Computing Batched Depth Queries and the Depth of a Set of Points

Stephane Durocher* Alexandre Leblanc† Sachini Rajapakse*

Abstract

Simplicial depth and Tukey depth are two common measures for expressing the depth of a point q relative to a set P of points in $\mathbb{R}^d$. We introduce definitions that generalize these notions to express the depth of a set Q of points relative to a set P of points in $\mathbb{R}^d$, and we examine algorithms for computing these in $\mathbb{R}^2$, capitalizing on the relative cardinalities of P and Q.

1 Introduction

Depth measures quantify the centrality of an object relative to a set of objects. For univariate quantitative data, a natural definition for the depth of a point q relative to a set P of points in $\mathbb{R}$ is to measure how deeply nested q is in P by the lesser of the number of points of P less than q, and the number of points of P greater than q. By this measure, outliers relative to P have low depth, whereas a median of P has maximum depth. Various generalizations to higher dimensions exist, including simplicial depth and Tukey depth.

The *simplicial depth* of a query point q relative to a set P of points is the number of simplices determined by points in P that contain q:

Definition 1.1 (Simplicial depth [14]) *Given a set P of n points in $\mathbb{R}^d$ and a point q in $\mathbb{R}^d$, the* simplicial depth *of q relative to P is*

$$SD_P(q) = \sum_{S \in \mathcal{S}} I(q \in S), \tag{1}$$

where $\mathcal{S}$ denotes the set of $\binom{n}{d+1}$ closed simplices, each of which is the convex hull of $d+1$ points from P, and I is an indicator function such that $I(A) = 1$ if A is true and $I(A) = 0$ otherwise.

The *Tukey depth* of a query point q relative to a set P of points is the minimum number of points of P in any closed half-space containing q:

Definition 1.2 (Tukey depth [19]) *Given a set P of n points in $\mathbb{R}^d$ and a point q in $\mathbb{R}^d$, the* Tukey depth *(or* half-space depth*) of q relative to P is*

$$TD_P(q) = \min_{\substack{H \in \mathcal{H} \\ H \cap q \neq \varnothing}} |H \cap P|, \tag{2}$$

where $\mathcal{H}$ is the set of all closed half-spaces in $\mathbb{R}^d$.

Given q and P in $\mathbb{R}^2$, the simplicial depth and the Tukey depth of q relative to P can be computed in $O(n \log n)$ time, respectively, where $n = |P|$ [9, 17, 10], both of which have matching lower bounds of $\Omega(n \log n)$ worst-case time [1].

A *depth median* of a set P is a point of maximum depth relative to P for a given depth measure. We refer to a *simplicial median* and *Tukey median*, which can be computed in $O(n^4)$ time [2] and $O(n \log^3 n)$ time [12] in $\mathbb{R}^2$, respectively. An *in-sample median* of P is a point of P with maximum depth, which can be identified in $O(n^2)$ time in $\mathbb{R}^2$ for simplicial depth [9].

Depth measures are typically defined to describe the location of a single query point (an individual) relative to a set of points (a population). In this paper, we examine (1) computing a batch of depth queries relative to a common set of points, and (2) deriving a single estimator for the depth of a set of query points relative to another set of points.

Computing a batch of depth queries by iteratively running an algorithm designed to calculate the depth of a single query point can be inefficient. To address this, we present algorithms to compute a batch of depth queries; the choice of which algorithm to apply to minimize running time depends on the relative cardinalities of the query point set Q to the input point set P. Defining and evaluating the depths of a set of query points has various applications in data analysis, e.g., finding a center-outward ordering of a set Q relative to a set P. Next, we derive a single estimator to express the depth of Q relative to P. Applications include (1) measuring the centrality of Q relative to P (e.g., the position of one soccer team relative to the opposing team), (2) classifying a set Q selected from the same distribution as the sets $P_1, \ldots, P_m$ to determine within which set P_i the set Q is most deeply contained.

Our results In Section 3.1 we present three algorithms for computing a batch of k simplicial depth queries in $\mathbb{R}^2$ in $O(kn \log n)$ time, $O(n^2 + nk)$ time, and $O(n^4 + k \log n)$, respectively. The first algorithm is fastest when

*Department of Computer Science, University of Manitoba, stephane.durocher@umanitoba.ca, rajapak1@myumanitoba.ca

†Department of Statistics, University of Manitoba, alex.leblanc@umanitoba.ca

$k \in O(\frac{n}{\log n})$, the second when $k \in \Omega(\frac{n}{\log n})$ and $k \in O(n^3)$, and the third when $k \in \Omega(n^3)$. In Section 3.2 we present two algorithms for computing a batch of k Tukey depth queries in $\mathbb{R}^2$ in $O(kn \log n)$ time and $O(n^2 + k \log n)$ time, respectively. The first algorithm is fastest when $k \in O(\frac{n}{\log n})$, and the second when $k \in \Omega(\frac{n}{\log n})$.

In Section 4 we introduce definitions for the simplicial depth and Tukey depth of a set Q relative to a set P, which can be computed in $\mathbb{R}^2$ by applying the algorithms above. Finally, we examine properties and probabilistic interpretations for the simplicial depth of a set of points.

2 Related Work

2.1 Simplicial Depth and Simplicial Median

Multiple algorithms compute the simplicial depth of a point q relative to a set P of n points in $\mathbb{R}^2$ in $O(n \log n)$ time [9, 17, 10]. Given the radial ordering of P around q, the simplicial depth of q can be computed in $O(n)$ time [9]. Given a set $P = \{p_1, \ldots, p_n\}$ of points in $\mathbb{R}^2$, Lee and Ching [13] showed that the radial order of $P \setminus \{p_i\}$ with respect to p_i for all $i \in \{1, \ldots, n\}$ can be determined in $O(n^2)$ time. Consequently, the simplicial depths of all points in P can be obtained in $O(n^2)$ time [9], and an in-sample simplicial median can be identified in $O(n^2)$ time [9]. Khuller and Mitchell studied a similar problem independently [10].

When defined in terms of closed simplices, a simplicial median lies at an intersection of simplex boundaries [2]. Rousseeuw and Ruts described how to find a simplicial median by searching the set of intersection points in $O(n^5 \log n)$ time [17]. Aloupis et al. [2] derived a faster algorithm to compute a simplicial median in $O(n^4 \log n)$ time, which they further reduced to $O(n^4)$ time. We apply a technique similar to that of Aloupis et al. [2] in Algorithm S.III in Section 3.1.

2.2 Tukey Depth and Tukey Median

The Tukey depth of a point q relative to a set P of n points in $\mathbb{R}^2$ can be computed in $O(n \log n)$ time [17]. Tukey depth contours are a collection of nested polygons that partition the plane into regions of equal Tukey depth, which can be computed in $O(n^2)$ time [15]. A Tukey median can be found in $O(n \log^3 n)$ time [12].

2.3 Depth of a Set of Points

Recently, Pilz and Schnider introduced a definition for the Tukey depth of a set of points [16]:

Definition 2.1 (Generalized Tukey depth [16]) *The* generalized Tukey depth *of a set $Q \subseteq \mathbb{R}^d$ with respect to a set $P \subseteq \mathbb{R}^d$ is*

$$GTD_P(Q) = \min_{\substack{H \in \mathcal{H} \\ Q \cap H \neq \varnothing}} \frac{|H \cap P|}{|H \cap Q|}, \tag{3}$$

where $\mathcal{H}$ is the set of all closed half-spaces in $\mathbb{R}^d$.

Definition 2.1 differs from our Definition 4.2 introduced in Section 4.2. Definition 2.1 selects a single non-empty half-space that minimizes the ratio (3), i.e., the number of points of P in the half-space H relative to the number of points of Q in H. On the other hand, Definition 4.2 incorporates the respective Tukey depths for each point in Q, i.e., different half-spaces may be selected for each point.

Depth histograms provide a characterization of the combinatorial structure of a point set [6, 4]. Bertschinger et al. studied Tukey depth histograms of k-flats [4] and defined variations of Tukey depth for a set Q relative to P, including affine Tukey depth and convex Tukey depth.

Recently, Barba et al. [3] introduced a definition for the cardinal simplicial depth[1] of a set of points:

Definition 2.2 (Cardinal simplicial depth [3]) *The* cardinal simplicial depth *of a set $Q \subseteq \mathbb{R}^d$ with respect to a set $P \subseteq \mathbb{R}^d$ is*

$$CSD_P(Q) = \sum_{S \in \mathcal{S}} I(Q \cap S \neq \varnothing), \tag{4}$$

where $\mathcal{S}$ denotes the set of $\binom{n}{d+1}$ closed simplices, each of which is the convex hull of $d+1$ points from P, and I is an indicator function such that $I(A) = 1$ if A is true and $I(A) = 0$ otherwise.

Definition 2.2 differs from our Definition 4.1 introduced in Section 4.1. Definition 2.2 counts the number of non-empty simplicies (the cardinality of the set of non-empty simplicies), whereas Definition 4.1 is a normalized sum of the number of points of Q contained in each simplex. See further discussion in Section 4.1. Barba et al. gave an algorithm to compute $CSD_P(Q)$ for given sets P and Q in $O(N^{7/3} \log^{O(1)} N)$ time, where $N = |P| + |Q| = n + k$.

3 Computing a Batch of Depth Queries

In this section, we describe algorithms that compute a batch of simplicial depth queries or Tukey depth queries for k points in a set Q relative to a set P of n points, where $P \cup Q$ is in general position in $\mathbb{R}^2$. For simplicial depth we propose three algorithms: Algorithm S.I is not

[1] To disambiguate between Definitions 2.2 and 4.1, we refer to Definition 2.2 as the *cardinal simplicial depth* because it corresponds to the cardinality of the set of non-empty simplicies.

new [9, 17, 10]; Algorithms S.II and S.III are new. For Tukey depth we propose two algorithms: Algorithms T.I and T.II apply techniques used in existing algorithms for Tukey depth and Tukey depth contours.

3.1 Computing a Batch of Simplicial Depth Queries

3.1.1 Algorithm S.I

In Section 2.1 we mentioned algorithms for computing the simplicial depth of a single query point q relative to a set P of n points in $\mathbb{R}^2$ in $O(n \log n)$ time [9, 17, 10]. When the number of query points k is small relative to n, a straightforward approach for computing the depths of k points is to iteratively compute the simplicial depth of each query point using one of these existing algorithms. Using this approach, we can compute the simplicial depth of all k points in $O(kn \log n)$ time and $O(n)$ space to store the angular order of P around each query point (this space is reused for each query point). Due to the lower bound of $\Omega(n \log n)$ on the worst-case time required for computing the simplicial depth of a single point [1], this approach is optimal when $k \in O(1)$.

Lemma 1 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, Algorithm S.I computes $SD_P(q)$ for every $q \in Q$ in $O(kn \log n)$ total time and $O(n + k)$ space.*

3.1.2 Algorithm S.II

Algorithm S.I is efficient when k is small relative to n, but more efficient approaches are possible for larger values of k. We describe an algorithm that computes the simplicial depths of points in Q relative to P in $O(n^2 + nk)$ time and $O(n^2)$ space. Using an approach similar to the in-sample simplicial median algorithm of Gil et al. [9] (Step 1), we compute the radial order of the n points of P around each point in Q, and (Step 2) we use this ordering to compute the simplicial depth of each point in Q.

To perform Step 1, we modify the method described by Gil et al. [9] and Khuller and Mitchell [10]. First, the sets P and Q are transformed into sets of lines L_P and L_Q in the dual plane, respectively. The sorted order of P around a point q can be obtained by considering the intersection order of L_P with the dual-line L_q using a method described in [13]. This step requires $O(n)$ time for each point in Q. The planar graph construction method in [5] can be implemented to find the line intersection order of L_P set with each line in L_q. We construct a graph G of the arrangement of lines induced by L_P incrementally by introducing one line at a time, and construct the doubly connected edge list of L_P, which requires $O(n^2)$ time and $O(n^2)$ space. Then we continue this process by temporarily inserting each line in L_q to G, and finding the order of intersections of lines in L_P with L_q by traversing the sequence of edges in G along L_q, which takes $O(n)$ time. Then, applying a method analogous to that described in [13], the angular sorted order of P around each point q can be obtained in $O(n)$ time. Step 1 requires $O(n^2)$ time and $O(n^2)$ space for preprocessing. In Step 2, the simplicial depth of each point $q \in Q$ relative to P can be found in $O(n)$ time using the angular order of points of P around q [9]. This takes $O(nk)$ time, giving a total time of $O(n^2 + nk)$.

Step 1 requires finding the order of intersections between L_P and each line in L_q. Finding the order of intersections between one line and a set of m lines can be achieved using one of various methods: (a) incremental planar graph construction [5] in $O(m^2)$ time and $O(m^2)$ space, (b) line sweeping [18] in $O(m^2 \log m)$ time [8], or (c) topological sweeping [7] in $O(m^2)$ time and $O(m)$ space. Despite its lower costs as a function of m, when applied to our problem, topological sweeping takes $O(n^2 + k^2)$ time and $O(n + k)$ space because it processes additional intersections in L_P and L_Q that are not needed for Step 1. The most efficient method for finding the ordered intersections between L_P and each L_q line is incremental planar graph construction, which takes $O(n^2 + nk)$ time and $O(n^2)$ space.

Lemma 2 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, Algorithm S.II computes $SD_P(q)$ for every $q \in Q$ in $O(n^2 + nk)$ total time and $O(n^2 + k)$ space.*

3.1.3 Algorithm S.III

When k is large relative to n, construct the arrangement L formed by lines between every pair of points in P. This arrangement partitions the plane into $\Theta(n^4)$ convex cells in which every point within a cell has equal simplicial depth. By modifying the $O(n^4)$-time simplicial median algorithm of Aloupis et al. [2], we can compute the depths of all cells in $O(n^4)$ time. Aloupis et al. consider the arrangement of line segments connecting every pair of points in P, which also has $O(n^4)$ intersections and $O(n^4)$ cells. This method computes the number of points on each side of each line segment of P in $O(n^3)$ time. Further, Aloupis et al. showed that starting from a known depth value on a line segment, by processing each intersection point in $O(1)$ time, the simplicial depth along the line segment can be computed in $O(n)$ time [2]. We adapt this depth-finding method along a line segment to find the simplicial depth of cells in our arrangement L as described below.

Each line l in L is partitioned into three by the two points p_1 and p_2 in P that determine l: the line segment between p_1 and p_2 (colour this segment blue) and two rays (colour the rays red) on l rooted at p_1 and p_2, respectively. In the arrangement determined by L, only the blue segments are boundaries of simplices. There-

fore, when crossing from one cell to an adjacent cell, the depth changes if the two cells share a blue line segment on their common boundary. Similarly, if the two cells share a red segment on their common boundary, then both cells have the same simplicial depth.

We compute the number of points on each side of each line in L in $O(n^3)$ time. Starting from a cell C_i with known simplicial depth, the algorithm traverses the arrangement, calculating the simplicial depth of each cell relative to the depth of an adjacent cell whose depth was already computed. To find the simplicial depth of a cell C_j that shares a blue edge with C_i, subtract the number of points in P on the side of C_i to the blue edge and add the number of points in P on the side C_j to the blue edge. The simplicial depth inside C_i includes simplicies (triangles) bounded by the blue line segment and points in P on the side of C_i. When crossing the blue edge to C_j, we exit (subtract) one set of triangles and enter (add) a new set of triangles based on the blue line segment and points of P on the C_j side of the blue edge. If depth on simplex boundaries is required, then the depth on the blue edge is calculated by adding depth in C_i to the number of points in the side of C_j; no query point lies on a simplex boundary when $P \cup Q$ is in general position.

All cells outside the convex hull of P have depth zero; we can initiate our algorithm at any of these cells. The algorithm proceeds to compute the depths of all cells by traversing the planar graph determined by L starting from an extreme cell (with depth zero) using the technique described above. The depth of each individual cell is computed in $O(1)$ time. Therefore, the traversal takes time and space proportional to the number of cells: $\Theta(n^4)$.

Finally, for each point q in Q we apply a point location algorithm to identify the cell in the arrangement determined by L that contains q. Kirkpatrick's point location algorithm can be implemented in a t-edge planar subdivision with $O(t)$ preprocessing time, $O(t)$ space, and $O(\log t)$ query time [11]. In our case, $t \in \Theta(n^4)$, corresponding to $\Theta(n^4)$ cells in the planar subdivision determined by L (the number of edges is also $\Theta(n^4)$). Therefore, Kirkpatrick's point location algorithm can be used to find the locations of each point in Q in $O(n^4)$ preprocessing time, $O(n^4)$ space and $O(k \log n)$ query time. The simplicial depths of all points in Q can be computed in $O(n^4 + k \log n)$ time and $O(n^4)$ space.

Lemma 3 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, Algorithm S.III computes $SD_P(q)$ for every $q \in Q$ in $O(n^4 + k \log n)$ total time and $O(n^4 + k)$ space.*

Lemmas 1–3 give:

Theorem 4 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, the simplicial depths of points in Q with respect to P can be computed in $O(\min\{kn \log n, n^2 + nk, n^4 + k \log n\})$ time.*

3.2 Computing a Batch of Tukey Depth Queries

In this section, we describe two methods for computing a batch of Tukey depth queries based on previous work related to computing Tukey depth [17] and Tukey depth contours [15].

3.2.1 Algorithm T.I

In $\mathbb{R}^2$, the Tukey depth of a point q relative to a set P of n points can be computed in $O(n \log n)$ time [17]. Similar to Algorithm S.I in Section 3.1, a straightforward method for computing the Tukey depths of k query point is to apply a Tukey depth algorithm iteratively for each point of Q. This process take $O(kn \log n)$ time and $O(n)$ space to store the sorted order of P around each point of Q.

Lemma 5 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, Algorithm T.I computes $TD_P(q)$ for every $q \in Q$ in $O(kn \log n)$ total time and $O(n + k)$ space.*

3.2.2 Algorithm T.II

Algorithm T.I is efficient when k is small relative to n, but more efficient approaches are possible for larger values of k. Algorithm T.II begins by computing the Tukey depth contours of P using the algorithm of Miller et al. in $O(n^2)$ time and space [15]. Miller et al. showed how to build a point location data structure on the contours in $O(n^2)$ time to support $O(\log n)$-time Tukey depth queries. Therefore, the Tukey depths of k points can be calculated in $O(n^2 + k \log n)$ time and $O(n^2)$ space.

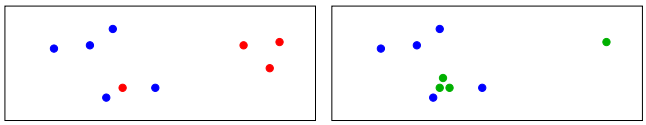
Lemma 6 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, Algorithm T.II computes $TD_P(q)$ for every $q \in Q$ in $O(n^2 + k \log n)$ total time and $O(n^2 + k)$ space.*

Lemmas 5 and 6 give:

Theorem 7 *Given a set P of n points and a set Q of k query points in general position in $\mathbb{R}^2$, the Tukey depths of points in Q with respect to P can be computed in $O(\min\{kn \log n, n^2 + k \log n\})$ time.*

4 Depth of a Set of Query Points

We introduce definitions for the simplicial depth and Tukey depth of a *set* Q of points relative to a set P of points. As we discuss below, our new definitions differ from previous definitions introduced by Barba et al. [3] and Pilz and Schnider [16].



A B

Figure 1: Relative to the blue set, the green and red sets have the same cardinal simplicial depth (Definition 2.2). However, by Definition 4.1, the simplicial depth of the green set is triple that of the red set. An analogous property holds for Tukey depth: the green and red sets have the same generalized Tukey depth (by Definition 2.1) relative to the blue set, but their Tukey depths differ (by Definition 4.2).

4.1 Simplicial Depth of a Set of Query Points

We define the simplicial depth of a set Q relative to a set P as the normalized sum of the number of points of Q contained in each simplex determined by points in P:

Definition 4.1 (Simplicial depth of a set of points) *Given a set P of n points and a set Q of k points in $\mathbb{R}^d$, the* simplicial depth *of Q relative to P is*

$$SD^*_P(Q) = \frac{1}{|Q|} \sum_{S \in \mathcal{S}} |Q \cap S|, \tag{5}$$

where $\mathcal{S}$ denotes the set of $\binom{n}{d+1}$ closed simplicies, each of which is the convex hull of $d+1$ points from P.

$SD^*_P(Q)$ can be expressed as the average simplicial depth of points in Q:

$$SD^*_P(Q) = \frac{1}{|Q|} \sum_{q \in Q} SD_P(q). \tag{6}$$

A derivation of (6) is given in Appendix A. (6) implies that $SD^*_P(Q)$ also has a natural probabilistic interpretation. If q is selected uniformly at random from Q, the expected value of the simplicial depth of q relative to P is $SD^*_P(Q)$.

Definition 4.1 differs from $CSD_P(Q)$ (Definition 2.2) introduced by Barba et al. [3]. $CSD_P(Q)$ counts the number of non-empty simplicies, which can result in similar depth values for significantly different point sets. Specifically, a small subset of points in the set Q can determine the depth of Q relative to P. See Figure 1. On the other hand, Definition 4.1 is a normalized sum of the number of points contained in each simplex. Equation (6) also suggests a family of measures that can be used to define the simplicial depth of a set Q with respect to a set P by substituting the average with another summary statistic of the distribution of the depths of points in Q. We discuss this briefly in Section 5.

We can compute $SD^*_P(Q)$ by computing the simplicial depth $SD_P(q)$ for each point $q \in Q$, and taking the average of these depth values. This can be achieved efficiently using the algorithms introduced in Section 3.1, which gives the following corollary.

Corollary 8 *Given a set P of n points and a set Q of k points in general position in the plane, $SD^*_P(Q)$ can be computed in $O(\min\{kn \log n, n^2 + nk, n^4 + k \log n\})$ time.*

As mentioned earlier, $CSD_P(Q)$ can be computed in $O(N^{7/3} \log^{O(1)} N)$ time, where $N = n + k$. By Corollary 8, the simplicial depth, $SD^*_P(Q)$, introduced in this paper can be computed asymptotically faster for any values of n and k.

Next, we consider another generalization of simplicial depth to sets, which we show is equivalent to Definion 4.1. For this, we introduce the *normalized simplicial depth* (NSD) of a query point q relative to P as

$$NSD_P(q) = \frac{1}{|\mathcal{S}|} \sum_{S \in \mathcal{S}} I(q \in S) = \frac{SD_P(q)}{|\mathcal{S}|}, \tag{7}$$

that is, it is the proportion of simplices from $\mathcal{S}$ that contain q. Interestingly, this normalized simplicial depth can also be interpreted as the probability that the query point q lies in a simplex whose vertices are selected at random from P or, equivalently,

$$NSD_P(q) = \mathbb{P}(q \in S), \tag{8}$$

where S is selected uniformly at random from $\mathcal{S}$. Liu [14] argued that this is an estimator of the probability that the query point q lies in a simplex formed from $d+1$ independent random points selected from a common distribution F in $\mathbb{R}^d$.

Now, consider generalizing the idea described above by selecting a simplex at random from $\mathcal{S}$, but by instead focusing on the expected number of points of Q that lie in that simplex. This depth measure, which we denote $ERS_P(Q)$ (Expected number of points of Q in a Random Simplex from P) is then

$$ERS_P(Q) = \mathbb{E}[Y_Q(S)], \tag{9}$$

where S is again randomly selected from $\mathcal{S}$, and where the random variable $Y_Q(S)$ denotes the number of points of Q that lie inside S. This is a reasonable measure of the depth of Q with respect to P, has an elegant and intuitive interpretation, and reduces to (8) when Q contains a single point. Indeed, when Q contains a single point, $\mathbb{E}[Y_Q(S)] = \mathbb{E}[I(q \in S)] = \mathbb{P}(q \in S)$, the normalized simplicial depth of q. We now justify that ERS_P and SD^*_P are equivalent measures of depth.

The number of points of Q that lie inside a simplex S constructed from points of P can be expressed as

$$Y_Q(S) = \sum_{q \in Q} I(q \in S), \tag{10}$$

and takes values in $\{0, 1, \ldots |Q|\}$. Also, the proportion of simplices in $\mathcal{S}$ that contain exactly y points of Q is

$$P_{\mathcal{S}}(y) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} I\big[Y_Q(s) = y\big], \tag{11}$$

for $y = 0, 1, \ldots, |Q|$. This also corresponds to the probability that the simplex constructed from three points selected at random from P contains exactly y points of Q.

In this context, the expectation of $Y_Q(S)$, which corresponds to the mean of the probability distribution in (11), can be shown to satisfy (see Appendix A)

$$SD^*_P(Q) = \frac{|\mathcal{S}|}{|Q|} ERS_P(Q). \tag{12}$$

From this, the simplicial depth of Q, as defined in Definition 4.1, is equivalent to $ERS_P(Q)$, the expected number of points in Q that lie a randomly selected simplex constructed from points of P, as the two depth measures are always proportional to each other.

We conclude this section by highlighting how CSD_P, defined in (4) as the number of simplices constructed from points of P that contain at least one point of Q, relates to the discussion above. Specifically, it is possible to write (see Appendix A)

$$CSD_P(Q) = |\mathcal{S}|\, \mathbb{P}(Y_Q(S) > 0). \tag{13}$$

This implies that, as a measure of depth, $CSD_P(Q)$ is equivalent to $\mathbb{P}(Y_Q(S) > 0)$, the probability that a random simplex contains at least one point of Q. In the case where Q contains a single point, this further reduces to $\mathbb{P}(Y_Q(S) > 0) = \mathbb{P}(q \in S)$ and justifies that $CSD_P(Q)$ is also a direct generalization of simplicial depth that applies to sets, but differs from $ERS_P(Q)$.

4.2 Tukey Depth of a Set of Query Points

We define the Tukey depth of a set Q relative to a set P as follows:

Definition 4.2 (Tukey depth of a set of points) *Given a set P of n points and a set Q of k points in $\mathbb{R}^d$, the* Tukey depth *of Q relative to P is*

$$TD^*_P(Q) = \frac{1}{|Q|} \sum_{q \in Q} TD_P(q). \tag{14}$$

As with (6), (14) corresponds to the average Tukey depth of points in Q relative to P, and carries the same probabilistic interpretation as for simplicial depth: (14) corresponds to the expected depth of a point selected uniformly at random from Q.

To compute $TD^*_P(Q)$, we can compute the Tukey depth of each point in Q relative to P using the algorithms introduced in Section 3.2, and take the average of those depth values. Therefore, we have the following corollary.

Corollary 9 *Given a set P of n points and a set Q of k points in general position in the plane, $TD^*_P(Q)$ can be computed in $O(\min\{kn \log n, n^2 + k \log n\})$ time.*

As mentioned in Section 2.3, Pilz and Schnider [16] introduced the generalized Tukey depth of a set Q relative to a set P, $GTD_P(Q)$. This definition can result in similar depth values for significantly different point sets. Specifically, a small subset of points in the set Q can determine the depth of Q relative to P. See Figure 1. Pilz and Schnider did not provide an algorithm to compute $GTD_P(Q)$, but based on Definition 2.1, a straightforward iterative approach for computing $GTD_P(Q)$ would require $O(n^3 + k^3)$ time. This time can likely be reduced to $O(n^2 + k^2)$ time by constructing the arrangement of lines determined by pairs of points in $Q \cup P$, and traversing the arrangement to examine all possible subsets of $Q \cup P$ contained in a half-plane; traversing from one cell in the arrangement to a neighbouring cell corresponds to adding or removing $O(1)$ points from $Q \cup P$.

5 Discussion and Directions for Future Research

In this paper, we introduced new definitions for the simplicial depth and Tukey depth of a set Q of points relative to a set P of points in $\mathbb{R}^d$, and we presented algorithms for computing these in $\mathbb{R}^2$.

This work suggests various possible generalizations of simplicial depth and Tukey depth to measure the depth of a query set Q. As the computation of these depth measures involves computing the depth of each point in Q, we could instead define a depth measure as a function of a different summary of the distributions of the simplicial depths and Tukey depths of individual points of Q relative to P. For instance, we could summarize the distribution of depths using a median, a minimum, a maximum, or a measure of spread, such as variance, range, skewness, or quantiles of this distribution. These different summaries of the constructed depth distributions over the points of Q can all be computed in the same time and space complexities as in Corollaries 8 and 9. One could also define the depth of a set using another depth for individual points altogether.

Future work is warranted to investigate the characterization of these depth measures of sets of points. SD^*_P and TD^*_P are invariant under affine transformations and vanish at infinity. TD^*_P is consistent across dimensions. Other properties such as convexity, stability, and robustness remain to be analyzed, requiring appropriate generalizations for the depth of a set of points. Finally, some questions remain unanswered with respect to the possibility of improving the running times of the algorithms presented in Theorems 4 and 7. In particular, can we show corresponding lower bounds on the worst-case running time expressed in terms of n and k?

References

[1] G. Aloupis, C. Cortés, F. Gómez, M. Soss, and G. Toussaint. Lower bounds for computing statistical depth. *Computational Statistics & Data Analysis*, 40(2):223–229, 2002.

[2] G. Aloupis, S. Langerman, M. Soss, and G. Toussaint. Algorithms for bivariate medians and a Fermat-Torricelli problem for lines. *Computational Geometry*, 26(1):69–79, 2003.

[3] L. Barba, S. Lochau, A. Pilz, and P. Schnider. Simplicial depth for multiple query points. In *Proc. European Workshop on Computational Geometry*, pages 29:1–29:7, 2019.

[4] D. Bertschinger, J. Passweg, and P. Schnider. Tukey depth histograms. *arXiv preprint arXiv:2103.08665*, 2021.

[5] B. Chazelle, L. J. Guibas, and D.-T. Lee. The power of geometric duality. *BIT Numerical Mathematics*, 25(1):76–90, 1985.

[6] S. Durocher, R. Fraser, A. Leblanc, J. Morrison, and M. Skala. On combinatorial depth measures. *International Journal of Computational Geometry & Applications*, 28(04):381–398, 2018.

[7] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. *Journal of Computer and System Sciences*, 38(1):165–194, 1989.

[8] H. Edelsbrunner and E. Welzl. Constructing belts in two-dimensional arrangements with applications. *SIAM Journal on Computing*, 15(1):271–284, 1986.

[9] J. Gil, W. Steiger, and A. Wigderson. Geometric medians. *Discrete Mathematics*, 108(1-3):37–51, 1992.

[10] S. Khuller and J. S. Mitchell. On a triangle counting problem. *Information Processing Letters*, 33(6):319–321, 1990.

[11] D. Kirkpatrick. Optimal search in planar subdivisions. *SIAM Journal on Computing*, 12(1):28–35, 1983.

[12] S. Langerman and W. Steiger. Optimization in arrangements. In *Proc. Symposium on Theoretical Aspects of Computer Science*, pages 50–61. Springer, 2003.

[13] D. T. Lee and Y.-T. Ching. The power of geometric duality revisited. *Information Processing Letters*, 21(3):117–122, 1985.

[14] R. Y. Liu. On a notion of data depth based on random simplices. *The Annals of Statistics*, pages 405–414, 1990.

[15] K. Miller, S. Ramaswami, P. Rousseeuw, J. A. Sellares, D. Souvaine, I. Streinu, and A. Struyf. Efficient computation of location depth contours by methods of computational geometry. *Statistics and Computing*, 13(2):153–162, 2003.

[16] A. Pilz and P. Schnider. Extending the centerpoint theorem to multiple points. *Leibniz International Proceedings in Informatics*, 123:53–1, 2018.

[17] P. J. Rousseeuw and I. Ruts. Bivariate location depth. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 45(4):516–526, 1996.

[18] M. I. Shamos and D. Hoey. Geometric intersection problems. In *Proc. Symposium on Foundations of Computer Science*, pages 208–215. IEEE, 1976.

[19] J. W. Tukey. Mathematics and the picturing of data. In *Proc. International Congress of Mathematicians*, volume 2, pages 523–531, 1975.

A Proofs

In this Appendix we include complete details of proofs and arguments omitted from the main text due to space constraints.

Derivation of Equation (6)**:** Starting with Definition 4.1, and noting that

$$|Q \cap S| = \sum_{q \in Q} I(q \in S),$$

we see that

$$\begin{aligned} SD^*_P(Q) &= \frac{1}{|Q|} \sum_{S \in \mathcal{S}} |Q \cap S| \\ &= \frac{1}{|Q|} \sum_{S \in \mathcal{S}} \sum_{q \in Q} I(q \in S) \\ &= \frac{1}{|Q|} \sum_{q \in Q} \sum_{S \in \mathcal{S}} I(q \in S) \\ &= \frac{1}{|Q|} \sum_{q \in Q} SD_P(q), \end{aligned}$$

as claimed.

Derivation of Equation (12)**:** To avoid confusion in what follows, we reserve S to denote a randomly selected simplex and use s otherwise. First, we note that

$$\begin{aligned} \mathbb{E}[Y_Q(S)] &= \sum_{y=0}^{|Q|} y\, P_{\mathcal{S}}(y) \\ &= \sum_{y=0}^{|Q|} \frac{y}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} I\big[Y_Q(s) = y\big] \\ &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \sum_{y=0}^{|Q|} y\, I\big[Y_Q(s) = y\big] \\ &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} Y_Q(s). \end{aligned} \tag{15}$$

Now, using (10), we can further simplify (15) to get

$$\begin{aligned} \mathbb{E}[Y_Q(S)] &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \sum_{q \in Q} I(q \in s) \\ &= \frac{1}{|\mathcal{S}|} \sum_{q \in Q} \sum_{s \in \mathcal{S}} I(q \in s) \\ &= \frac{1}{|\mathcal{S}|} \sum_{q \in Q} SD_P(q) \\ &= \frac{|Q|}{|\mathcal{S}|}\, SD^*_P(Q). \end{aligned} \tag{16}$$

Finally, (9) and (16) together imply that

$$ERS_P(Q) = \frac{|Q|}{|\mathcal{S}|} SD^*_P(Q),$$

which is equivalent to (12).

Derivation of Equation (13)**:** First, we write

$$CSD_P(Q) = \sum_{s \in \mathcal{S}} I\big[Y_Q(s) > 0\big].$$

Then, making use of (11), derivations similar to those provided above allow one to see that

$$\begin{aligned} CSD_P(Q) &= \sum_{s \in \mathcal{S}} \sum_{y=1}^{|Q|} I\big[Y_Q(s) = y\big] \\ &= \sum_{s \in \mathcal{S}} \sum_{y=0}^{|Q|} I\big[y > 0\big] I\big[Y_Q(s) = y\big] \\ &= \sum_{y=0}^{|Q|} I\big[y > 0\big] \sum_{s \in \mathcal{S}} I\big[Y_Q(s) = y\big] \\ &= |\mathcal{S}| \sum_{y=0}^{|Q|} I\big[y > 0\big] P_{\mathcal{S}}(y) \\ &= |\mathcal{S}|\, \mathbb{E}\big[I(Y_Q(S) > 0)\big] \\ &= |\mathcal{S}|\, \mathbb{P}(Y_Q(S) > 0), \end{aligned}$$

as claimed.