Approximating Majority Depth

Dan Chen* Pat Morin†

Abstract

We consider the problem of approximating the majority depth (Liu and Singh, 1993) of a point q with respect to an n-point set, S, by random sampling. At the heart of this problem is a data structures question: How can we preprocess a set of n lines so that we can quickly test whether a randomly selected vertex in the arrangement of these lines is above or below the median level. We describe a Monte-Carlo data structure for this problem that can be constructed in $O(n \log n)$ time, can answer queries $O((\log n)^{4/3})$ expected time, and answers correctly with high probability.

1 Introduction

A *data depth measure* quantifies the centrality of an individual (a point) with respect to a population (a point set). Depth measures are an important part of multivariate statistics and many have been defined, include Tukey depth [17], Oja depth [13], simplicial depth [10], majority depth [11], and zonoid depth [8]. For an overview of data depth from a statistician's point of view, refer to the survey by Small [15]. For a computational geometer's point of view refer to Aloupis' survey [1].

In this paper, we focus on the bivariate majority depth measure. Let S be a set of n points in $\mathbb{R}^2$. For a pair $x, y \in S$, the *major side* of x, y is the union of the (at most 2) closed halfplanes having both x and y on their boundary that contain at least $n/2$ points of S. The *majority depth* [11, 14] of a point, q, with respect to S, is defined as the number of pairs $x, y \in S$ that have q in their major side.

Under the usual projective duality [9], the set S becomes a set, S^*, of lines; pairs of points in S become vertices in the arrangement, $A(S^*)$, of S^*; and q becomes a line, q^*. The *median-level* of $A(S^*)$ is the closure of the set of points on lines in S that have exactly $\lfloor n/2 \rfloor$ lines of S above them. Then the majority depth of q with respect to S is equal to the number of vertices, x, in $A(S^*)$ such that

1. x is above q^* and x is above the median level; or
2. x is below q^* and x is below the median level.

Chen and Morin [5] present an algorithm for computing majority depth that works in the dual. Their algorithm works by computing the median level, computing the intersections of q^* with the median level, and using fast inversion counting to determine the number, t, of vertices of the arrangement sandwiched between q^* and the median level. The majority depth of q is then equal to $\binom{n}{2} - t$. The running time of this algorithm is within a logarithmic factor of m, the complexity of the median level.

The maximum complexity of the median level of n lines has been the subject of intense study since it was first posed. The current best upper bound is $O(n^{4/3})$, due to Dey [7] and the current best lower bound is $2^{\Omega(\sqrt{\log n})}$, due to Tóth [16]. The median level can be computed in time $O(\min\{m \log n, n^{4/3}\})$ [2, 3]. Thus, the worst-case running time of Chen and Morin's majority depth algorithm is $\omega(n(\log n)^c)$ for any constant c, but no worse than $O(n^{4/3} \log n)$.

It seems difficult for any algorithm that computes the exact majority depth of a point to avoid (at least implicitly) computing the median level of $A(S^*)$. This motivates approximation by random sampling. In particular, one can use the simple technique of sampling vertices of $A(S^*)$ and checking whether

1. each sample lies above or below q^*; and
2. each sample lies above or below the median level of S^*.

In the primal, this is equivalent to taking random pairs of points in S and checking, for each such pair, (x, y), if, (1) the closed upper halfplane, h_{xy}, with x and y on its boundary, contains q and (2) if h_{xy} contains $n/2$ or more points of S.

The former test takes constant time but the latter test leads to a data structuring problem: Preprocess the set S^* so that one can quickly test, for any query point, x, whether x is above or below the median level of $A(S^*)$. (Equivalently, does a query halfplane, h, contain $n/2$ or more points of S.) We know of two immediate solutions to this problem. The first solution is to compute the median level explicitly, in $O(\min\{m \log n, n^{4/3}\})$ time, after which any query can be answered in $O(\log n)$ time by binary search on the x-coordinate of x. The second solution is to construct a half-space range counting

*School of Computer Science, Carleton University, dchen4@connect.carleton.ca

†School of Computer Science, Carleton University, morin@scs.carleton.ca

structure—a partition tree—in $O(n \log n)$ time that can count the number of points of S in h_{xy} in $O(n^{1/2})$ time [4].

The first solution is not terribly good, since Chen and Morin's algorithm shows that computing the *exact* majority depth of q can be done in time that is within a logarithmic factor of m, the complexity of the median level. (Though if the goal is to preprocess in order to approximate the majority depth for many different points, then this method may be the right choice.)

In this paper, we show that the second solution can be improved considerably, at least when the application is approximating majority depth. In particular, we show that when the query point is a randomly chosen vertex of the arrangement $A(S^*)$, a careful combination of partition trees [4] and ε-approximations [12] can be used to answer queries in $O((\log n)^{4/3})$ expected time. This faster query time means that we can use more random samples which leads to a more accurate approximation.

The remainder of this paper is organized as follows. In Section 2 we review results on range counting and ε-approximations and show how they can be used for approximate range counting. In Section 3 we show how these approximate range counting results can be used to quickly answer queries about whether a random vertex of S^* is above or below the median level of S^*. In Section 4 we briefly mention how all of this applies to the problem of approximating majority depth. Finally, Section 5 concludes with an open problem.

2 Approximate Range Counting

In this section, we consider the problem of approximate range counting. That is, we study algorithms to preprocess S so that, given a closed halfplane h and an integer $i \geq 0$, we can quickly return an integer $r_i(h, S)$ such that

$$||h \cap S| - r_i(h, S)| \leq i \ .$$

This data structure is such that queries are faster when the allowable error, i, is larger.

There are no new results in this section. Rather it is a review of two relevant results on range searching and ϵ-approximations that are closely related, but separated by nearly 20 years. The reason we do this is that, without a guide, it can take some effort to gather and assemble the pieces; some of the proofs are existential, some are stated in terms of discrepancy theory, and some are stated in terms of VC-dimension. The reader who already knows all this, or is uninterested in learning it, should skip directly to Lemma 2.

The first tools we need come from a recent result of Chan on optimal partition trees and their application to exact halfspace range counting [4, Theorems 3.2 and 5.3, with help from Theorem 5.2]:

Theorem 1. *Let S be a set of n points in $\mathbb{R}^2$ and let $N \geq n$ be an integer. There exists a data structure that can preprocess S in $O(n \log N)$ expected time so that, with probability at least $1-1/N$, for any query halfplane, h, the data structure can return $|h \cap S|$ in $O(n^{1/2})$ time.*

We say that a halfplane, h, *crosses* a set, X, of points if h neither contains X nor is disjoint from X. The partition tree of Theorem 1 is actually a *binary space partition tree*. Each internal node, u, is a subset of $\mathbb{R}^2$ and the two children of a node are the subsets of u obtained by cutting u with a line. Each leaf, w, in this tree has $|w \cap S| \leq 1$. The $O(n^{1/2})$ query time is obtained by designing this tree so that, with probability at least $1 - 1/N$, there are only $O(n^{1/2})$ nodes crossed by any halfplane.

For a geometric graph $G = (S, E)$, the *crossing number* of G is the maximum, over all halfplanes, h, of the number of edges $uw \in E$ such that h crosses $\{u, w\}$. From Theorem 1 it is easy to derive a spanning tree of S with crossing number $O(n^{1/2})$ using a bottom-up algorithm: Perform a post-order traversal of the partition tree. When processing a node u with children v and w, add an edge to the tree that joins an arbitrary point in $v \cap S$ to an arbitrary point in $w \cap S$. Since a halfplane cannot cross any edge unless it also crosses the node at which the edge was created, this yields the following result [4, Corollary 7.1]:

Theorem 2. *For any n point set, S, and any $N \geq n$, it is possible to compute, in $O(n \log N)$ expected time, a spanning tree, T, of S that, with probability at least $1 - 1/N$, has crossing number $O(n^{1/2})$.*

A spanning tree is not quite what is needed for what follows. Rather, we require a matching of size $\lfloor n/2 \rfloor$. To obtain this, we first convert the tree, T, from Theorem 2 into a path by creating a path, P, that contains the vertices of T in the order they are discovered by a depth-first traversal. It is easy to verify that the crossing number of P is at most twice the crossing number of T. Next, we take every second edge of P to obtain a matching:

Corollary 1. *For any n point set, S, and any $N \geq n$, it is possible to compute, in $O(n \log N)$ expected time, a matching, M, of S of size $\lfloor n/2 \rfloor$ that, with probability at least $1 - 1/N$ has crossing number $O(n^{1/2})$.*

The following argument is due to Matoušek, Welzl and Wernsich [12, Lemma 2.5]. Assume, for simplicity, that n is even and let $S' \subset S$ be obtained by taking exactly one endpoint from each edge in the matching M obtained by Corollary 1. Consider some halfplane h, and let M_h^I be the subset of the edges of M contained in h and let M_h^C be the subset of edges crossed by h. Then

$$|h \cap S| = 2|M_h^I| + |M_h^C| \ .$$

In particular,

$$|h \cap S| - |M_h^C| \leq 2|h \cap S'| \leq |h \cap S| + |M_h^C|$$

Since $|M_h^C| \in O(n^{1/2})$, this is good news in terms of approximate range counting; the set S' is half the size of S, but $2|h \cap S'|$ gives estimate of $|h \cap S|$ that is off by only $O(n^{1/2})$. Next we show that this can be improved considerably with almost no effort.

Rather than choose an arbitrary endpoint of each edge in M to take part in S', we choose each one of the two endpoints at random (and independently of the other $n/2 - 1$ choices). Then, each edge in M_h^C has probability $1/2$ of contributing a point to $h \cap S'$ and each edge in M_h^I contributes exactly one point to $h \cap S'$. Therefore,

$$\mathrm{E}[2|h \cap S'|] = 2\left(1|M_h^I| + \frac{1}{2}|M_h^C|\right) = |h \cap S| \ .$$

That is, $2|h \cap S'|$ is an unbiased estimator of $|h \cap S|$. Even better: the error of this estimator is (2 times) a binomial$(|M_h^C|, 1/2)$ random variable, with $|M_h^C| \in O(n^{1/2})$. Using standard results on the concentration of binomial random variables (i.e., Chernoff Bounds [6]), we immediately obtain:

$$\Pr\{|2|h \cap S'| - |h \cap S|| \geq cn^{1/4}(\log N)^{1/2}\} \leq 1/N \ ,$$

for some constant $c > 0$. That is, with probability $1 - 1/N$, $2|h \cap S'|$ estimates $|h \cap S|$ to within an error of $O(n^{1/4}(\log N)^{1/2})$. Putting everything together, we obtain:

Lemma 1. *For any n point set, S, and any $N \geq n$, it is possible to compute, in $O(n \log N)$ expected time, a subset S' of S of size $\lceil n/2 \rceil$ such that, with probability at least $1 - 1/N$, for every halfplane h,*

$$|2|h \cap S'| - |h \cap S|| \in O(n^{1/4}(\log N)^{1/2}) \ .$$

What follows is another argument by Matoušek, Welzl and Wernisch [12, Lemma 2.2]. By repeatedly applying Lemma 1, we obtain a sequence of $O(\log n)$ sets $S_0 \supset S_1 \cdots \supset S_r$, $S_0 = S$ and $|S_j| = \lceil n/2^j \rceil$. For $j \geq 1$, the set S_j can be computed from S_{j-1} in $O(2^{-j} n \log N)$ time and has the property that, with probability at least $1 - 1/N$, for every halfplane h,

$$\left|2^j |h \cap S_j| - |h \cap S|\right| \in O(2^{3j/4} n^{1/4} (\log N)^{1/2}) \ . \quad (1)$$

At this point, we have come full circle. We store each of the sets $S_0, \ldots, S_r$ in an optimal partition tree (Theorem 1) so that we can do range counting queries on each set S_i in $O(|S_i|^{1/2})$ time. This (finally) gives the result we need on approximate range counting:

Lemma 2. *Given any set S of n points in $\mathbb{R}^2$ and any $N \geq n$, there exists a data structure that can be constructed in $O(n \log N)$ expected time and, with probability at least $1 - 1/N$, can, for any halfspace h and any integer $i \in \{0, \ldots, n\}$, return a number $r_i(h, S)$ such that*

$$||h \cap S| - r_i(h, S)| \leq i \ .$$

Such a query takes $O(\min\{n^{1/2}, (n/i)^{2/3}(\log N)^{1/3}\})$ expected time.

Proof. The data structure is a sequence of optimal partition trees on the sets $S_0, \ldots, S_r$. All of these structures can be computed in $O(n \log N)$ time, since $|S_0| = n$ and the size of each subsequent set decreases by a factor of 2.

To answer a query, (h, i), we proceed as follows: If $i \leq n^{1/4}$, then we perform exact range counting on the set $S_0 = S$ in $O(n^{1/2})$ time to return the value $|h \cap S|$. Otherwise, we perform range counting on the set S_j where j is the largest value that satisfies

$$C 2^{3j/4} n^{1/4} (\log N)^{1/2} \leq i \ ,$$

where the constant C depends on the constant in the big-Oh notation in (1). This means $|S_j| = O((n/i)^{4/3}(\log N)^{2/3}))$ and the query takes expected time

$$O(|S_j|^{1/2}) = O((n/i)^{2/3}(\log N)^{1/3}) \ ,$$

as required. □

Our main application of Lemma 2 is a test that checks whether a halfspace, h, contains $n/2$ or more points of S.

Lemma 3. *Given any set S of n points in $\mathbb{R}^2$ and any $N \geq n$, there exists a data structure that can be constructed in $O(n \log N)$ expected time and, with probability at least $1 - 1/N$, can, for any halfspace h determine if $|h \cap S| \geq n/2$. Such a query takes expected time*

$$Q(i) = \begin{cases} O(n^{1/2}) & \text{for } 0 \leq i \leq n^{1/4} \\ O((n/i)^{2/3}(\log N)^{1/3}) & \text{otherwise,} \end{cases}$$

where $i = ||h \cap S| - n/2|$.

Proof. As preprocessing, we construct the data structure of Lemma 2. To perform a query, we perform a sequence of queries (h, i_j), for $j = 0, 1, 2, \ldots$, where $i_j = n/2^j$. The jth such query takes $O(2^{2j/3}(\log N)^{1/3})$ time and the question, "is $|h \cap S| \geq n/2$?" is resolved once $n/2^j < i/2$. Since the cost of successive queries is exponentially increasing, this final query takes time $O(\min\{n^{1/2}, (n/i)^{2/3}(\log N)^{1/3}\})$ and dominates the total query time. □

3 Side of Median Level Testing

We are now ready to tackle the main problem that comes up in trying to estimate majority depth by random sampling: Given a range pair of points $x, y \in S$, determine if there are more than $n/2$ points in the upper halfspace, h_{xy}, whose boundary is the line through x and y. In this section, though, it will be more natural to work in the dual setting. Here the question becomes: Given a random vertex, x, of $A(S^*)$, determine whether x is above or below the median level of S^*. The data structure in Lemma 3 answers these queries in time $O((n/i)^{2/3}(\log N)^{1/3})$ when the vertex x is on the $n/2-i$ or $n/2+i$ level.

Before proving our main theorem, we recall a result of Dey [7, Theorem 4.2] about the maximum complexity of a sequence of levels.

Lemma 4. *Let L be any set of n lines and let s be the number of vertices of $A(L)$ that are on levels $k, k+1, \ldots, k+j$. Then, $s \in O(nk^{1/3}j^{2/3})$.*

We are interested in the special case of Lemma 4 where $k = n/2 - i$ and $j = 2i$:

Corollary 2. *Let L be any set of n lines. Then, for any $i \in \{1, \ldots, n/2\}$ the maximum total number of vertices of $A(L)$ whose level is in $\{n/2-i, \ldots, n/2+i\}$ is $O(n^{4/3}i^{2/3})$.*

Corollary 2 is useful because it gives bounds on the distribution of the level of a randomly chosen vertex of $A(S^*)$.

Theorem 3. *Given any set, L, of n lines and any $c > 0$, there exists a data structure that can test if a point x is above or below the median level of L. For any constant, c, this structure can be made to have the following properties:*

1. *It can be constructed in $O(n \log n)$ expected time and uses $O(n)$ space;*
2. *with probability at least $1-n^{-c}$, it answers correctly for all possible queries; and*
3. *when given a random vertex of $A(L)$ as a query, the expected query time is $O((\log n)^{4/3})$.*

Proof. The data structure is, of course, the data structure of Lemma 3 with $N = n^c$. Let n_i be the number of vertices of $A(L)$ on levels $n/2-i$ and $n/2+i$. Then the expected query time of this data structure is at most

$$F(n_0, \ldots, n_{n/2}) = \frac{1}{\binom{n}{2}} \sum_{i=0}^{n/2} n_i Q(i) \ , \tag{2}$$

where, for sufficiently large n, $Q(i)$ is upper-bounded by

$$Q(i) \leq \begin{cases} \beta n^{1/2} & \text{if } 0 \leq i \leq n^{1/4} \\ \beta (n/i)^{2/3} (\log N)^{1/3} & \text{otherwise.} \end{cases}$$

for some constant $\beta > 0$. Our goal, therefore, is to upper-bound $F(n_0, \ldots, n_{n/2})$ subject to Dey's constraints (Lemma 4):

$$\sum_{i=0}^{j} n_i \leq \gamma n^{4/3} j^{2/3}$$

for some constant $\gamma > 0$ and all $j \in \{0, \ldots, n/2\}$.

Working in our favour is that $Q(i) \geq Q(i')$ for all $i \leq i'$. This implies that, to obtain an upper bound on $F(n_0, \ldots, n_{n/2})$, we can set

$$\sum_{i=0}^{j} n_i = \begin{cases} \gamma n^{4/3} & \text{if } j = 0 \\ \gamma n^{4/3} j^{2/3} & \text{otherwise} \end{cases} \tag{3}$$

for all $j \in \{0, \ldots, n/2\}$. To see why this is so, suppose we have a sequence $S = n_0, \ldots, n_{n/2}$ that satisfies Dey's constraints but for which $\sum_{i=0}^{j} n_i < \gamma n^{4/3} j^{2/3}$ for some index j. If $j = n/2$ then we can obviously increase the value of n_j, still satisfy Dey's constraints and increase the value of $F(n_0, \ldots, n_j)$. Otherwise ($j \in \{0, \ldots, n/2-1\}$), the sequence

$$S' = n_0, \ldots, n_j + \delta, n_{j+1} - \delta, \ldots, n_{n/2} \ ,$$

where $\Delta = \gamma n^{4/3} j^{2/3} - \sum_{i=0}^{j} n_i$, also satisfies Dey's constraints. Furthermore,

$$F(S') - F(S) = \Delta Q(j) - \Delta Q(j+1) \geq 0 \ ,$$

so $F(S') \geq F(S)$. Repeatedly applying this type of modification (or using induction) shows that the sequence $S = n_0, \ldots, n_{n/2}$ that satisifies (3) is a sequence that maximizes $F(S)$.

Finally, we can bound the sequence that satisfies (3) by differentiating $\gamma n^{4/3} j^{2/3}$ with respect to j. This yields $n_i \in O(n^{4/3}/i^{1/3})$ for all $i \in \{1, \ldots, n/2\}$. Plugging this back into (2) yields

$$\begin{aligned} & F(n_0, \ldots, n_{n/2}) \qquad (4) \\ & \leq \frac{1}{\binom{n}{2}} \left(O(n^{4/3} Q(0)) + \sum_{i=1}^{n/2} O(n^{4/3} Q(i)/i^{1/3}) \right) \\ & \leq o(1) \\ & \quad + \frac{1}{\binom{n}{2}} \sum_{i=1}^{n^{1/4}} O(n^{4/3} n^{1/2}/i^{1/3}) \qquad (5) \\ & \quad + \frac{1}{\binom{n}{2}} \sum_{i=n^{1/4}+1}^{n/2} O(n^{4/3} (n/i)^{2/3} (\log N)^{1/3}/i^{1/3}) \qquad (6) \end{aligned}$$

Recall that $\int_1^n i^{-1/3} \, \mathrm{d}i = \frac{3}{2}(n^{2/3} - 1)$. Using this integral

to bound the sum in (5) allows us to just squeak by:

$$\begin{aligned}(5) &= \frac{1}{\binom{n}{2}} \sum_{i=1}^{n^{1/4}} O(n^{4/3} n^{1/2} / i^{1/3}) \\ &= \frac{1}{\binom{n}{2}} O(n^{4/3} n^{1/2} (n^{1/4})^{2/3}) \quad \text{(bounding by integral)} \\ &= O(1)\end{aligned}$$

We are not so lucky with the sum in (6), which ends up being harmonic:

$$\begin{aligned}(6) &= \frac{1}{\binom{n}{2}} \sum_{i=n^{1/4}+1}^{n/2} O(n^{4/3} (n/i)^{2/3} (\log N)^{1/3} / i^{1/3}) \\ &= \sum_{i=n^{1/4}+1}^{n/2} O((\log N)^{1/3} / i) \\ &= O((\log n)(\log N)^{1/3}) \quad (\text{since } \textstyle\sum_{i=1}^{n} 1/i = O(\log n)) \\ &= O((\log n)^{4/3}) \ ,\end{aligned}$$

since $N = n^c$ and c is constant. To summarize, the expected running time of the query algorithm is at most

$$F(n_0, \ldots, n_{n/2}) \leq o(1) + (5) + (6) = O((\log n)^{4/3}) \ . \ \square$$

4 Estimating Majority Depth

Finally, we return to our application, namely estimating majority depth.

Theorem 4. *Given a set S of n points in $\mathbb{R}^2$ and any constant $c > 0$, there exists a data structure that can preprocess S in $O(n \log n)$ expected time such that, for any point q, the data structure can compute, in $O(r(\log n)^{4/3})$ expected time, a value $d'(q, S)$ such that*

$$\Pr\left\{ \frac{|d'(q,S) - d(q,S)|}{d(q,S)} \geq \varepsilon \right\} \leq \exp\left(-\Omega\left(\varepsilon^2 rp\right)\right) + n^{-c} \ ,$$

where $d(q, S)$ is the majority depth of p with respect to S and $p = d(q, S)/\binom{n}{2}$ is the normalized majority depth of q.

Proof. The data structure is the one described in Theorem 3. Let $p = d(q, S)/\binom{n}{2}$. Select r random vertices of $A(S^*)$ (by taking random pairs of lines in S^*) and, for each sample, test if it contributes to $d(q, S)$. This yields a count $r' \leq r$ where

$$\mathrm{E}[r'] = rp \ .$$

We then return the value $d'(q, S) = (r'/r)\binom{n}{2}$, so that $\mathrm{E}[d'(q, S)] = d(q, S)$, as required.

To prove the error bound, we use the fact that r' is a binomial(p, r) random variable. Applying Chernoff Bounds [6] on r' yields:

$$\Pr\{|r' - rp| \geq \varepsilon rp\} \leq \exp(-\Omega(\varepsilon^2 rp)) \ .$$

Finally, the algorithm may fail not because of badly chosen samples, but rather, because the data structure of Theorem 3 fails. The probability that this happens is at most n^{-c}. Therefore, the overall result follows from the union bound. $\square$

5 Conclusions

Although the estimation of majority depth is the original motivation for studying this problem, the underlying question of the tradeoffs involved in preprocessing for testing whether a point is above or below the median level seems a fundamental question that is still far from answered. In particular, we have no good answer to the following question:

Open Problem 1. What is the fastest linear-space data structure for testing if an arbitrary query point is above or below the median level of a set of n lines?

To the best of our knowledge, the current state of the art is partition trees, which can only answer these queries in $O(n^{1/2})$ time.

References

[1] G. Aloupis. Geometric measures of data depth. In R.Liu, R.Serfling, and D.Souvaine, editors, *Data Depth: Robust Multivariate Analysis, Computational Geometry and Applications*, volume 72 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 147–158. American Mathematical Society, 2006.

[2] G. S. Brodal and R. Jacob. Dynamic planar convex hull. In *FOCS*, pages 617–626. IEEE Computer Society, 2002.

[3] T. M. Chan. Remarks on k-level algorithms in the plane. Manuscript, 1999.

[4] T. M. Chan. Optimal partition trees. *Discrete & Computational Geometry*, 47(4):661–690, 2012.

[5] D. Chen and P. Morin. Algorithms for bivariate majority depth. In *Proceedings of the 23rd Canadian Conference on Computational Geometry (CCCG'11)*, pages 425–430, 2011.

[6] H. Chernoff. A measure of the asymptotic efficient of tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.

[7] T. K. Dey. Improved bounds for planar k-sets and related problems. *Discrete & Computational Geometry*, 19(3):373–382, 1998.

[8] 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.

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

[10] R. Liu. On a notion of data depth based on random simplices. *Annals of Statistics*, 18(1):405–414, 1990.

[11] R. Liu and K. Singh. A quality index based on data depth and multivariate rank tests. *Journal of the American Statistical Association*, 88(421):252–260, 1993.

[12] J. Matoušek, E. Welzl, and L. Wernisch. Discrepancy and approximations for bounded VC-dimension. *Combinatorica*, 13:455–466, 1993.

[13] H. Oja. Descriptive statistics for multivariate distributions. *Statistics and Probability Letters*, 1(6):327–332, 1983.

[14] K. Singh. A notion of majority depth. Technical report, Department of Statistics, Rutgers University, 1991.

[15] C. Small. A survey of multidimensional medians. *International Statistical Review*, 58(3):263–277, 1990.

[16] G. Tóth. Point sets with many k-sets. In *Symposium on Computational Geometry*, pages 37–42, 2000.

[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.