AN OPTIMAL RANDOMIZED ALGORITHM FOR d-VARIATE ZONOID DEPTH

Pat Morin
School of Computer Science
Carleton University
morin@scs.carleton.ca

Abstract. A randomized linear expected-time algorithm for computing the zonoid depth (Dyckerhoff *et al* 1996, Mosler 2002) of a point with respect to a fixed dimensional point set is presented.

1 Introduction

Let S be a set of n points in $\mathbb{R}^d$. For a real number $k \geq 1$, the *k-zonoid* of S is defined as

$$Z_k(S) = \left\{ \sum_{p \in S} \lambda_p p : 0 \leq \lambda_p \leq 1/k \text{ for all } p \in S \text{ and } \textstyle\sum_{p \in S} \lambda_p = 1 \right\}$$

[7, 15]. Notice that, for $k = 1$ the 1-zonoid of S is the convex hull of S, i.e., $Z_1(S) = \text{conv}(S)$. As k increases, $Z_k(S)$ becomes smaller and smaller until the limiting case $k = n$, for which $Z_n(S)$ consists of a single point, the mean of S. The *zonoid depth* of a point $p \in \text{conv}(S)$ with respect to S is defined as

$$Z(p, S) = \sup\{k : p \in Z_k(S)\} \ ,$$

and is a real number in the interval $[1, n]$.

Dyckerhoff *et al* [7] give an algorithm to compute $Z(p, S)$ by solving a linear program in the variables $\{\lambda_p : p \in S\}$. To obtain an efficient algorithm they make use of the fact that most of the constraints on the λ's are independent of S. The worst-case running time of their algorithm is unclear.

Bern and Eppstein [1] study zonoids (also called *reduced convex hulls*) in the context of support vector machines used in machine learning. Among other things they solve a more general problem than that of zonoid depth: Given two sets S_1 and S_2 in $\mathbb{R}^d$, compute the minimum value k such that $Z_k(S_1) \cap Z_k(S_2)$ is non-empty. Their algorithm has a running time of $O(n(Ld \log n)^{O(1)})$, where L is the number of bits used to describe the points in S_1 and S_2. Their algorithm uses Kachiyan's ellipsoid method for linear programming [11] to exploit the fact that, for a given direction v, it is easy (see Section 3) to test if there is a hyperplane orthogonal to v that separates $Z_k(S_1)$ and $Z_k(S_2)$.

Gopala and Morin [10] consider algorithms for bivariate ($d = 2$) zonoid depth and give a randomized $O(n)$ expected time algorithm for computing $Z(p, S)$ when p and S are in $\mathbb{R}^2$. Their algorithm is a combination of two techniques, namely a prune-and-search algorithm due to Lo *et al* [13] for searching the k-level of a line arrangement and an optimization method due to Chan [3] for efficiently converting

decision algorithms into optimization algorithms. While the latter technique extends efficiently into arbitrary (constant) dimensions [4] the former technique, unfortunately, does not. Note that this algorithm is combinatorial in that it makes all decisions based on the results of evaluating a few different kinds of predicates. Other than evaluating these predicates, the running time does not depend on the input precision.

The current paper extends and bridges the above results by giving an $O(n)$ time algorithm to compute $Z(p, S)$ when p and S are in $\mathbb{R}^d$ for any constant dimension d. The algorithm uses a recent method, due to Chan [4], of solving linear programs with many constraints that are defined implicitly by a small number of objects. Again, the algorithm is combinatorial, and the input precision affects only some low-level predicates.

Besides being the first linear-time algorithm for solving the zonoid depth problem in constant dimensions, the current results are interesting for two other reasons:

1. Zonoid depth is one of many definitions of depth proposed in the robust statistics literature [12]. Perhaps the gold standard in this regard is *Tukey (halfspace) depth* [17]:

   $$T(p, S) = \min\{|h \cap S| : h \text{ is a closed halfspace containing } p\} \ .$$

   Tukey depth and zonoid depth have an interesting feature in common; under duality, the combinatorial structure of the depth k contour is determined by the k-level and the $(n - k + 1)$-level of a set of hyperplanes. The structure of k-levels has been extensively studied by combinatorial geometers [14, Chapter 11] although our understanding of their complexity is still not complete, even in 2 dimensions.

   The current result shows a divergence in the computational complexity of Tukey and zonoid depth. In constant dimensions $d \geq 3$ the fastest algorithms for computing the Tukey depth of a point have running times of $\Omega(n^{d-1})$ [5], whereas the current result shows that zonoid depth can be computed in $O(n)$ time in any constant dimension d. When the dimension grows arbitarily large the situation is even worse. Computing $T(p, S)$ is NP-hard in general [2], while the result of Bern and Eppstein [1] yields a polynomial time algorithm for computing $Z(p, S)$ in any dimension. Thus, together these results show that zonoid depth is computationally more tractable than Tukey depth in both large and small dimensions.

2. Our algorithm makes use of Chan's recent technique for solving implicit linear programs in small dimensions [4]. Interestingly, this technique was introduced in order to solve a problem related to Tukey depth, namely the problem of finding a point p that maximizes $T(p, S)$. Unfortunately, the resulting algorithm runs in $O(n \log n + n^{d-1})$ time, limiting its usefulness for dimensions $d \geq 3$.[1] Indeed, although Chan's technique itself does not asymptotically increase the running time as the dimension d increases, it seems that most applications of the technique either break down or have quickly increasing running times as d increases.[2] The current result is therefore an atypical example that illustrates the full utility of this extremely powerful technique.

In the following, all points, vectors, and hyperplanes are assumed to live in $\mathbb{R}^d$ and $\mathbb{H}^d$ denotes the set of all hyperplanes in $\mathbb{R}^d$. The notation x_i denotes the ith coordinate of the point x. We use the $\cdot$ notation to denote the inner-product of two points/vectors, i.e., $x \cdot y = \sum_{i=1}^{d} x_i y_i$. For a set S of n points and a non-zero vector r, $S_1^r, \ldots, S_n^r$ is the sequence of elements of S ordered by decreasing projections onto r, i.e., $S_i^r \cdot r \geq S_{i+1}^r \cdot r$ for all $1 \leq r \leq n - 1$.

[1] In fact, this running time is probably optimal. See Chan [4, Section 1.4] for details.

[2] One notable exception is parametric minimum spanning trees [9].

For a point x and a hyperplane h, we denote by $x{\downarrow}h$ the dth coordinate of the vertical projection of x onto h (the height of x when dropped onto h). For a set H of n hyperplanes, let H_i^x be the ith hyperplane in H encountered by a downward vertical ray originating at $(x_1, \ldots, x_{d-1}, \infty)$. For ease of notation we use the shorthand $H_{-i}^x = H_{|H|-i+1}^x$. For $i > |H|$ we use the convention that H_i^x (respectively H_{-i}^x) is the "horizontal hyperplane at infinity" $\{x : x_d = -\infty\}$ (respectively, $\{x : x_d = +\infty\}$).

The remainder of this paper is organized as follows: Section 2 reviews Chan's generalized optimization technique. Section 3 discusses properties of zonoids in primal and dual space. Section 4 presents an algorithm to answer to the decision problem, given p, S, and k, is $p \in Z_k(S)$? Finally, Section 5 describes our algorithm for, given p and S, computing $Z(p, S)$.

2 Chan's Generalized Optimization Technique

Chan [4] used the following theorem to provide an $O(n \log n)$ time algorithm for maximum Tukey depth.[3] In the following, and throughout the remainder of the paper, we use the shorthand $\cap S$ to denote $\bigcap_{s \in S} s$.

Theorem 1 (Chan 2004)**.** *Let $\mathcal{H}$ denote the set of all halfspaces in $\mathbb{R}^d$, let $\mathcal{P}$ denote the set of all possible inputs to some problem, let $f : \mathcal{P} \mapsto 2^{\mathcal{H}}$ be any function mapping problem inputs to sets of halfspaces, let $g : \mathbb{R}^d \mapsto \mathbb{R}$ be any linear objective function, and let $D(n) = \Omega(n^\epsilon)$, for some $\epsilon > 0$, be a non-decreasing function of n. Suppose that f and g satisfy:*

0. *Given inputs $P_1, \ldots, P_d \in \mathcal{P}$ each of constant size, a point $p \in \cap(f(P_1) \cup \cdots \cup f(P_d))$ maximizing $g(p)$ can be found in constant time.*
1. *Given a point $p \in \mathbb{R}^d$ and an input $P \in \mathcal{P}$ of size n, there exists a $D(n)$ time algorithm to determine whether $p \in \cap f(P)$.*
2. *There exists constants $\alpha < 1$ and r such that, for any input $P \in \mathcal{P}$ of size n, it is possible to compute, in $D(n)$ time, inputs $P_1, \ldots, P_r$, each of size at most $\lceil \alpha n \rceil$, and such that $\cap f(P) = \cap(f(P_1) \cup \cdots \cup f(P_r))$.*

Then there exists a randomized $O(D(n))$ expected time algorithm to compute, for any input $P \in \mathcal{P}$ of size n a point $p \in \cap f(P)$ that maximizes $g(p)$.

It is worth noting that the codomain of the function f may contain infinite sets. That is, it is acceptable (and common) to have inputs $P \in \mathcal{P}$ that generate an infinite number of constraints, i.e., $|f(P)| = \infty$.

[3] Actually, Theorem 1 applies to LP-type problems [16]. Here we only state it's specialization to linear programming problems.

3 Properties of Primal and Dual Zonoids

The k-zonoid $Z_k(S)$ is a convex polytope. The extreme-most vertex of $Z_k(S)$ in direction x can be obtained as a convex combination of the $\lceil k \rceil$ extreme-most points of S in direction x. More precisely,

$$\operatorname*{argmax}_p \{p \cdot x : p \in Z_k(S)\} = \left(\sum_{i=1}^{\lfloor k \rfloor} \frac{1}{k} S_i^x \right) + (1 - \lfloor k \rfloor / k) S_{\lceil k \rceil}^x \tag{1}$$

[1, 10]. Intuitively, we assign the maximum allowable coefficient $(1/k)$ to each of the $\lfloor k \rfloor$ extreme-most vertices and the "leftover" $(1 - \lfloor k \rfloor / k)$ is assigned to the next vertex.

We wish to arrive at a situation in which we can apply Theorem 1 and this is best done by working in the dual. Consider the point-hyperplane duality function φ given by

$$\varphi(x) = \{y \in \mathbb{R}^d : y_d = x_1 y_1 + \cdots + x_{d-1} y_{d-1} - x_d\}$$

when x is a point in $\mathbb{R}^d$ and

$$\varphi(X) = \{\varphi(x) : x \in X\}$$

when S is a subset of $\mathbb{R}^d$. See Edelsbrunner's book [8] for properties of this duality.

Let $H = \varphi(S)$. Then, under this duality, the *dual k-zonoid* $\varphi(Z_k(S))$ is the set of all hyperplanes in $\mathbb{R}^d$ that do not intersect either of two convex sets $A_k(S)$ and $B_k(S)$. That is,

$$\varphi(Z_k(S)) = \{h \in \mathbb{H}^d : h \cap (A_k(S) \cup B_k(S)) = \emptyset\} \ ,$$

where

$$A_k(S) = \left\{ x \in \mathbb{R}^d : x_d \geq \left(\sum_{i=1}^{\lfloor k \rfloor} \frac{1}{k} (x \downarrow H_i^x) \right) + (1 - \lfloor k \rfloor / k)(x \downarrow H_{\lceil k \rceil}^x) \right\} \tag{2}$$

and

$$B_k(S) = \left\{ x \in \mathbb{R}^d : x_d \leq \left(\sum_{i=1}^{\lfloor k \rfloor} \frac{1}{k} (x \downarrow H_{-i}^x) \right) + (1 - \lfloor k \rfloor / k)(x \downarrow H_{-\lceil k \rceil}^x) \right\} \ . \tag{3}$$

The definitions of $A_k(S)$ and $B_k(S)$ follow from (1) and the duality φ. The two sets $A_k(S)$ and $B_k(S)$ are convex, unbounded from above, respectively, below, and piecewise linear. Indeed, the linear pieces of $A_k(S)$ (respectively $B_k(S)$) are in one to one correspondence with the linear pieces of the $\lceil k \rceil$-level (respectively the $(n - \lfloor k \rfloor + 1)$-level) of the hyperplanes in H. Thus, $A_k(S)$ and $B_k(s)$ are convex polytopes that are implicitly defined by the hyperplanes in H and it is these implicit "linear programs" that will ultimately allow us to apply Theorem 1.

4 The Decision Algorithm

Next we consider the following decision problem: Given a point set S and an integer k, is the origin contained in $Z_k(S)$? By translation, a solution to this problem allows us to test if an arbitrary point $p \in \mathbb{R}^d$ is contained in $Z_k(S)$. One approach to solving this problem is to compute the intersection of $Z_k(S)$ with the vertical line $\{x \in \mathbb{R}^d : x_0 = x_1 = \cdots = x_{d-1} = 0\}$ through the origin and then check if this intersection contains the origin.

Under the duality φ, the above strategy is equivalent to finding the lowest point on $A_k(S)$ and the highest point on $B_k(S)$ and checking that each of these points is above, respectively, below, the hyperplane $\{x \in \mathbb{R}^d : x_d = 0\}$. In the remainder, we focus on determining the lowest point in $A_k(S)$. Finding the highest point in $B_k(S)$ is done in a symmetric manner. However, before we can proceed, we need to define a slightly more general problem involving weights.

Let S be a set of n points in $\mathbb{R}^d$ and let $w : S \mapsto \mathbb{N}$ be a function assigning positive integer weights to the elements of S. We denote by S^w the multiset in which each element $p \in S$ occurs $w(p)$ times. The *w-weighted zonoid* $Z_k(S, w)$ is simply the k-zonoid of the multiset S^w, i.e., $Z_k(S, w) = Z_k(S^w)$. As with standard zonoids, the weighted zonoid $Z_k(S, w)$ dualizes to the set of all lines that do not intersect either of two convex regions $A_k(S, w)$ and $B_k(S, w)$, where $A_k(S, w) = A_k(S^w)$ and $B_k(S, w) = B_k(S^w)$.

This definition of weighted zonoids allows us to naturally define subproblems. For a subset $C \subseteq S$, define the *total weight*

$$w(C) = \sum_{p \in C} w(p)$$

and the *weighted mean*

$$\mu(C) = \frac{1}{w(C)} \sum_{p \in C} p \times w(p) \ .$$

The *contraction* of (S, w) by C is obtained by replacing the points of C by their weighted average, $\mu(C)$. More precisely, the contraction of (S, w) by C is the pair (R, v) where

$$R = (S \setminus C) \cup \{\mu(C)\}$$

and

$$v(p) = \begin{cases} w(p) & \text{if } p \in S \setminus C \\ w(C) & \text{if } p = \mu(C) \end{cases}$$

The following lemma shows that contraction results in strictly smaller zonoids:

Lemma 1. *If (R, v) is a contraction of (S, w) by C then $Z_k(R, v) \subseteq Z_k(S, w)$.*

Proof. Let x be any point in $Z_k(R, v)$. Then, by the definition of zonoids:

$$\begin{aligned} x &= \sum_{p \in R^v} \lambda_p p \\ &= \left(\sum_{p \in (R \setminus \{\mu(C)\})^v} \lambda_p p \right) + \left(\sum_{p \in \{\mu(C)\}^v} \lambda_{\mu(C)} p \right) \\ &= \left(\sum_{p \in (S \setminus C)^w} \lambda_p p \right) + \left(\sum_{p \in C^w} \lambda_{\mu(C)} p \right) \\ &\in Z_k(S, w) \end{aligned}$$

as required. □

We now have all the tools required to apply Theorem 1 to solve our decision problem.

Theorem 2. *Given a set S of n points in $\mathbb{R}^d$ and a function $w : S \mapsto \mathbb{N}$ that is computable in constant time, the point $x \in A_k(S, w)$ such that x_d is minimum can be found in $O(n)$ expected time.*

Proof. Let f be the function that maps the pair (S, w) onto a set of halfspaces whose intersection is $A_k(S, w)$ and let the objective function $g(p) = p_d$. We need to show that the function f and g satisfy Conditions 0–2 of Theorem 1.

To satisfy Condition 0 of Theorem 1 we can enumerate all the linear constraints generated by each of the d subproblems and use any linear programming algorithm to find a point x that satisfies all constraints and such that x_d is minimum. There are only d subproblems, each of constant size, so this step takes constant time, as required.

To satisfy Condition 1 of Theorem 1 we observe that testing if $x \in A_k(S, w)$ simply involves checking if x satisfies (2). Let $H = \varphi(S)$. This check can be accomplished by using a $D(n) = O(n)$ time weighted selection algorithm [6, Exercise 9-2] to compute the smallest index t and the hyperplanes $H_1^x, \ldots, H_t^x$ such that $\sum_{i=1}^{t} w(\varphi(H_i^x)) \geq k$. Once this is done we need only check (2) which, in the weighted setting, becomes

$$x \geq \left(\sum_{i=1}^{t-1} \frac{1}{k} (x{\downarrow}H_i^x) \times w(\varphi(H_i^x)) \right) + \frac{1}{k} (x{\downarrow}H_t^x) \times \left(k - \sum_{i=1}^{t-1} v(\varphi(H_i^x)) \right) \quad .$$

To satisfy Condition 2 of Theorem 1 we make use of *cuttings* [14, Section 4.7]. In particular, we use the fact that, in $O(n)$ time, it is possible to partition $\mathbb{R}^d$ into $r = O(1)$ simplices $\Delta_1, \ldots, \Delta_r$ such that the interior of each simplex is intersected by at most $n/2$ of the hyperplanes in $\varphi(S)$. For each simplex Δ_i we create a subproblem (S_i, w_i) as follows: Let $C_i \subseteq S$ contain every point $p \in S$ such that $\varphi(p)$ is above the interior of Δ_i. We first construct the pair (T_i, w_i) by contracting (S, w) by C_i. Next, we obtain S_i by removing from T_i every point p such that $\varphi(p)$ is below the interior of Δ_i. The subproblems (S_i, v_i) for $1 \leq i \leq r$ that we obtain in this manner are each of size at most $n/2 + 2$.

It follows from Lemma 1 (the contraction step) and the definition of $Z_k(S, w)$ (the deletion step) that $Z_k(S_i, w_i) \subseteq Z_k(S, w)$. In the dual, this means that $A_k(S_i, w_i) \supseteq A_k(S, w)$. To satisfy Condition 2 of Theorem 1 we must show that $\bigcap_{i=1}^{r} A_k(S_i, w_i) = A_k(S, w)$. To do this, consider any point x on the boundary of $A_k(S, w)$. It is sufficient to show that x is also on the boundary of at least one region $A_k(S_i, w_i)$ for $1 \leq i \leq r$. The point x is defined by $\lceil k \rceil$ hyperplanes $h_1, \ldots, h_{\lceil k \rceil} \in \varphi(S^w)$ in the sense that

$$x_d = \left(\sum_{i=1}^{\lfloor k \rfloor} \frac{1}{k} (x{\downarrow}h_i) \right) + (1 - \lfloor k \rfloor / k)(x{\downarrow}h_{\lceil k \rceil}) \quad .$$

Let $q = x{\downarrow}h_{\lceil k \rceil}$. There is some simplex Δ_i that contains q. Observe that each of $h_1, \ldots, h_{\lceil k \rceil - 1}$ is either completely above the interior of Δ_i or intersects Δ_i. Furthermore, any hyperplane in $\varphi(S)$ that is completely above Δ_i is one of $h_1, \ldots, h_{\lceil k \rceil - 1}$. Therefore, the subproblem (S_i, w_i) is obtained from (S, w) by contracting $C_i \subseteq \{\varphi(h_1), \ldots, \varphi(h_{\lceil k \rceil - 1})\}$ and then deleting some subset of $S \setminus \{\varphi(h_1), \ldots, \varphi(h_{\lceil k \rceil - 1})\}$. Let $I = \varphi(S_i^{w_i})$. Then, every point x in $A_k(S_i, w_i)$ must satisfy

$$x_d \quad \geq \quad \left(\sum_{i=1}^{\lfloor k \rfloor} \frac{1}{k} (x{\downarrow}I_i^x) \right) + (1 - \lfloor k \rfloor / k)(x{\downarrow}I_{\lceil k \rceil}^x)$$

$$= \left(\sum_{i=1}^{\lfloor k \rfloor} \frac{1}{k} (x \downarrow h_i) \right) + (1 - \lfloor k \rfloor / k)(x \downarrow h_{\lceil k \rceil}) \ .$$

Thus x is on the boundary of $A_k(S_i, w_i)$, as required. We have now satisfied all three conditions necessary to apply Theorem 1, completing the proof. □

5 The Optimization Algorithm

In the previous section we showed, given p, S and k, how to answer the question: Is $p \in Z_k(S)$? In this section we consider the optimization problem, given p and S: What is the maximum value of k such that $p \in Z_k(S)$? For this problem, we can apply Theorem 1 again, this time on a problem in $\mathbb{R}^{d+1}$.

Consider the point set

$$Z(S) = \{p \in \mathbb{R}^{d+1} : (p_1, \ldots, p_d) \in Z_{p_{d+1}}(S)\} \ .$$

The set $Z(S)$ is a convex polytope in $\mathbb{R}^{d+1}$. Dualizing $Z(S)$ as before gives two regions $A(S)$ and $B(S)$. We are interested in the hyperplane $h_0 = \{x \in \mathbb{R}^{d+1} : x_d = 0\}$. In particular, the value k we are searching for is the minimum of k_A and k_B where

$$k_A = \min\{p_{d+1} : p \in h_0 \cap A(S)\}$$

and

$$k_B = \min\{p_{d+1} : p \in h_0 \cap B(S)\}$$

In words, we want the minimum value of k such that $A_k(S)$ (respectively, $B_k(S)$) intersects the hyperplane $h_0 = \{x \in \mathbb{R}^d : x_d = 0\}$.

Theorem 3. *Given a set S of n points in $\mathbb{R}^d$ and a point $p \in \mathbb{R}^d$, the maximum value k such that $p \in Z_k(S)$ can be found in $O(n)$ expected time.*

Proof Sketch. The proof is another application of Theorem 1 to find the values k_A and k_B described above. We focus only on finding k_A, as finding k_B is a symmetric problem. The details are much the same as in Theorem 2 so we only sketch them. As before we generalize $A(S)$ and $B(S)$ to the weighted setting using multisets and let $f(S, w)$ be the function that maps (S, w) on to the set of linear constraints that define $h \cap A(S^w)$.

As before, S satisfies Condition 0 of Theorem 1 since, for constant size subproblems we can explictly generate the polytopes $Z(S_1, w_1), \ldots, Z(S_{d+1}, w_{d+1})$, compute their common intersection with h_0 and find a point in the intersection maximizing the objective function $g(p) = p_{d+1}$.

The decision problem we must solve to satisfy Condition 1 of Theorem 1 is the problem of testing whether a point $p \in \mathbb{R}^{d+1}$ is contained in $h_0 \cap A(S)$. But this is simply a matter of checking that $p \in h_0$ and that $(p_1, \ldots, p_d)$ is in $A_{p_{d+1}}(S)$, a problem for which we described an $O(n)$ time algorithm in the proof of Theorem 2.

The partitioning into subproblems required to satisfy Condition 2 of Theorem 1 can be done in exactly the same manner as described in the proof of Theorem 2. To see that this partitioning works in the current case we need only observe that the paritioning makes no use of the value k and the argument used to show its correctness holds for all values of k. This completes the proof sketch. □

References

[1] M. W. Bern and D. Eppstein. Optimization over zonotopes and training support vector machines. In *Workshop on Algorithms and Data Structures*, pages 111–121, 2001.

[2] N. Chakravarti. Some results concerning post-infeasibility analysis. *European Journal of Operations Research*, 73:139–143, 1994.

[3] T. M. Chan. Geometric applications of a randomized optimization technique. *Discrete & Computational Geometry*, 22(4):547–567, December 1999.

[4] T. M. Chan. An optimal randomized algorithm for maximum Tukey depth. In *Proc. 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 423–429, 2004.

[5] T. M. Chan. Low-dimensional linear programming with violations. *SIAM Journal on Computing*, 34:879–893, 2005.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. McGraw-Hill, second edition, 2001.

[7] R. Dyckerhoff, G. Koshevoy, and K. Mosler. Zonoid data depth: Theory and computation. In A. Prat, editor, *COMPSTAT 1996 - Proceedings in Computational Statistics*, pages 235–240. Physica-Verlag, Heidelberg, August 1996.

[8] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, Heidelberg, Germany, 1997.

[9] D. Eppstein. Setting parameters by example. *SIAM Journal on Computing*, 82:638–653, 2003.

[10] H. Gopala and P. Morin. Algorithms for bivariate zonoid depth. *Computational Geometry: Theory and Applications*, 2006. Special issue of selected papers from the *16th Canadian Conference on Computational Geometry (CCCG 2004)*.

[11] L. G. Khachiyan. A polynomial time algorithm for linear programming. *Soviet Mathematics Doklady*, 20:1092–1096, 1979.

[12] R. Liu, J. M. Parelius, and K. Singh. Multivariate analysis by data depth: Descriptive statistics, graphics and inference. *The Annals of Statistics*, 27(3):783–858, June 1999.

[13] C.-Y. Lo, J. Matousek, and W. Steiger. Algorithms for ham-sandwich cuts. *Discrete & Computational Geometry*, 11:433–452, 1994.

[14] J. Matoušek. *Lectures on Discrete Geometry*. Springer-Verlag, New York, 2002.

[15] K. Mosler. *Multivariate Dispersion, Central Regions and Depth. The Lift Zonoid Approach*, volume 165 of *Lecture Notes in Statistics*. Springer-Verlag New York, Inc, 2002.

[16] M. Sharir and E. Welzl. A combinatorial bound for linear programming and related problems. In *Proceedings of Symposion on Theoretical Aspects of Computer Science (STACS '92)*, pages 569–588, 1992.

[17] J. W. Tukey. Mathematics and the picturing of data. In Ralph D. James, editor, *Proceedings of the International Congress of Mathematicians*, volume 2, pages 523–531, Vancouver Canada, August 1974.