Discrete Comput Geom (2008) 39: 656–677
DOI 10.1007/s00454-007-9046-6

Efficient Algorithms for Maximum Regression Depth

Marc van Kreveld · Joseph S.B. Mitchell · Peter Rousseeuw · Micha Sharir · Jack Snoeyink · Bettina Speckmann

Received: 5 September 2005 / Revised: 3 December 2007 /
Accepted: 3 December 2007 / Published online: 21 December 2007

A preliminary version of this paper appeared in the proceedings of the 15th Annual ACM Symposium on Computational Geometry (1999)

M. van Kreveld partially funded by the Netherlands Organization for Scientific Research (NWO) under FOCUS/BRICKS grant number 642.065.503.
J.S.B. Mitchell's research largely conducted while the author was a Fulbright Research Scholar at Tel Aviv University. The author is partially supported by NSF (CCR-9504192, CCR-9732220), Boeing, Bridgeport Machines, Sandia, Seagull Technology, and Sun Microsystems.
M. Sharir supported by NSF Grants CCR-97-32101 and CCR-94-24398, by grants from the U.S.–Israeli Binational Science Foundation, the G.I.F., the German–Israeli Foundation for Scientific Research and Development, and the ESPRIT IV LTR project No. 21957 (CGAL), and by the Hermann Minkowski—MINERVA Center for Geometry at Tel Aviv University.
J. Snoeyink supported in part by grants from NSERC, the Killam Foundation, and CIES while at the University of British Columbia.

M. van Kreveld (✉)
Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands
e-mail: marc@cs.uu.nl

J.S.B. Mitchell
Department of Applied Mathematics and Statistics, SUNY Stony Brook, Stony Brook, USA
e-mail: jsbm@ams.sunysb.edu

P. Rousseeuw
Department of Mathematics and Computer Science, Universitaire Instelling Antwerpen, Antwerpen, Belgium
e-mail: Peter.Rousseeuw@ua.ac.be

M. Sharir
School of Computer Science, Tel Aviv University, Tel Aviv, Israel
e-mail: michas@tau.ac.il

J. Snoeyink
Department of Computer Science, UNC Chapel Hill, Chapel Hill, USA
e-mail: snoeyink@cs.unc.edu

B. Speckmann
Department of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands
e-mail: speckman@win.tue.nl

Abstract We investigate algorithmic questions that arise in the statistical problem of computing lines or hyperplanes of maximum *regression depth* among a set of n points. We work primarily with a dual representation and find points of maximum *undirected depth* in an arrangement of lines or hyperplanes. An $O(n^d)$ time and $O(n^{d-1})$ space algorithm computes undirected depth of all points in d dimensions. Properties of undirected depth lead to an $O(n\log^2 n)$ time and $O(n)$ space algorithm for computing a point of maximum depth in two dimensions, which has been improved to an $O(n\log n)$ time algorithm by Langerman and Steiger (Discrete Comput. Geom. 30(2):299–309, 2003). Furthermore, we describe the structure of depth in the plane and higher dimensions, leading to various other geometric and algorithmic results.

1 Introduction

The notion of the *depth* of a point with respect to a set of point data is important in statistical analysis. Several proposals for depth have been made which include, for example, Tukey depth, Oja depth, simplicial and convex-layers depth, as well as regression depth. Because of its application to statistics the design of algorithms for the computation of points of maximal or minimal depths (depending on the precise definition) has attracted a great deal of attention in the Computational Geometry community in recent years. See [7, 29] and the references therein for the precise definitions of these depth measures and an overview of the contribution of Computation Geometry to the computation of robust statistics.

Motivated by the study of *robust regression* in statistics [18, 24–28, 30, 32, 33], Peter Rousseeuw posed the question of computing maximum regression depth in his invited talk at the 14th ACM Symposium on Computational Geometry: Given a set P of n points in the plane, the *regression depth* of a line is the minimum number of points that must be removed from P to allow the line to rotate about a pivot point on the line to a vertical position without ever containing a remaining point of P.[7] This definition is given more generally in the next section.

A line (or hyperplane) of maximum depth has statistical properties that are desirable as a robust regression estimator [1, 2]. The experimental investigation of these properties has been hampered by the inefficiency of the straightforward algorithms for computing maximum depth. These required $\Theta(n^3)$ time in the plane [26] and $\Theta(n^{2d-1}\log n)$ time in dimensions $d \geq 3$ [25, 28].

In the next section, we define an equivalent dual problem, computing *undirected depth* in an arrangement of lines or hyperplanes. The properties of undirected depth will lead to an $O(n^d)$ algorithm for computing regression depth for all dimensions. In Sect. 3, we focus on arrangements in the plane and we present an algorithm to compute a cell of maximum depth in $O(n\log^2 n)$ time. Langerman and Steiger subsequently improved this to $O(n\log n)$ time, but as their algorithm is based on our approach and lemmas, we think it is important to present our algorithm nevertheless. In Sect. 4, we continue the analysis of depth with algorithms to determine the

[7]Rousseeuw also posed a combinatorial question, resolved by Amenta et al. [3], who show that for any set of n points in R^d, there exists a hyperplane with regression depth at least $\lceil n/(d+1)\rceil$.

deepest vertex in an arrangement, show a connection with k-sets, and deal with depth in degenerate arrangements. In Sect. 5, we comment on computing depth in higher dimensions, and show how to reduce space requirements from $O(n^d)$ to $O(n^{d-1})$.

2 Duality and Undirected Depth in Arrangements

Although regression depth is defined for a line or hyperplane among n points, it is easier to work with a duality transformation that maps points to hyperplanes and vice versa. We use the duality from Edelsbrunner's book [13]: an inversion about the unit paraboloid $x_d = x_1^2 + x_2^2 + \cdots + x_{d-1}^2$ that maps a point $p = (p_1, p_2, \ldots, p_d)$ to the hyperplane $p^D : x_d = 2p_1x_1 + 2p_2x_2 + \cdots + 2p_{d-1}x_{d-1} - p_d$ and maps a hyperplane $h : x_d = a_1x_1 + a_2x_2 + \cdots + a_{d-1}x_{d-1} + b$ to the point $h^D = (a_1/2, a_2/2, \ldots, a_{d-1}/2, -b)$. This duality preserves point/line incidence and above/below relationships. Note that the duality mapping will neither accept nor produce vertical hyperplanes, which have equations that do not involve the variable x_d.

All rotations of a hyperplane h can be generated as follows. Choose a set Q of d affinely independent points whose affine hull is h. Move one of the points $q_0 \in Q$ by increasing (or decreasing) its last coordinate toward infinity. If the points Q are still taken to span h, then h rotates toward the vertical about the $(d-1)$-flat spanned by points of $Q \setminus \{q_0\}$.

The dual of a rotation is easy to interpret. The points of Q map to hyperplanes through a common point h^D. Hyperplane ${q_0}^D$ moves parallel to itself up (or down) the x_d axis, so the point common to all hyperplanes moves from h^D toward infinity along a ray that is contained in the duals of the stationary points.

Given n primal points P, the number that must be removed to allow a particular rotation are the number that are passed over by the rotation, plus the number that are on the final vertical plane (which our rotation never reaches). This number can be counted in the dual as the number of hyperplanes dual to points in P that are crossed by the ray corresponding to the rotation, plus the number of hyperplanes parallel to the ray. Therefore, for an arrangement of n hyperplanes $\mathcal{A}$, we define the *undirected depth*, or just *depth*, of a point p to be the minimum number of hyperplanes intersected by some ray from p, counting parallel hyperplanes as intersecting at infinity. Hyperplanes containing p are counted for all rays. For the rest of this paper we focus on computing depth of a point in an arrangement of n lines or hyperplanes.

We use the notation depth(p) for the value of undirected depth. Since, for any cell C of an arrangement, all points have the same depth, we can as well use the notation depth(C). (In this paper, unless otherwise stated, we use the word *cell* to refer to a full-dimensional cell in an arrangement.) Figure 1 shows a two-dimensional example with labels for some cells of depth 0, 1, and 2; the maximum depth of 3 occurs at 8 vertices and two edges.

The *directions for a cell C* are the directions of rays that intersect depth(C) lines or hyperplanes of the arrangement. We can call such rays *witnesses* that the cell has a certain depth. We next observe three simple lemmas about depth by translating witness rays in the arrangement of hyperplanes in R^d: (1) depth of lower-dimensional features in the arrangement can be determined from depth of adjacent d-dimensional cells, (2) directions are disjoint for adjacent cells of the same depth, and (3) directions determining depth are inherited from adjacent cells of lower depth.

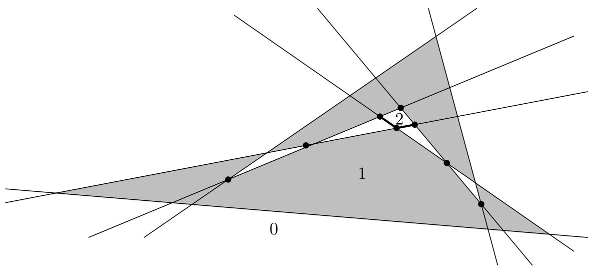


Fig. 1 Arrangement with cells of depth 0 (unbounded), 1 (*shaded*), and 2 (*unshaded*, bounded); maximum depth of 3 occurs at the indicated eight vertices and two edges

Lemma 1 *In an arrangement of hyperplanes, let p be a point on k hyperplanes, and let i be the minimum of the depths of cells whose closure contains p. Then depth$(p) = i + k$.*

Proof First we can observe that depth$(p) \le i + k$: a ray that starts in the cell and crosses i hyperplanes can be translated to start at p at the cost of crossing all hyperplanes through p that it did not cross before. Second, if we take a ray not contained in a hyperplane incident on p that witnesses depth(p) and translate its starting point infinitesimally into the first cell entered by the ray, we can observe that there is an adjacent cell with depth depth$(p) - k$, which is therefore the minimum cell depth i. □

Lemma 2 *In an arrangement of hyperplanes, let h be a hyperplane that separates a cell B of depth i from an adjacent cell A of depth at least i. No witness ray for B crosses h.*

Proof Let ρ be a ray from B that crosses h, and let ρ' be a translation of this ray that begins in A. Translated ray ρ' intersects the same hyperplanes as ρ, except for h. But since ρ' intersects at least i hyperplanes, ρ intersects at least $i + 1$ hyperplanes and is not a witness ray for B. □

Lemma 3 (Inheritance lemma) *The directions for a cell of depth i are the union of the directions for the adjacent cells of depth $i - 1$.*

Proof We prove that the set of directions for a cell A with depth$(A) = i$ contains the union. For any adjacent cell B of depth $i - 1$, let ray ρ be a witness for B. By Lemma 2, translating ρ to start in A adds at most one (and, therefore, exactly one) intersection, and provides a witness that A inherits the direction of ρ.

To prove the other inclusion, take a witness ray ρ' that depth$(A) = i$. We can choose the start point of ρ' so that ρ' does not pass through any vertex of the arrangement. By clipping ρ' to start in an adjacent cell B, we obtain a witness that depth$(B) \le i - 1$. But the depth of B cannot be less than $i - 1$, since depth$(A) = i$ and we already know that A inherits all directions for B with only one more intersection. Thus, the directions for A are contained in the union. □

As a corollary of Lemma 3, the depth of all points with respect to a set of hyperplanes can be computed by constructing the arrangement of hyperplanes [14, 15] and labeling cells in a breadth-first search. The unbounded cells are labeled with their depth zero. Then, for $i = 1, 2, \ldots$, all cells with label $i-1$ cause their adjacent, unlabeled cells to be labeled i. Finally, lower-dimensional cells can be labeled according to Lemma 1.

Corollary 4 *For n hyperplanes in R^d, the depths of all cells, and thus the maximum depth too, can be computed in $O(n^d)$ time by building the arrangement and traversing the graph of adjacent cells.*

3 An Algorithm for Maximum Depth Cells in the Plane

Undirected depth in two dimensions satisfies some additional properties that allow an efficient algorithm to compute a two-dimensional cell of maximum depth. Langerman and Steiger [21] built upon the results presented in this section to give an optimal algorithm that finds a maximum depth cell in $O(n \log n)$ time and linear space.

Suppose that we are given a set L of n lines in the plane. We first consider nondegenerate sets of lines (arrangements of lines) only, that is, no line is vertical, no two lines are parallel, and no three lines pass through a single point. We will relax this assumption in Sect. 4.5. Our goal is to find, among all the points of the plane that do not lie on lines of L, a point p whose depth is maximum. Note that vertices and edges of the arrangement $\mathcal{A}(L)$ may attain greater depth than p—we return to these in Sect. 4.1.

We will use a binary search on x-coordinates of vertices of the arrangement $\mathcal{A}(L)$, with a test for which side of a vertical line contains a maximum depth cell. Section 3.1 establishes properties that allow a sidedness test; Sect. 3.2 describes a tournament data structure needed to implement the sidedness test.

3.1 A Sidedness Test

In the plane, we use two concepts to determine which side of a vertical test line can have cells of maximum depth: a *wedge lemma* and the notion of *top directions*.

Lemma 5 (Wedge lemma) *Let p be a point, possibly on a line $\ell \in L$, let u and v be two rays starting at p, let W be the convex wedge (cone) defined by p, u, and v, and let f be a feature of the arrangement $\mathcal{A}(L)$.*

(i) *If ℓ intersects W, f is a cell, and u and v intersect at most i other lines each, then f has depth at most i.*
(ii) *If ℓ intersects W, f is an edge, and u and v intersect at most i and $i-1$ other lines, then f has depth at most i.*
(iii) *If ℓ intersects W, f is a vertex, and u and v intersect at most $i-1$ other lines each, then f has depth at most i.*
(iv) *If ℓ does not intersect W or p does not lie on a line, f is a cell, and u and v intersect at most $i+1$ and i (other) lines, then f has depth at most i.*
(v) *If ℓ does not intersect W or p does not lie on a line, f is an edge, and u and v intersect at most i (other) lines each, then f has depth at most i.*
(vi) *If ℓ does not intersect W or p does not lie on a line, f is a vertex, and u and v intersect at most i and $i-1$ (other) lines, then f has depth at most i.*

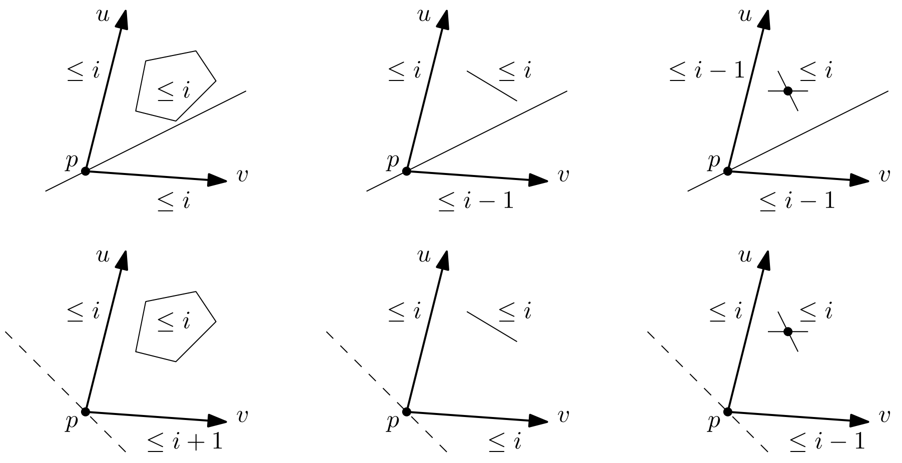


Fig. 2 Six cases of the Wedge lemma

Proof For case (i) (see Fig. 2, top left), consider the lines that intersect the union of rays from p in directions u and v. There are at most $2i + 1$ intersections, if we count the line containing p only once. If we translate this union within the wedge, although we may lose intersections with lines that intersect both rays, we will not gain intersections. Thus, if the apex is inside a cell of the line arrangement, one of the translated rays will witness that the depth is at most i. All other cases follow by a similar counting argument. In case (iv), the possible line through p does not intersect the wedge so u or v can intersect one more line. In the other cases, the line or lines containing f influence the count by one or two as well. □

The Wedge lemma is helpful for identifying maximum depth cells, as in the following corollary.

Corollary 6 *Suppose that a cell C has three directions u, v, and w that span the plane by positive linear combinations and witness the value of depth(C). Then C is a deepest cell.*

Proof Apply the Wedge lemma, case (i), to the three wedges defined by pairs of directions. □

We can order the witness rays for a cell C by increasing slope to the right of C and decreasing slope to the left. We call the two extreme directions for witness rays the *top directions* for the cell C. (In general, the top directions need not be witnesses.) There will be a single *top direction* when one side of the line has no witness rays, or when the ray upward is a witness. Figure 3 illustrates a cell with two top directions.

Suppose that we can determine the top directions for all cells along a vertical line ℓ. Then next lemma shows that we can then determine whether a maximum depth cell occurs to the left or right of ℓ. We give an algorithmic proof, since this becomes part of our procedure for computing maximum depth.

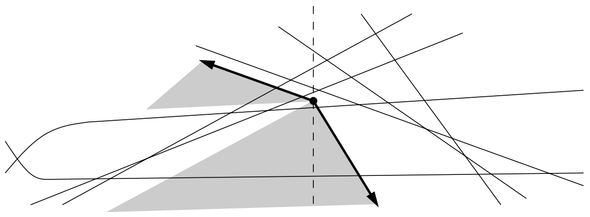


Fig. 3 Directions (*shaded*) and top directions

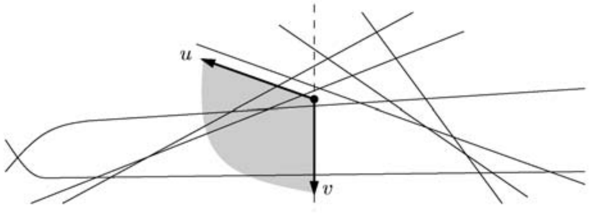


Fig. 4 Region R

Lemma 7 *Given a vertical test line ℓ that does not pass through any vertex in an arrangement of n lines in the plane, and given a top direction for each cell intersected by ℓ, one can determine one side of ℓ that intersects a maximum depth cell.*

Proof Let i denote the maximum depth of the cells intersected by ℓ. We will be able to sweep up the line ℓ and, on one of the sides of ℓ, maintain a region R that does not intersect a cell of depth greater than i. Region R is, in fact, a wedge from the vertical downward direction, v, to a top direction, u, as illustrated in Fig. 4.

Initially, we choose a point $p \in \ell$ in the lowest cell, which will have two top directions, to the right and left of the vertical downward direction v (parallel to the lines with smallest and largest slopes, respectively). We choose a top direction as u and form the wedge R between v and u. Note that R is contained in this lowest cell, which has depth $i \geq 0$.

Now, move the point p up the line ℓ. As long as p remains in its cell, the top direction does not change; the region R is enlarged by this motion, but cannot intersect a cell of depth greater than i.

When p crosses a line of the arrangement, we may obtain a new top direction u'. Let W denote the convex wedge with apex p and directions u and u'. Applying the Wedge lemma to W, we see that no cell of depth greater than i intersects W.

If this new wedge W contains the vertically downward direction v (which may happen when u and u' point to different sides of ℓ), then we take the new region R from v to u', which is contained in W. Otherwise, we take the new R to be the union of R with W (this happens either when u and u' point to the same side of ℓ, or when the convex wedge between u and u' contains the vertical upward direction $-v$). In either case, R does not intersect a cell of depth greater than i. Finally, if W contains the upward direction $-v$, then the new R contains one of the halfplanes defined by ℓ and we may stop the algorithm.

Since the upward direction is the top direction for the uppermost cell, the algorithm must terminate. □

As an aside, one can use a similar argument along a curved path to show that the maximum depth cells are connected.

Corollary 8 *In an arrangement of lines in the plane, the closure of the cells of depth at least i is simply connected.*

Proof Consider a connected component of the union of the closures of cells of depth $\geq i$, and draw a path in the neighboring cells (which have depths $i-1$ and $i-2$). Applying the Wedge lemma as one traverses the path, as in the proof of Lemma 7, shows that no cell of depth $\geq i$ lies outside the path, so there can be only one component. Note that this component must be simply connected, since every point has a witness ray for depth along which the depth decreases monotonically. □

3.2 Computing Top Directions

In this section we describe a data structure that can determine the top directions for a sequence of adjacent cells in an arrangement of n lines using logarithmic time per cell, after $O(n \log n)$ preprocessing. Preprocessing takes linear time if the lines of the arrangement are sorted by slope.

Let us continue to assume that no line is vertical and let $l_1, l_2, \dots, l_n$ be the lines ordered by increasing slope. We can identify a cell C in the arrangement with its *bit string* $b(C) = b_1 \dots b_n$, where bit $b_i = 1$ if line l_i is above the cell C, and $b_i = 0$ otherwise.

Notice that the number of 1 bits in $b(C)$ is exactly the number of lines crossed by a ray ρ from C in the downward direction. Consider rotating the ray ρ from C counter-clockwise. The set of lines crossed by ρ does not change until ray ρ reaches the direction of the line l_1—then bit b_1 is complemented, since ρ will begin to intersect or cease to intersect l_1.

We therefore consider an *extended bit string* $B(C) = b(C)\overline{b(C)}b(C)$, which is the bit string for C, followed by its complement, and the bit string again. The extended string $B(C)$ has $2n+1$ contiguous subsequences of length n; we drop the last, since it equals the first. The counts of the number of 1 bits in these $2n$ subsequences give the number of lines intersected by a ray from C to the unbounded cells of the arrangement in the corresponding $2n$ directions. The minimum of these counts is the value $\mathrm{depth}(C)$.

With a relatively simple tournament we can maintain the minimum of the counts and information about directions in which the minimum occurs. We use a static, balanced, binary tree that stores in the leaves the sequence of $2n$ counts. The leftmost leaf stores the count for the upward direction. Each internal node stores three integers: the size of its subtree, the minimum count of the leaves in its subtree, and a correction value.

The correction value is a positive or negative integer that should be added to the counts of all leaves in the subtree. It is processed as follows: before the count of a node is inspected, the correction value is added to the count and to the correction values of the two children nodes, then set to zero. Since tree operations will process nodes from root to leaf, the value of inspected nodes will always be properly corrected.

The tree supports two operations: a query and an update. The query asks for the leaf with minimum count; in case of equal counts we want both the leftmost leaf and the rightmost leaf with these counts—these give the top directions for the cell C. Since each internal node stores the minimum count in its subtree, such a query is easy to perform in $O(\log n)$ time by following two paths in the tree.

The update operation corresponds to moving from a cell C to a cell C' by crossing some line l_i. This means that the bit string of $b(C')$ differs from $b(C)$ in the ith bit. In the extended string $B(C')$, three bits change to their complements. Since the $2n$ counts for a cell are obtained by adding n consecutive bits, every count changes—if b_i changes from 0 to 1, then the first i counts increase by one, the next n counts decrease by one, and the final $n-i$ counts increase by one. Thus, we should not update the counts in the leaves explicitly, since this would take linear time; instead we update correction values.

We follow the two paths in the tree to the ith leaf and the $(i+n)$th leaf using the size-of-subtree integers stored at the internal nodes. The paths partition the tree into three parts. For all highest nodes left of the search path to the ith leaf we increment the correction value (or decrement, if b_i changes from 1 to 0). This is done also for the highest nodes right of the search path to the $(i+n)$th leaf. For the highest nodes between the search paths we decrement (or increment) the correction value. Since there can be at most $O(\log n)$ highest nodes left (or right) of any path in the tree, only $O(\log n)$ correction values are updated.

Because the structure of the tree is static, we implement it by indexing into a fixed array, and subtree sizes are calculated rather than stored.

Lemma 9 *Using the data structure described above, one can determine the top directions for a sequence of adjacent cells in an arrangement of n lines using logarithmic time per cell, after $O(n \log n)$ preprocessing.*

3.3 Binary Search for a Maximum Depth Cell

It is probably no surprise that we use the sidedness test in a binary search on x-coordinates of vertices of the arrangement $\mathcal{A}(L)$. A Java prototype can be seen at `www.win.tue.nl/~speckman/demos/maxdepth`.

Standard results on slope selection [5, 10, 19, 22] allow us to consider the portion of the arrangement $\mathcal{A}(L)$ that lies between two vertical lines, and to generate the vertex of median x coordinate in $O(n \log n)$ time. We base our implementation on a randomized algorithm of Dillencourt, Mount, and Netanyahu [12].

At a vertical test line ℓ through (or, rather, slightly near) this median vertex, we sort the intersections with the lines of L and use the tournament described in Sect. 3.2 to compute the depth of each point on the test line ℓ and the top directions in $O(n \log n)$ time. Lemma 7 then allows us to discard one side of the line ℓ, and to continue the search on the other side. The search terminates when there are no intersection points remaining, which occurs after at most $\log(n^2) = 2\log n$ steps. Thus, we claim the following result.

Theorem 10 *A cell of maximum undirected depth in an arrangement of n lines can be computed in $O(n \log^2 n)$ time and $O(n)$ space.*

As remarked before, Langerman and Steiger improved this result to $O(n \log n)$ time [21]. They make use of the Wedge lemma presented here, but replace our sidedness test by a version that is more efficient: in recursive steps, a constant fraction of the lines can be pruned out.

4 The Structure of Depth

Although our binary search identifies a deepest cell, we know from Lemma 1 that the maximum depth in an arrangement will always occur at a vertex. In statistical analysis, we may also wish to know the set of all lines with maximum regression depth, which corresponds to the set of all points at maximum depth. In this section, we characterize the set of points at maximum depth in nondegenerate arrangements in the plane. We also establish relationships with k-sets in all dimensions and show how to efficiently approximate a maximum depth point in degenerate arrangements.

4.1 Finding a Deepest Vertex in a Nondegenerate Arrangement

Figure 1 showed an example in which edges and isolated vertices attain the maximum depth, but no cell does. Once we have found a point in a cell of maximum depth, we still must determine whether there is a vertex with greater depth. For arrangements of lines in general position, this is not difficult to do. When the maximum depth of a cell is i, then the maximum depth of a vertex is i, $i+1$, or $i+2$, as illustrated in Fig. 5. These cases can be detected by postprocessing after computing a maximum depth cell. Recall that an arrangement is nondegenerate if no line is vertical, no two lines are parallel, and no three lines pass through a single point.

When the maximum depth vertex v has depth $i+2$ in a nondegenerate arrangement, then the two lines crossing at v form four quadrants containing incident cells at depth i. Lemma 2 says that the directions for these cells are contained in the respective quadrants. During the binary search, test lines to the right of the vertex will eliminate their right side and those to the left will eliminate their left side. Thus, there is at most one such vertex and the binary search will find it.

The maximum depth vertex could instead have depth i—equal to the depth of the maximum depth cell. In this case, every vertex incident on a cell of depth i must also

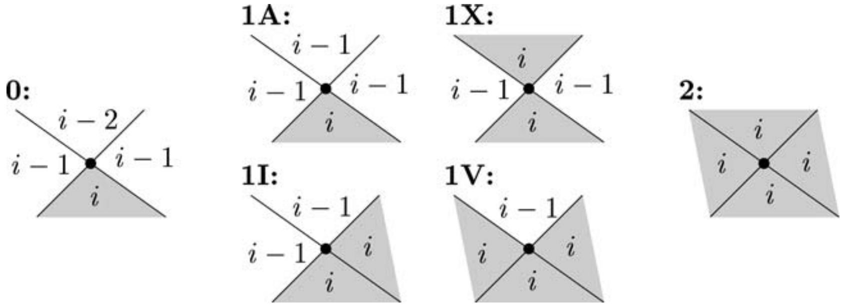


Fig. 5 Cases for maximum vertex depth

be incident on two cells of depth $i-1$ and one of depth $i-2$, otherwise the vertex depth would be greater than i, as illustrated in Fig. 5. This, together with the fact that cells are convex and the maximum depth is connected, implies that there can be only one cell that attains the maximum, which will be found by our binary search.

Finally if the maximum depth vertex has depth $i+1$, then every cell of depth i has to have at least one incident vertex of depth $i+1$. This follows, e.g., from the case analysis shown in Fig. 5, and from the fact that the set of points at depth $\geq i$ is connected. Since our binary search finds a cell of depth i a traversal of its boundary will yield a vertex of depth $i+1$.

Theorem 11 *After computing a deepest cell, one can compute a deepest vertex in $O(n \log n)$ additional time.*

Proof Once we have computed some cell of maximum depth i, we must determine whether a maximum depth vertex has depth i, $i+1$, or $i+2$. This is most easily done by constructing the cell as the intersection of the n halfplanes that are defined by lines of the arrangement and that contain the cell. Intersection is equivalent to convex hull computation, and takes $O(n \log n)$ time. Then we can use the tournament to check the depth of all vertices, also in $O(n \log n)$ time. By the above discussion, we either find that all vertices are of depth i, or there is a unique vertex of depth $i+2$, or some vertex is of depth $i+1$. □

4.2 Connections with k-Sets

It is natural to ask for the set of all points with maximum undirected depth, which corresponds to the set of all lines that have maximum regression depth. This appears to be more difficult; in this section we observe the connections between the complexity of points with given undirected depth and the concept of k-sets in a configuration of points. There has been considerable attention in computational geometry devoted to k-sets, and the dual concept of k-levels in an arrangement of lines or hyperplanes; see, e.g., [9, 11, 13, 23, 31].

The *k-level* of an arrangement $\mathcal{A}$ for a particular direction θ consists of all points p such that a ray from p in direction θ intersects exactly k hyperplanes. (Usually, hyperplanes containing p are not counted.) In the dual, the k intersected hyperplanes become a *k-set*: k points that can be separated from the configuration by an open half-space bounded by a hyperplane, namely p^D. Note that point p has undirected depth at most k (assuming that p does not lie on any hyperplane) and that the hyperplane p^D has regression depth at most k as shown by rotation about any line outside the convex hull of the dual points. The combinatorial complexity of k-levels and algorithms to compute them have been intensively studied, although many open problems remain.

In a similar manner, we define the *k-envelope* in an arrangement $\mathcal{A}$ to be the union of all points with undirected depth k. An example can be seen in Fig. 1. There have been some results on 1-envelopes of lines [16, 20], but we know of no deeper results.

We show that the worst-case combinatorial complexity of k-envelopes is asymptotically the same as the worst-case complexity of a k-level in any fixed dimension.

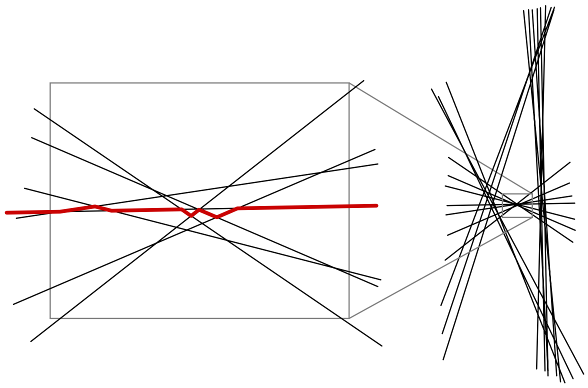


Fig. 6 Median level to maximum depth

The exact asymptotic worst-case complexity of a k-level is still unknown [11, 34]. In the plane, it known to be between $n \cdot 2^{\Omega(\sqrt{\log n})}$ and $O(n^{4/3})$.

We begin with the lower bounds that show that the complexity of a k-envelope is at least as great as that of a k-level.

Lemma 12 *The worst-case complexity of the k-envelope of an arrangement of n hyperplanes is at least as large as the worst-case complexity of a k-level in an arrangement of $n - dk$ hyperplanes, for $k < n/d$.*

Proof Consider the k-level in an arrangement of $n - kd > 0$ hyperplanes, none of which are parallel to the x_d axis. There is a unique unbounded cell in this arrangement that contains the vertically downward direction, θ. In this cell we can construct a simplex Δ with one horizontal face such that all rays through the horizontal face from the opposite vertex remain inside the cell. Scale and translate Δ until Δ contains the full complexity of the k-level. Then add to the arrangement k perturbed copies of the hyperplanes through each of the d nonhorizontal faces of Δ.

For points on the k-level, rays in the downward direction intersect k old hyperplanes and none of the new ones. Rays in directions outside the cell of the downward direction intersect at least k of the new hyperplanes. Thus, the k-level appears on the k-envelope. □

The construction above does not state what the combinatorial complexity is of the points with maximum depth of $k \approx n/d$. With another construction, illustrated in Fig. 6, we can show that the complexity of the points with maximum depth in the plane is lower bounded by the complexity of a median level.

Lemma 13 *The worst-case complexity of the set of points with maximum undirected depth in an arrangement of n lines is at least as large as the worst-case complexity of the median level in an arrangement of $n/3$ lines.*

Proof Consider any arrangement with $2m$ lines, none of which is parallel to the vertical y axis, and enclose it in a triangle with a vertical longest side, and two other nearly vertical sides. Add $2m$ lines through the longest side and m through each of the others, then perturb the new lines to be in general position.

Unbounded cells in the original arrangement now have undirected depth at most $2m$ by crossing only new lines. Bounded cells in the original arrangement also have undirected depth at most $2m$ by crossing m old lines and m new with a near-vertical ray. The former median level has undirected depth of exactly $2m$, and thus contributes points of maximum depth. □

4.3 Deepest Points in Nondegenerate Arrangements

We expand on the discussion given earlier in this section to characterize the whole set of maximum depth points in nondegenerate arrangements. As we just showed, this set can have superlinear complexity.

Lemma 14 *If the maximum cell depth is i, then the maximum depth points form either*

1. *a single point of depth $i + 2$;*
2. *a convex polygon whose vertices, edges, and interior all have depth i; or*
3. *a single chain of segments and some isolated points of depth $i + 1$, where either the single chain or the isolated points need not be present.*[1]

Proof The first and second cases are discussed in Sect. 4.1; we establish the structure of the third by considering the configurations of Fig. 5 that give vertices and edges of depth $i + 1$. If we consider the witness directions for cells of depth $i - 1$ adjacent to cells of depth i in these cases, and apply the Wedge lemma, we can make the following observations.

In configuration 1I, there is a wedge defined by directions for the two cells of depth $i - 1$ that includes a ray on the line separating these two cells. In configuration 1A, there are two such wedges. The Wedge lemma implies that cells in these wedges are of depth at most $i - 1$. This immediately implies that all edges in the wedge have depth at most i. In fact, vertices in the wedge also have depth at most i, since the only way for a vertex to have depth $i + 1$ would be to have four incident cells of depth $i - 1$, but then $i - 1$ would be the maximum depth of a cell in the arrangement.

In configuration 1X, we consider two witness directions for the cells of depth $i - 1$, and let them define rays that originate at the intersection point of configuration 1X. The rays define a wedge that contains one of the two incident cells of depth i. Translate the wedge slightly so that its apex is in the other cell of depth i. Cases (iv) and (v) of the Wedge lemma show that all cells and edges in the wedge have depth at most i. There may be isolated vertices of depth $i + 1$ in the wedge.

It is clear that configurations 1A and 1X give isolated vertices of depth $i + 1$, that 1I gives the end of a chain of vertices and edges of depth $i + 1$, and that 1V gives the middle of such a chain (see Fig. 7). We need to show that there is at most one chain,

[1]The conference version claimed that the chain has $O(n)$ segments, but our proof of that claim turned out to be incorrect.

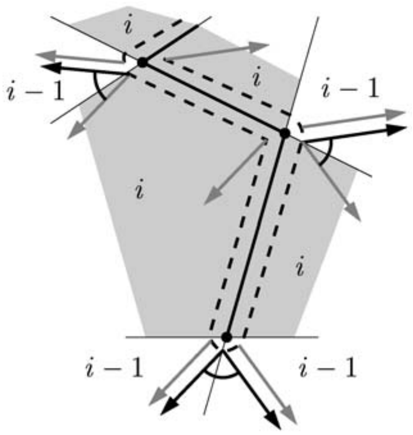


Fig. 7 A chain of maximum depth

so assume there is some chain. We construct a path enclosing this chain by infinitesimally translating copies of each segment into its adjacent cells and connecting the endpoints (see Fig. 7). Every point on this cycle has at least one witness ray of depth at most i and the union of these rays cover the whole plane except the original chain. This certifies—again by the Wedge lemma—that no edge of depth $i+1$ exists that is not part of the original chain. □

We can go further to characterize the isolated points that are at maximum depth $i+1$: they appear as the connections for stings of cells of depth i, and are *antipodal*, meaning that they are the points of tangency for parallel tangent lines.

Lemma 15 *Let c be a cell of maximum depth i, let c' be an edge-adjacent cell of depth $i-1$, and let r be a witness ray for c'. Then any cell c'' that is edge-adjacent to c and for which no translated ray $\bar{r}$ of r originating in c'' intersects c, has depth $i-1$ with $\bar{r}$ as a witness ray.*

Proof Since c'' is edge-adjacent to c, it must have depth at least $i-1$. If c' and c'' are as in the lemma and vertex-adjacent, then obviously $\bar{r}$ intersects exactly one line that r does not intersect, and r intersects one line that $\bar{r}$ does not intersect. By induction the same holds if c' and c'' are not vertex-adjacent; see Fig. 8. Since a witness ray for c'' exists that intersects $i-1$ lines, c'' has depth at most $i-1$, and the lemma follows. □

Lemma 16 *Assume that the maximum cell depth is i, and the maximum depth is realized by one or more vertices of depth $i+1$ (no edge has depth $i+1$). If two vertices of depth $i+1$ are incident to the same depth i cell, then they are antipodal. At most one cell has three incident vertices of depth $i+1$, and all other cells have fewer vertices of depth $i+1$.*

Proof Let i be the depth of a deepest cell. Clearly, a cell of depth $i-1$ cannot be edge-adjacent to three cells of depth i in such a way that the bounded triangle formed

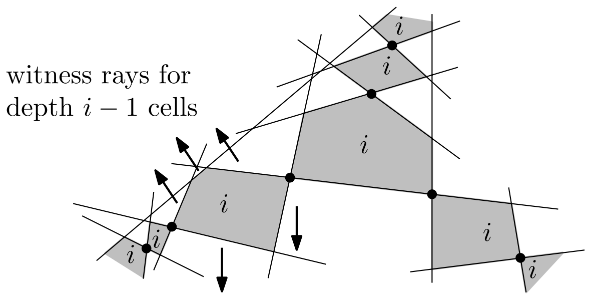


Fig. 8 Antipodal vertices of depth $i+1$ incident to depth i cells

by the three lines supporting these three edges contains the cell of depth $i-1$. A consequence is that any two vertices in 1X configuration and incident to the same cell of depth i are antipodal in that cell, otherwise we will have a contradiction with Lemma 15. Therefore, a cell of depth i can have at most three vertices of depth $i+1$ in 1X configuration, and these vertices are pairwise antipodal.

In a 1X configuration, there are two vertex-adjacent depth $i-1$ cells with different witness directions. If rays with these directions are placed on the 1X vertex, then they form a wedge that encloses exactly one of the two depth i cells. For that cell, one of the two directions will apply for each of its edge-adjacent depth $i-1$ cells by Lemma 15. Therefore, it can have at most one other vertex in 1X configuration, and if there is another 1X vertex, then the lines parallel to the two witness rays must be tangent to the cell at the vertex. By the same argument, a cell with three vertices in 1X configuration cannot be contained in any of the three wedges formed at the 1X vertices.

From these structure observations it follows that there can be at most one depth i cell that has three vertices in 1X configuration. The three vertex-adjacent depth i cells can each have at most one more vertex in 1X configuration, so the vertices of depth $i+1$ occur on three "paths" of depth i cells; see Fig. 8. □

Lemma 17 *Assume that the maximum cell depth is i, and the maximum depth of a feature realized by a chain of depth $i+1$ and zero or more isolated vertices. Then any cell of depth i sharing some edge with the chain must have all edges of the chain consecutive in its boundary. Every isolated vertex of depth $i+1$ is antipodal to another isolated vertex or to a chain vertex of depth $i+1$.*

Proof If the chain would coincide with the boundary of a depth i cell c more than once, then there exists a cell that is enclosed by the union of the chain and cell c. The witness direction of that cell must cross the chain or cell c, and therefore would have depth $> i$, a contradiction.

The antipodality property follows in a similar way as in Lemma 16. If a depth i cell c is incident to at least one edge of the depth $i+1$ chain, then the witness directions in the edge-adjacent depth $i-1$ cells certify that there can only be one isolated depth $i+1$ vertex incident to cell c. □

4.4 Output-Sensitive Construction for Maximum Depth in Nondegenerate Planar Arrangements

A dynamic convex hull maintenance algorithm, when applied to the duals of the lines, allows us to maintain a description of the current cell as we walk from cell to cell in the arrangement. With the characterization of the points of maximum depth from Sect. 4.3, this allows us to compute a description of the maximum depth points in an output-sensitive manner.

Theorem 18 *After $O(n\log^2 n)$ preprocessing, the set of all edges and vertices at maximum depth in an arrangement of lines in general position can be computed at the cost of $O(\log^{3/2} n)$ time per feature.*

Proof For the preprocessing, use the result of Theorem 10 to find a deepest cell. Build the tournament structure for this cell (Lemma 9). Let H be the set of half-planes with the lines as bounding lines, such that the deepest cell that was found is $\bigcap_{h\in H} h$. Dualize the lines bounding positive half-planes to points and build a lower convex hull maintenance structure for them. We choose the one of Chan [6], which allows all necessary operations in $O(\log^{3/2} n)$ time. Similarly, build an upper convex hull maintenance structure for the points dual to lines that bound negative half-planes. The convex hull maintenance structures allow us to do a search on the boundary of a deepest cell. We can switch to an adjacent cell in $O(\log^{3/2} n)$ time by updating all three data structures. Therefore, we can decide in $O(\log^{3/2} n)$ time what the depth of an incident edge or vertex is, and also what the type of a vertex is. We can also do linear programming queries (find extreme vertices for a direction) in cells of the arrangement in $O(\log^{3/2} n)$ time, which are the dual of vertical line intersection queries with the convex hull.

We note that our algorithm to find a deepest cell will find a cell of depth i that is adjacent to the depth $i+1$ chain, if it exists. Also, it will find the cell with three incident antipodal vertices of depth $i+1$, if such a cell exists. So we can determine in $O(n\log^{3/2} n)$ additional time whether case 3 of Lemma 14 applies, by inspecting the whole boundary of the cell, or get the three depth $i+1$ vertices. Furthermore, using the data structures, we can easily obtain the whole depth $i+1$ chain in time $O(k\log^{3/2} n)$ if it has k edges.

When we check any other depth i cell by continuing over a depth $i+1$ edge or over the lines intersecting in a vertex in 1X configuration, we cannot examine the whole boundary and achieve the claimed time bound. Instead we will perform a search in the cell to find any other features of depth $i+1$. This will be done using the witness directions of adjacent depth $i-1$ cells (which was already suggested by the proof of Lemma 16).

Let v be a known depth $i+1$ vertex of a cell c of depth i, and assume that it is isolated. Then v is incident to two depth $i-1$ cells, and the rays with their witness directions, when originating in v, form a wedge that contains c. If v is part of the depth $i+1$ chain, then we get the witness directions at the most clockwise and counterclockwise depth $i+1$ vertices of the chain. When placed at these vertices, the rays with the witness directions and the subchain together form an unbounded convex

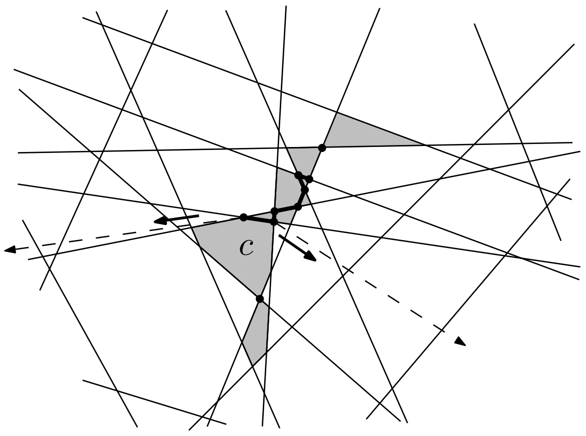


Fig. 9 A depth $i+1$ chain (*fat*), two depth $i+1$ isolated vertices, a cell c of depth i, and the two witness directions (*fat arrows*) for the search illustrated

polygon that contains c. Figure 9 shows an example with two (dashed) rays and one edge (fat) of the chain.

To find the at most one other vertex of depth $i+1$—in 1X configuration—we perform a linear programming query with each of the two witness rays. If they find the same vertex in c, then this vertex may be in 1X configuration, and the vertex-adjacent cell would then have depth i. We can test this using our data structures in $O(\log^{3/2} n)$ time. If they find different vertices, then the cell has no other depth $i+1$ vertices due to Lemma 15 and the fact that the initial witness rays form a wedge that contains the cell (when the rays are placed at the known depth $i+1$ vertex). In case we find another depth $i+1$ vertex, the search proceeds in the vertex-adjacent depth i cell with the same two witness rays.

After processing all depth i cells we have found all depth $i+1$ features in at most $O(\log^{3/2} n)$ time per feature. □

It may be possible to use a more efficient dynamic convex hull maintenance structure like the one of Brodal and Jacob [4], but it is unclear if all necessary operations that we need can be performed more efficiently using their data structure.

4.5 Depth of Vertices in Degenerate Arrangements

Efficiently finding a deepest vertex in a degenerate arrangement of lines appears to be difficult. However, we can efficiently find a vertex whose depth is within a factor of $(1-o(1))$ from the maximum depth.

Lemma 19 *A point whose depth is at least* $(1-\frac{\log(\log n)}{\log n})$ *times the maximum can be found in* $O(n\log n)$ *time.*

Proof First, compute the cell of maximum depth in the arrangement. Then, using an algorithm of Guibas et al. [17], find all vertices V that are contained in at least $n\frac{3\log(\log n)}{\log n}$ lines in $O(n\log n)$ time. There are at most $O(\frac{\log n}{\log(\log n)})$ of these vertices, and their depth can be tested in $O(n)$ time each once the lines are sorted by slope.

Either a vertex of V has maximum depth, or, by Lemma 1, a point in the cell of maximum depth is less than $n\frac{3\log(\log n)}{\log n}$ from the true maximum value. Since Amenta et al. proved in [3] that the maximum value is at least $\lceil n/3\rceil$ we therefore have an approximation factor of at least $(1-\frac{\log(\log n)}{\log n})$. □

One heuristic that involves less programming is to symbolically perturb the lines of the arrangement to simulate general position and compute the cell of maximum depth. In the original arrangement this cell may correspond to a vertex, in which case we evaluate the depth of this vertex, or to a cell, in which case we construct the cell and evaluate the depth of all of its vertices. From the Wedge lemma it can be seen that the actual maximum depth will be at most double the computed depth.

5 Computing Depth in Higher Dimensions

For three and higher dimensions, Corollary 4 states that we can compute the maximum depth in $O(n^d)$ time and space by evaluating depth at all cells and vertices of an arrangement. It is challenging to develop more efficient algorithms.

5.1 The Wedge Lemma Cannot Extend to $\Re^3$

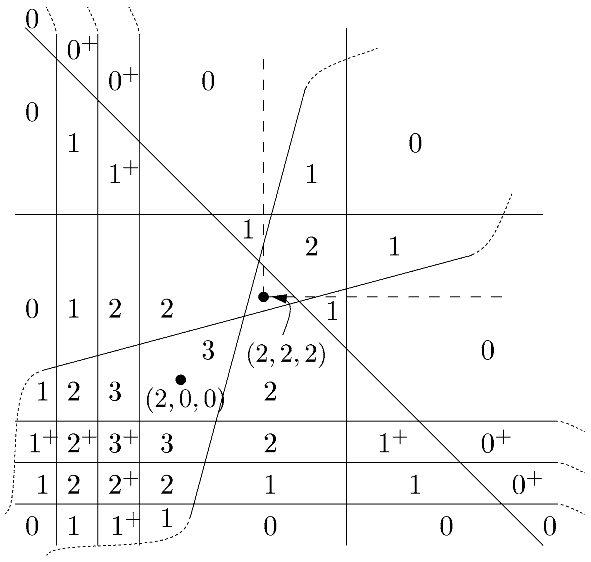
The solution for the planar case was based on the Wedge lemma, which allowed us to argue that certain regions of the plane could not contain a cell of maximum depth. When thinking about the extension to three dimensions, one would first try to generalize the Wedge lemma: that for a point p whose depth i is witnessed by three vectors $\vec{u}$, $\vec{v}$, and $\vec{w}$, the cone defined by $\vec{u}$, $\vec{v}$, and $\vec{w}$ does not contain a cell of depth greater than i. The following construction shows that this is not true.

Let point p be the origin of the coordinate system. We construct an arrangement of 15 planes such that the positive x-axis, the positive y-axis, and the positive z-axis each witness that depth$(p)=2$, the point $q=(2,2,2)$ will have depth$(q)=3$.

There are six planes that intersect the positive octant: Planes $x=4$, $y=4$, and $z=4$ are parallel to the coordinate planes. Planes $5=-x-y+5z$, $5=-x+5y-z$, and $5=5x-y-z$ pass through a common point $(5/3,5/3,5/3)$, and each intersect one of the positive coordinate axes. Note that the first intersects the z axis at $(0,0,1)$ and the x and y axes at $(-5,0,0)$ and $(0,-5,0)$. Note that if the coordinate frame is translated from the origin to $q=(2,2,2)$, then each positive axis intersects *three* of these six planes, which already shows that the argument used to prove the two-dimensional Wedge lemma does not hold in the three-dimensional case.

The remaining nine planes are chosen to make sure that only directions in or near the positive octant can give depth counts below three for all cells in the positive octant. They are perturbed versions of $x=-1$, $x=-2$, $x=-3$ and similarly, $y,z=-1,-2,-3$. The perturbations are such that none of the planes intersect the positive octant. The common intersection of the half-spaces bounded by these planes and containing the origin can be seen as the perturbed positive octant. These make sure that for any point in the positive octant, and any direction outside the positive octant by a small angle, the depth count in that direction will be at least three.

Fig. 10 Cross-section of nine planes with the plane $x=2$, with depth values for directions in this plane. Plus signs denote a possible increment by one due to perturbation



Let us first consider the number of planes intersected by rays from q inside the positive quadrant of the plane $x=2$ (which itself is not one of the six planes). By constructing a figure of this cross-section, one can easily verify that all of these directions give rays intersecting three or four planes; see Fig. 10. The perturbation of the planes does not influence the depth of the cell containing q.

Finally, when we consider directions from q where x, y, z-contributions are all strictly positive, we simply observe that any such direction intersects each one of $x=4$, $y=4$, and $z=4$. Thus, the depth$(q)=3$.

This example also shows that the closures of cells of a particular depth need not be connected. The proof of the Wedge lemma does imply that the positive quadrants of the three coordinate planes do not intersect cells of depth greater than two—if we translate a pair of positive coordinate axes within the quadrant that they define, we do not gain new intersections. Thus, Corollary 8 cannot be extended beyond the plane.

5.2 Computing Depth in Higher Dimensions with Reduced Space

We close this paper by showing how to reduce the space requirement by a linear factor using hyperplane cuttings.

Theorem 20 *For $d \geq 3$, one can compute the depth of all cells of a set of hyperplanes in $\Re^d$ in time $O(n^d)$, using $O(n^{d-1})$ space.*

Proof Let H be a set of n hyperplanes. For parameter $r=n^{1/d}$, a $(1/r)$-cutting of H is a set of $O(r^d)=O(n)$ simplices covering $\Re^d$, each intersecting at most n/r hyperplanes. It requires $O(nr^{d-1})$ time to compute the cutting, using a result of Chazelle [8].

We consider the subproblem for each simplex τ and its intersecting hyperplanes H_τ. If we considered only the hyperplanes in H_τ, then by Corollary 4 we

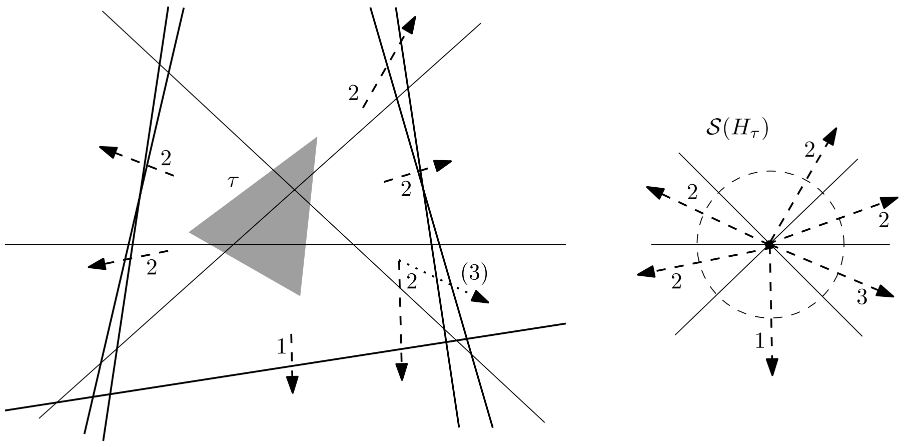


Fig. 11 A triangle (simplex) τ, three planes in H_τ, five lines in $H \setminus H_\tau$ (thicker), and the directions giving the lowest depth for the six unbounded cells. To the *right*, weights for certain directions (corresponding to directions in the *left figure*) in the cells of $\mathcal{S}(H_\tau)$

could solve the subproblem for τ by building the arrangement $\mathcal{A}(H_\tau)$ in $\Re^d$. This takes $O(|H_\tau|^d)$ time and space, which is $O(n^{d-1})$ since $|H_\tau| \le n/r$. We can modify this algorithm to capture the additional depth caused by the other hyperplanes. Since hyperplanes of $H \setminus H_\tau$ do not intersect τ, we can translate them away from τ until each one has all vertices of $\mathcal{A}(H_\tau)$ (and τ itself) to one side, without changing the depth of any point inside τ. We can then compute the depth of all unbounded cells of $\mathcal{A}(H_\tau)$ with respect to the translated hyperplanes in $H \setminus H_\tau$, and use this as the base for computing the depth of all cells in $\mathcal{A}(H_\tau)$.

More precisely, consider the (hyper)sphere of directions. Each hyperplane in H induces a great circle on this sphere, and their arrangement, which we call $\mathcal{S}(H)$, has $O(n^{d-1})$ cells. Choosing one direction in each cell of $\mathcal{S}(H)$ gives a set $\{\rho_1, \rho_2, \dots, \rho_k\}$ of $k = O(n^{d-1})$ directions sufficient to witness the maximum depth for any point $p \in \Re^d$.

We return to the subproblem for simplex τ and its hyperplanes $H_\tau \subset H$. For each direction ρ_i, define a weight w_i that equals the number of planes of $H \setminus H_\tau$ intersected by a ray from τ in direction ρ_i; see Fig. 11. It does not matter which point of τ is the ray origin for this definition, since planes of $H \setminus H_\tau$ do not intersect τ.

We can compute the arrangement $\mathcal{S}(H_\tau)$, and a direction with lowest weight for each cell of $\mathcal{S}(H_\tau)$ in two steps. First, compute weights for all directions in $\mathcal{S}(H)$ with respect to hyperplanes of $H \setminus H_\tau$ by choosing a cell with its direction ρ_i, counting the number of hyperplanes intersected by a ray in direction ρ_i, then traversing the remaining cells of $\mathcal{S}(H)$, adding or subtracting one each time we cross the great circle for a hyperplane of $H \setminus H_\tau$. Second, delete hyperplanes of $H \setminus H_\tau$ from $\mathcal{S}(H)$ to form arrangement $\mathcal{S}(H_\tau)$. Whenever two cells merge, keep a direction with lowest weight. The time for each subproblem is proportional to $|\mathcal{S}(H)| = O(n^{d-1})$, for a total of $O(n^d)$ time overall.

Now, form the hyperplane arrangement $\mathcal{A}(H_\tau)$, and initially label all bounded cells with depth ∞. Each unbounded cell of $\mathcal{A}(H_\tau)$ corresponds to a cell of the spherical arrangement $\mathcal{S}(H_\tau)$; label it with the corresponding weight and direction. The minimum assigned depth is correct, so we can perform the loop as before: for $i = 1, 2, \ldots, n$, all cells with label $i - 1$ cause their adjacent, higher-labeled cells to be relabeled i. This takes $O(n^{d-1})$ time and space for each simplex τ, leading to a total time of $O(n^d)$. □

6 Conclusions and Open Problems

This paper presented an $O(n \log^2 n)$ time algorithm for computing a point with maximum regression depth in the plane. Langerman and Steiger used our algorithm as a basis for their improved $O(n \log n)$ time algorithm [21]. We also gave an $O(n^d)$ time, $O(n^{d-1})$ space algorithm for maximum regression depth in higher dimensions and various other results.

The most interesting open problems that remain are a more efficient algorithm for computing regression depth in higher dimensions, and good approximation algorithms in the plane and higher dimensions.

Acknowledgements J. Mitchell and M. Sharir thank E. Arkin and S. Har-Peled for several helpful discussions and suggestions. We also thank an anonymous referee for useful comments.

References

1. Aelst, S.V., Rousseeuw, P.J.: Robustness of deepest regression. J. Multivar. Anal. **73**, 82–106 (2000)
2. Aelst, S.V., Rousseeuw, P.J., Hubert, M., Struyf, A.: The deepest regression method. J. Multivar. Anal. **81**(1), 138–166 (2002)
3. Amenta, N., Bern, M., Eppstein, D., Teng, S.-H.: Regression depth and center points. Discrete Comput. Geom. **23**, 305–323 (2000)
4. Brodal, G.S., Jacob, R.: Dynamic planar convex hull. In: Proc. 43rd Symposium on Foundations of Computer Science (FOCS), pp. 617–626 (2002)
5. Brönnimann, H., Chazelle, B.: Optimal slope selection via cuttings. Comput. Geom. Theory Appl. **10**(1), 23–29 (1998)
6. Chan, T.M.: Dynamic planar convex hull operations in near-logarithmic amortized time. J. ACM **48**(1), 1–12 (2001)
7. Chan, T.M.: An optimal randomized algorithm for maximum turkey depth. In: Proc. 15th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 430–436 (2004)
8. Chazelle, B.: Cutting hyperplanes for divide-and-conquer. Discrete Comput. Geom. **9**(2), 145–158 (1993)
9. Chazelle, B., Preparata, F.P.: Halfspace range search: An algorithmic application of k-sets. Discrete Comput. Geom. **1**, 83–93 (1986)
10. Cole, R., Salowe, J., Steiger, W., Szemerédi, E.: An optimal-time algorithm for slope selection. SIAM J. Comput. **18**(4), 792–810 (1989)
11. Dey, T.K.: Improved bounds on planar k-sets and related problems. Discrete Comput. Geom. **19**, 373–382 (1998)
12. Dillencourt, M.B., Mount, D.M., Netanyahu, N.S.: A randomized algorithm for slope selection. Int. J. Comput. Geom. Appl. **2**, 1–27 (1992)
13. Edelsbrunner, H.: Algorithms in Combinatorial Geometry. Springer, New York (1987)
14. Edelsbrunner, H., O'Rourke, J., Seidel, R.: Constructing arrangements of lines and hyperplanes with applications. SIAM J. Comput **15**, 341–363 (1986)

15. Edelsbrunner, H., Seidel, R., Sharir, M.: On the zone theorem for hyperplane arrangements. SIAM J. Comput. **22**(2), 418–429 (1993)
16. Eu, D., Guévremont, E., Toussaint, G.T.: On envelopes of arrangements of lines. J. Algorithms **21**, 111–148 (1996)
17. Guibas, L.J., Overmars, M.H., Robert, J.-M.: The exact fitting problem for points. Comput. Geom. Theory Appl. **6**, 215–230 (1996)
18. Hubert, M., Rousseeuw, P.J.: The catline for deep regression. J. Multivar. Anal. **66**, 270–296 (1998)
19. Katz, M.J., Sharir, M.: Optimal slope selection via expanders. Inf. Process. Lett. **47**, 115–122 (1993)
20. Keil, M.: A simple algorithm for determining the envelope of a set of lines. Inf. Process. Lett. **39**, 121–124 (1991)
21. Langerman, S., Steiger, W.: The complexity of hyperplane depth in the plane. Discrete Comput. Geom. **30**(2), 299–309 (2003)
22. Matoušek, J.: Randomized optimal algorithm for slope selection. Inf. Process. Lett. **39**, 183–187 (1991)
23. Peck, G.W.: On k-sets in the plane. Discrete Math. **56**, 73–74 (1985)
24. Rousseeuw, P.J., Aelst, S.V., Hubert, M.: Regression depth: Rejoinder. J. Am. Stat. Assoc. **94**, 419–433 (1999)
25. Rousseeuw, P.J., Hubert, M.: Depth in an arrangement of hyperplanes. Discrete Comput. Geom. **22**, 167–176 (1999)
26. Rousseeuw, P.J., Hubert, M.: Regression depth. J. Am. Stat. Assoc. **94**, 388–402 (1999)
27. Rousseeuw, P.J., Ruts, I.: Constructing the bivariate Tukey median. Stat. Sin. **8**, 827–839 (1998)
28. Rousseeuw, P.J., Struyf, A.: Computing location depth and regression depth in higher dimensions. Stat. Comput. **8**, 193–203 (1998)
29. Rousseeuw, P.J., Struyf, A.: Computation of robust statistics: depth, median, and related measures. In: Goodman, J.E., O'Rourke, J. (eds.) The Handbook of Discrete and Computational Geometry, 2nd edn., pp. 1279–1292. Chapman & Hall/CRC, Boca Raton (2004)
30. Ruts, I., Rousseeuw, P.J.: Computing depth contours of bivariate point clouds. Comput. Stat. Data Anal. **23**, 153–168 (1996)
31. Sharir, M.: k-sets and random hulls. Combinatorica **13**, 483–495 (1993)
32. Struyf, A., Rousseeuw, P.J.: Halfspace depth and regression depth characterize the empirical distribution. J. Multivar. Anal. **69**, 135–153 (1999)
33. Struyf, A., Rousseeuw, P.J.: High-dimensional computation of the deepest location. Comput. Stat. Data Anal. **34**, 415–426 (2000)
34. Tóth, G.: Point sets with many k-sets. Discrete Comput. Geom. **26**, 187–194 (2001)