hyperplane $H_0 \in \mathcal{H}$ with v in $n-2d+1$ steps, such that there is some hyperplane $H_1 \in \mathcal{H}$ that contains u but not u'. The induction hypothesis can then be used to bound the distance between u' and v in the graph of the restriction of $\mathcal{A}_{\mathcal{H}\backslash\{H_1\}}$ on H_0. Since this restriction is isomorphic to an arrangement of $n-2$ hyperplanes in $\mathbb{R}^{d-1}$, the bound is $(d-1)(n-2d+1)$; adding this to the $(n-2d+1)$ steps it takes to reach u' from u gives the result.

To reach u' from u, consider an arbitrary $(d-1)$-tuple of hyperplanes of $\mathcal{H}$ incident to u. They intersect in a line γ that intersects each of the other $n-d+1$ hyperplanes of $\mathcal{H}$ once. The graph of the restriction of $\mathcal{A}_\mathcal{H}$ on γ is a path on $n-d+1$ vertices. Since v is incident to d hyperplanes of $\mathcal{H}$ that are not incident to u, d of the vertices on γ lie on a common hyperplane of $\mathcal{H}$ with v. At least one of these d vertices lies within distance at most $(n-2d+1)$ from u. □

Similar arguments can be used to derive the following theorem.

Theorem 4.2. *The diameter of $\mathcal{G}(\mathcal{A}_\mathcal{C})$ is at most $(d-1)\left(\left\lceil\frac{n}{2}\right\rceil - d\right) + (n-2d+2)$.*

4.3 Expansion

In this section we show that the arrangement graph $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is an expander, in the sense that it has inverse polynomial conductance. Recall that the conductance of a graph $G(V,E)$ is a measure of the connectivity of $G(V,E)$ and is defined to be the minimum over all cuts $(S,\bar{S})$, $S, \bar{S} \subseteq V$, of the conductance of the cut $(S,\bar{S})$. The conductance of the cut $(S,\bar{S})$ is in turn defined as

$$\frac{|E_{S,\bar{S}}|}{\min\{|E_S|, |E_{\bar{S}}|\}}.$$

Here $E_{S,\bar{S}}$ is the set of edges with one endpoint in S and the other in $\bar{S}$, and E_S (resp., $E_{\bar{S}}$) is the set of edges with at least one endpoint in S (resp., in $\bar{S}$).

Theorem 4.3. *The conductance of $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is*

$$\Omega\left(\frac{n-d}{n^3 \log n}\right).$$

When $n > (1+\delta)d$ for any constant $\delta > 0$, this bound is $\Omega(1/n^2 \log n)$.

Proof. We first define a related graph, the hyperplane interchange graph, $\mathcal{G}_d^n$, whose conductance is easier to bound. The vertex set of $\mathcal{G}_d^n$ is the same as that of the arrangement graph, and its edges form a superset of those of the arrangement graph. Each vertex of $\mathcal{G}_d^n$ (and of $\mathcal{G}(\mathcal{A}_\mathcal{H})$) is the intersection of exactly d hyperplanes. Two vertices are adjacent in $\mathcal{G}_d^n$ iff they share $d-1$ hyperplanes, i.e., if it is possible to get from one to the other by a single swap of hyperplanes. Note that the structure of $\mathcal{G}_d^n$ is independent of the collection of hyperplanes $\mathcal{A}_\mathcal{H}$ and this graph can be defined in purely combinatorial terms: The vertices of $\mathcal{G}_d^n$ correspond to all strings from $\{0,1\}^n$ that have Hamming weight d, and the edges of $\mathcal{G}_d^n$ correspond to pairs of vertices that have Hamming distance 2.

We prove that the conductance of $\mathcal{G}_d^n$ is $\Omega(1/n \log n)$. To this end we show that a random walk on $\mathcal{G}_d^n$ mixes in $O(n \log n)$ steps, and invoke the well known relationship between conductance and mixing time [21]. The $O(n \log n)$ bound on mixing time follows from a coupling argument that we now sketch. The random walk we consider stays at the current vertex with probability $1/2$, and with probability $1/2$ moves to an adjacent vertex in $\mathcal{G}_d^n$ in each step. To perform the coupling argument, consider two copies of the random walk on $\mathcal{G}_d^n$, one starting from a fixed vertex and the other from the stationary distribution. The coupling is carried out by specifying a correlated dynamics for the two Markov chains, such that watching only a single chain yields the original random walk on $\mathcal{G}_d^n$. To do so, let us match up the coordinates of the current vertices in the two random walks v and w as follows: coordinates with 1's are matched to 1's and 0's to 0's. If $v_i = w_i = 1$ or if $v_i = w_i = 0$ then coordinate i is mapped to itself. We now complete the bijection arbitrarily, subject only to the condition that the bijection maps 1's to 1's and 0's to 0's. Now the two chains are coupled in the natural way: both chains move or stay at the current vertex. For a move they select a corresponding pair of coordinates to interchange. We omit the proof that the expected time for the two chains to couple is $O(n \log n)$. By the coupling bound (see for example [27]) it thus follows that the mixing time is also $O(n \log n)$.

Returning to the arrangement graph $\mathcal{G}(\mathcal{A}_\mathcal{H})$ whose conductance we are actually interested in, let us show that the conductance of any cut of $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is $\Omega((n-d)/n^3 \log n)$. Let $(S, \bar{S})$ be a cut in $\mathcal{G}(\mathcal{A}_\mathcal{H})$. First consider the same cut in $\mathcal{G}_d^n$. Since the conductance of $\mathcal{G}_d^n$ is $\Omega(1/n \log n)$, it follows that the number of edges of $\mathcal{G}_d^n$ crossing the cut is $\Omega(|E_S|/n \log n) = \Omega(d(n-d)|S|/n \log n)$. (The last equality stems from the degree of every vertex in $\mathcal{G}_d^n$ being $d(n-d)$.) The crucial observation is that for every edge that crosses the cut $(S, \bar{S})$ in $\mathcal{G}(\mathcal{A}_\mathcal{H})$, there are at most $O(n^2)$ edges that cross the cut in $\mathcal{G}_d^n$. It follows that the number of edges crossing the same cut in $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is $\Omega(d(n-d)|S|/n^3 \log n)$. But in the graph $\mathcal{G}(\mathcal{A}_\mathcal{H})$, $|E_S| = \Theta(d|S|)$. Therefore the number of edges crossing the cut in $\mathcal{G}(\mathcal{A}_\mathcal{H})$ is $\Omega((n-d)|E_S|/n^3 \log n)$, thus proving the desired bound on the conductance of $\mathcal{G}(\mathcal{A}_\mathcal{H})$. □

In contrast to arrangement graphs, graphs of polytopes can have inverse exponential conductance, implying that a random walk needs exponential time to mix on graphs of certain polytopes. Here is the construction.

Observation 4.4. *There exists a polytope P with $n+d$ facets in $\mathbb{R}^d$, such that the conductance of the graph of P is $O(1/n^{\lfloor d/2 \rfloor})$.*

Proof. We give only a brief sketch here. P consists of two "caps" (upper and lower) connected by a vertical "torso". The n hyperplanes that define the upper cap are such that their lower envelope (with respect to the x_d-axis) has $\Omega(n^{\lfloor d/2 \rfloor})$ vertices; this can be accomplished using duals of cyclic polytopes. The lower cap is defined symmetrically. The torso is a product of the $(d-1)$-dimensional simplex with the x_d-axis. When the hyperplanes that define the caps and torso are appropriately translated, each cap has $\Omega(n^{\lfloor d/2 \rfloor})$ vertices and the caps are connected only by the d edges of the torso. The bound on the conductance follows. □

4.4 Connectivity

Theorem 4.5. *$\mathcal{G}(\mathcal{A}_\mathcal{C})$ is $2d$-connected. This bound is the best possible.*

Proof. We use induction on d to show that even after the removal of some set Λ of at most $2d-1$ vertices of $\mathcal{G}(\mathcal{A}_\mathcal{C})$ and their incident edges, any two surviving vertices u, v can be connected by a path through the surviving edges.

When $d = 1$, $\mathcal{G}(\mathcal{A}_\mathcal{C})$ is a cycle and remains connected after the removal of any one vertex. For $d \geq 2$, assume first that $n \geq d+2$. The vertex u lies on d great circles, each defined by some $d-1$ spheres of $\mathcal{C}$ incident to u. Only two vertices, u and its antipodal u^*, lie on more than one of these circles. Since at most $2d-1$ vertices were removed overall, and the vertex sets of these $d-1$ cycles are disjoint except at u and u^*, at least one of the cycles, γ, has had at most one vertex other than u and u^* removed. (Otherwise at least $2d$ vertices would have been removed overall.) Since by assumption u has not been removed, in the worst case two vertices have been removed from γ: u^* and some other vertex z.

The graph of the restriction of $\mathcal{A}_\mathcal{C}$ on γ any of these circles is a cycle on $n-d+1$ pairs of antipodal vertices, including u and u^*. u and u^* are connected by two paths in this cycle, which we call the left path and the right path, see Figure 6. Assume without loss of generality that z lies on the right path. Thus the left path is intact after the removal of z and u^* (except its last point u^*). Any sphere of $\mathcal{C}$ other than the d spheres that contain u intersects γ at two antipodal points, one of which lies on the left path. This implies that u can reach some vertex on any sphere in $\mathcal{C}$. Moreover, the same analysis applies to v, hence v can also reach some vertex on any sphere in $\mathcal{C}$.

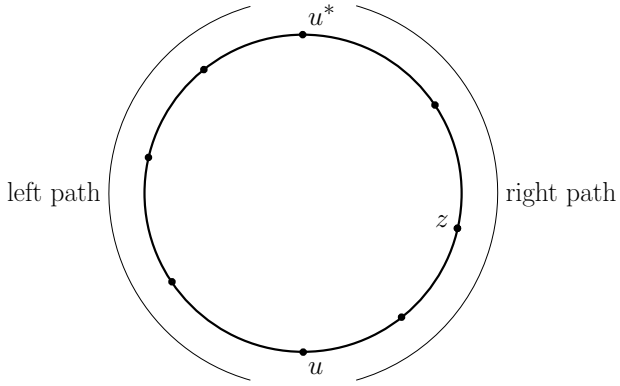


Figure 6: Left and right paths on γ.

We now show that if $n \geq d+2$ then at least one sphere S in $\mathcal{C}$ has had at most $2d-3$ vertices removed from its relative interior. The result then follows since u (resp., v) can reach some vertex $u' \in S$ (resp., $v' \in S$), and the induction hypothesis states that u' and v' are connected within the restriction of $\mathcal{A}_\mathcal{C}$ on S. To see that some sphere S must have had at most $2d-3$ vertices removed assume the contrary, i.e., that at least $2d-2$ vertices were removed from every one of the at least $d+2$ spheres of $\mathcal{S}$. Since every removed vertex lies on d spheres, the number of removed vertices is at least $(2d-2)(d+2)/d$, which is at least $2d$ for $d \geq 2$, a contradiction.

It remains to treat $n \leq d+1$. For $n \leq d-1$ the graph is empty. For $n = d$ it consists of two antipodal vertices connected by d edges; the removal of either or both of them leaves the graph connected, albeit possibly empty. For $n = d+1$ the only possible graph $\mathcal{G}(\mathcal{A}_\mathcal{C})$ is illustrated in Figure 7. It consists of $d+1$ pairs of antipodal vertices and every vertex u is a neighbor of all other vertices except its antipodal u^*. Thus if u and v are not antipodal and are not among the at most $2d-1$ removed vertices, they are connected by an edge. If u and v are antipodal, they have $2d$ common neighbors, at least one of which was not removed.

Finally, the bound is tight since removing the $2d$ neighbors of any vertex in $\mathcal{G}(\mathcal{A}_\mathcal{C})$ disconnects the graph. □

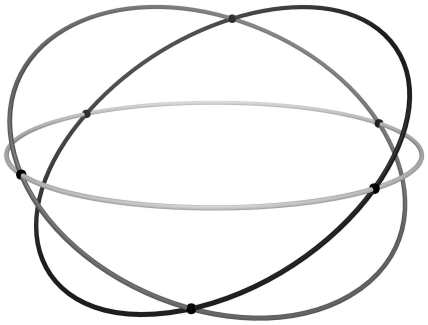


Figure 7: The graph $\mathcal{G}(\mathcal{A}_{\mathcal{C}})$ when $n = d + 1$ and $d = 2$.

5 Open problems

1. **Further analysis.** Show that some pivot rule guides the arrangement method along a path of polynomial length to F, starting from a randomly chosen vertex. Alternatively, can we prove lower bounds for the performance of the method with particular pivot rules when the starting vertex is chosen at random?
2. **Handling optimization.** Extend the arrangement method to naturally handle the optimization version of linear programming. One possibility is to find a direction that is optimized on $\mathcal{AP}$ by a vertex that corresponds to the optimum of the LP (if the LP is feasible). It is clear that such a direction exists, the issue is to compute it in strongly polynomial time. It is possible to reduce this computation to the linear programming feasibility problem. Thus if a strongly polynomial LP feasibility algorithm is developed, it can be used to bootstrap the arrangement method to handle the optimization problem. It would be interesting to find a more direct approach.
3. **Improving the diameter and conductance bounds.** It would be interesting to improve the bounds in Theorems 4.1 and 4.3 or show non-trivial lower bounds. One intriguing conjecture is the following: *The diameter of $\mathcal{G}(\mathcal{A}_{\mathcal{H}})$ is at most $n - d$.*
4. **Other questions concerning arrangement graphs.** Properties of arrangement graphs are a rich source of fascinating questions. For example, the chromatic number of $\mathcal{G}(\mathcal{A}_{\mathcal{H}})$ is at most $d+1$, which is tight. (Very easy, see [14] for the planar case.) Is it true that the chromatic number of $\mathcal{G}(\mathcal{A}_{\mathcal{C}})$ is at most $d + 1$ as well? This is open and interesting even for the case $d = 2$.

Acknowledgements

I would like to thank Umesh Vazirani for his contributions to this work and for many inspiring discussions. I would also like to acknowledge Günter Ziegler for his support and enlightening discussions, as well as Ilan Adler, Gil Kalai, Dick Karp, Nimrod Megiddo, Christos Papadimitriou, and Günter Rote for offering their insights and support at various points throughout this project.

References

[1] I. Adler and N. Megiddo. A simplex algorithm whose average number of steps is bounded between two quadratic functions of the smaller dimension. *Journal of the ACM*, 32:871–895, 1985.

[2] P. K. Agarwal and M. Sharir. Arrangements and their applications. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 49–119. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[3] N. Amenta and G. M. Ziegler. Deformed products and maximal shadows of polytopes. In B. Chazelle, J. E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, pages 57–90. Contemporary Mathematics 223, AMS, Providence, 1999.
[4] M. L. Balinski. On the graph structure of convex polyhedra in n-space. *Pacific Journal of Mathematics*, 11:431–434, 1961.
[5] K. H. Borgwardt. *The Simplex Method. A Probabilistic Analysis*, volume 1 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin Heidelberg, 1987.
[6] J. Bose, H. Everett, and S. Wismath. Properties of arrangement graphs. *International Journal of Computational Geometry and Applications*, 13:447–462, 2003.
[7] K. S. Brown and P. Diaconis. Random walks and hyperplane arrangements. *The Annals of Probability*, 26:1813–1854, 1998.
[8] K. L. Clarkson. Las Vegas algorithms for linear and integer programming. *Journal of the ACM*, 42:488–499, 1995.
[9] E. Cohen and N. Megiddo. Improved algorithms for linear inequalities with two variables per inequality. *SIAM Journal on Computing*, 23:1313–1347, 1994.
[10] G. B. Dantzig. Application of the simplex method to a transportation problem. In T. Koopmans, editor, *Activity Analysis of Production and Allocation*, pages 359–373. Wiley, NY, 1951.
[11] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, Princeton, NJ, 1963.
[12] J. Dunagan and S. Vempala. A simple polynomial-time rescaling algorithm for solving linear programs. In *Proc. 36th ACM Symposium on Theory of Computing*, pages 315–320, 2004.
[13] M. E. Dyer and A. M. Frieze. A randomized algorithm for fixed-dimension linear programming. *Mathematical Programming*, 44(2):203–212, 1989.
[14] S. Felsner, F. Hurtado, M. Noy, and I. Streinu. Hamiltonicity and colorings of arrangement graphs. In *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, pages 155–164, 2000.
[15] K. Fukuda and B. Kaluzny. The criss-cross method can take $\Omega(n^d)$ pivots. In *Proc. 20th ACM Symposium on Computational Geometry*, pages 401–408, 2004.
[16] B. Gärtner, M. Henk, and G. M. Ziegler. Randomized simplex algorithms on Klee-Minty cubes. *Combinatorica*, 18:349–372, 1998.
[17] B. Gärtner and I. Schurr. Linear programming and unique sink orientations. In *Proc. 17th ACM-SIAM Symposium on Discrete Algorithms*, 2006.
[18] B. Gärtner and E. Welzl. Linear programming – randomization and abstract frameworks. In *Proc. 13th Symposium on Theoretical Aspects of Computer Science*, pages 669–687, 1993.
[19] B. Grünbaum. *Arrangements and Spreads*. Regional Conf. Ser. Math. AMS, Providence, RI, 1972.
[20] M. Haimovich. The simplex algorithm is very good!: On the expected number of pivot steps and related properties of random linear programs. Technical report, Columbia University, New York, 1983.
[21] M. Jerrum and A. Sinclair. Approximate counting, uniform generation and rapidly mixing markov chains. *Information and Computation*, 82:93–133, 1989.
[22] V. Kaibel, R. Mechtel, M. Sharir, and G. M. Ziegler. The simplex algorithm in dimension three. *SIAM Journal on Computing*, 34:475–497, 2005.
[23] G. Kalai. A subexponential randomized simplex algorithm. In *Proc. 24th ACM Symposium on Theory of Computing*, pages 475–482, 1992.
[24] G. Kalai and D. J. Kleitman. A quasi-polynomial bound for the diameter of graphs of polyhedra. *Bulletins of the AMS*, 26:315–316, 1992.
[25] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.
[26] L. Khachyan. A polynomial algorithm in linear programming. In *Dokl. Akad. Nauk USSR 244*, pages 1093–1096. Wiley, 1979. English translation in: Soviet Math. Dokl. 20 (1979), 191-194.
[27] L. Lovasz and P. Winkler. Mixing times. In *Microsurveys in Discrete Probability*, volume 44 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 85–134. 1988.
[28] J. Matoušek. Lower bound for a subexponential optimization algorithm. *Random Structures & Algorithms*, 5(4):591–607, 1994.
[29] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16:498–516, 1996.
[30] J. Matoušek and T. Szabó. RANDOM EDGE can be exponential on abstract cubes. In *Proc. 45nd IEEE Symposium on Foundations of Computer Science*, pages 92–100, 2004.
[31] N. Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the ACM*, 31:114–127, 1984.
[32] N. Megiddo. A note on degeneracy in linear programming. *Mathematical Programming*, 35:365–367, 1986.
[33] M. L. Minsky and S. A. Papert. *Perceptrons: An introduction to computational geometry*. MIT Press, 1969.
[34] I. Schurr and T. Szabó. Finding the sink takes some time. *Discrete & Computational Geometry*, 31:627–642, 2004.

[35] R. Seidel. Small-dimensional linear programming and convex hulls made easy. *Discrete & Computational Geometry*, 6:423–434, 1991.
[36] S. Smale. On the average number of steps in the simplex method of linear programming. *Mathematical Programming*, 27:241–262, 1983.
[37] S. Smale. Mathematical problems for the next century. *Mathematical Intelligencer*, 20:7–15, 1998.
[38] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51:385–463, 2004.
[39] T. Szabó and E. Welzl. Unique sink orientations of cubes. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science*, pages 547–555, 2001.
[40] E. Tardos. A strongly polynomial algorithm to solve combinatorial linear programs. *Oper. Res.*, 34:250–256, 1986.
[41] T. Terlaky. A convergent crisscross method. *Math. Oper. und Stat. ser. Optimization*, 16:683–690, 1985.
[42] S. Vavasis and Y. Ye. A primal-dual interior-point method whose running time depends only on the constraint matrix. *Mathematical Programming*, 74:79–120, 1996.
[43] T. Zaslavsky. *Facing up to Arrangements: Face-Count Formulas for Partitions of Space by Hyperplanes*, volume 154 of *Memoirs of the AMS*. AMS, Providence, RI, 1975.
[44] G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1995. Revised edition 1998.
[45] S. Zionts. The crisscross method for solving linear programming problems. *Management Science*, 15:426–445, 1969.