Summary

This work is the first to validate theoretically the suspicions of many researchers — that the “average” Voronoi diagram is combinatorially quite simple and can be constructed quickly. Specifically, assuming that dimension $d$ is fixed, and that $n$ input points are chosen independently from the uniform distribution on the unit $d$-ball, it is proved that

- the expected number of simplices of the dual of the Voronoi diagram is $\Theta(n)$ (exact constants are derived for the high-order term), and

- a relatively simple algorithm exists for constructing the Voronoi diagram in $O(n)$ time.

It is likely that the methods developed in the analysis will be applicable to other related quantities and other probability distributions.

1 Introduction

The Voronoi diagram is a natural and intuitively appealing structure. Named for the mathematician Voronoi [22], it has been reinvented by researchers in several fields; in particular, meteorologists associate the two-dimensional version with the name Thiessen [21], and physicists honor Wigner and Seitz [24] for the three-dimensional version. It has been used by geologists, foresters, agriculturalists, medical researchers, geographers, crystallographers, and astronomers. Within the domain of the mathematical sciences, it is applied to simulate differential equations by finite element methods, to interpolate surfaces in geometric modeling systems, and to solve geometric problems such as finding Euclidean minimum spanning trees and largest empty circles. (Avis & Bhattacharya [1] present an extensive list of references for applications.)

The Voronoi diagram of a set of points — called sites — is a partition of $\mathbb{R}^d$ that assigns a surrounding region of “nearby” points to each of the sites. Each region is the $d$-polytope containing the points lying nearer to the site in its interior than to any other site. Formally, the (nearest-site) Voronoi diagram of the set $X_n = \{z_1, z_2, \ldots, z_n\}$ of $n$ sites in $\mathbb{R}^d$ is the set of $n$ convex regions $V_i = \{x \mid \forall j : \text{dist}(x, z_i) \leq \text{dist}(x, z_j)\}$ for $1 \leq i \leq n$.

The straight-line dual of the Voronoi diagram in the plane is called the Delaunay triangulation. In the planar case, sites $z_i$ and $z_j$ are joined by an edge in the Delaunay triangulation if and only if $V_i$ and $V_j$ share an edge. In $d$ dimensions, sites $z_{i_0}, z_{i_1}, \ldots, z_{i_k}$ define a $k$-face of the Voronoi dual if and only if $V_{i_0}, V_{i_1}, \ldots, V_{i_k}$ share a $(d-k)$-face in the Voronoi diagram. The Voronoi diagram can be constructed easily from its dual and vice versa. If, as is assumed in the sequel, no $d+2$ sites fall on the same hypersphere, the following facts are easily verified.

- $(d + 1)$ sites are vertices of a $d$-simplex in the Voronoi dual if and only if the interior of the hyp-
persphere passing through these sites contains no other sites. (We call such a hypersphere empty or site-free.)

- \((k + 1)\) sites are vertices of a \(k\)-face of some Voronoi dual simplex if and only if some empty hypersphere passes through these sites.
- Every convex-hull face is a face of some Voronoi dual simplex.
- The Voronoi dual partitions the convex hull of \(X_n\) into \(d\)-simplices.

Many researchers have considered Voronoi-diagram construction. Previous work on the two-dimensional case has been surveyed elsewhere \([7]\). Browstow \textit{et al.} \([5]\), Finney \([10]\), and Tanemura \textit{et al.} \([20]\) have addressed the construction of three-dimensional diagrams; none of these authors present detailed analyses of running time.

Bowyer \([3]\) and Watson \([23]\) describe algorithms for higher dimensions, but neither analyzes his algorithm rigorously. Bowyer gives some theoretical and empirical evidence suggesting that his algorithm requires \(O(n^{d+1/d})\) time on average for points uniform in a \(d\)-dimensional hypercube. Watson claims \(O(n^{2-1/d})\) time in the worst case. Avis and Bhattacharya's algorithms for the Voronoi diagram and its dual \([1]\) rely heavily on the simplex method for linear programming; since this method has exponential worst-case running time, they focus mainly on experimental studies of their algorithms' performance and of the expected complexity of the diagrams for points distributed uniformly in the unit hypercube. Their results suggest combinatorial complexity proportional to input size for this distribution.

Brown \([4]\) was the first to observe a pleasing connection between Voronoi diagrams in \(d\) dimensions and convex hulls in \(d + 1\) dimensions that allows any convex-hull algorithm to be used to construct Voronoi diagrams. The gift-wrapping algorithm \([6,2,19]\) may be used in \(O(n^{d+1/d})\) time, where \(S_n\) is the number of dual simplices in the result. Or Seidel's shelling algorithm \([17]\) may be used in \(O(n^2 + S_n \log n)\) time.

Seidel \([16,18]\) has showed that \(S_n\) can be extremely large — \(\Theta(n^{(d+1)/2})\) — in the worst case. On the other hand, it is not difficult to construct families of problem instances for which \(S_n = \Theta(n)\). Thus probabilistic estimates of the average value of the two quantities are useful.

Meijering \([13]\) and Gilbert \([11]\) have considered the Voronoi diagram of sites from a Poisson process of fixed intensity in \(\mathbb{R}^4\); Meijering showed that the expected number of nearest-site Voronoi neighbors of a site depends only on \(d\); in particular, it is 6 for \(d = 2\) and \(\approx 15.54\) for \(d = 3\). Such a set of sites may be thought of as an infinite set of sites drawn from a uniform distribution over all of \(\mathbb{R}^d\). In computational practice, however, one must deal with finite sets of sites drawn from a particular distribution, e.g., a uniform distribution on the interior of some convex body like a hypercube or hypersphere. Two sites are neighbors if and only if they lie on the surface of some ball that contains no other site. In the Poisson case, a pair of distant neighbors is always unlikely since it implies the existence of a large empty ball. In the case of a bounded set of sites, it can still be shown that sites far from the boundary of the body probably have only nearby neighbors, but some distant pairs of neighbors always occur near the boundary of the body, where most of the empty ball may lie outside the support of the distribution. Thus results dealing only with the Poisson case are inadequate for the average-case analysis of algorithms.

The next section describes a new method for determining \(ES_n\), the expected number of simplices in the dual of the Voronoi diagram. In \(\S 3\) this method is applied to the analysis of the asymptotic behavior of \(ES_n\) for sites drawn independently from the uniform distribution in the unit \(d\)-ball. \(\S 4\) presents a variation of the gift-wrapping algorithm that uses standard bucketing techniques. Finally, in \(\S 5\), the methods of \(\S 2\) are applied to show that the algorithm requires only linear time on average for i.i.d. points in the unit \(d\)-ball.

### 2 A General Method for Bounding the Expected Complexity of Voronoi Diagrams

This section describes a general method for bounding \(ES_n\), the expected number of simplices in the duals of nearest- and furthest-site Voronoi diagrams of random point sets.

The first \(d + 1\) points \(z_1, \ldots, z_{d+1}\) define a \(d\)-simplex with probability one. Let us first reckon the probability \(P_n\) that they also define a simplex in the dual of the nearest-site Voronoi diagram. This is just the probability that the other \(n - d - 1\) points lie outside the hypersphere passing through the \(d + 1\) points. Writing \(g(\cdot)\) for the density function of the \(z_i\) and \(\Gamma\) for the probability content of interior of the hypersphere, we see that this probability is

\[
P_n = \int_{\mathbb{R}^d} \cdots \int_{\mathbb{R}^d} \prod_{i=1}^{n-d-1} g(z_i) \, dz_1 \cdots dz_{d+1}.
\]
and that the expected number of simplices is
\[ ES_n = \binom{n}{d+1} P_n. \]

We next carry out a transformation of coordinates. The \(d+1\) points \( z_1, z_2, \ldots, z_{d+1} \) can be expressed in terms of a \(d\)-vector \( p \) representing the center of the \(d\)-sphere they define, a scalar \( r \) representing the radius of that sphere, and \(d-1\) angles \( \psi_1, \psi_2, \ldots, \psi_{d-1} \) for each \( z_i \). The determinant of this transformation can be shown to be
\[
d! \prod_{1 \leq i \leq d+1} \sin(\psi_i) d^{d-2} \sin(\psi_{d+1}) d^{d-3} \sin(\psi_i) \sin(\psi_{d+1}),
\]
where \( \sin(\psi_i) \) represents the volume of the simplex formed by the \(d+1\) points. Then
\[
P_n = d! \int_0^\infty \int_0^\infty \int_0^\infty \cdots \int_0^\infty h(z_1) \cdots h(z_{d+1}) \sin(\psi_1) \cdots \sin(\psi_{d+1}) d\psi_1 \cdots d\psi_{d+1}.
\]
with \( h(z_i) = r^{d-1} \cos(\psi_{d-1})^{d-3} \cos(\psi_{d})^{d-3} \cdots \cos(\psi_{i})^{d-3} \).

If we define
\[
\bar{g}(r, p) = \int_0^\infty \cdots \int_0^\infty h(z_1) \cdots h(z_{d+1}) d\psi_1 \cdots d\psi_{d+1},
\]
which is loosely the probability that a random point falls on the sphere defined by \( r \) and \( p \), and we also define
\[
esimp(r, p) = E(\sin(\psi_1, \psi_2, \ldots, \psi_{d+1}) \mid \|z_i - p\| = r \text{ for } 1 \leq i \leq d+1),
\]
the expected volume of the simplex formed by \(d+1\) random points on that sphere, then we can show eventually that \(P_n\) is
\[
d! \int_0^\infty \int_0^\infty \int_0^\infty \cdots \int_0^\infty h(z_1) \cdots h(z_{d+1}) \sin(\psi_1) \cdots \sin(\psi_{d+1}) d\psi_1 \cdots d\psi_{d+1}.
\]
If \( g \) is spherically symmetric, then \( \bar{g}, \Gamma, \) and \( esimp \) depend only on \( r \) and \( \|p\| \). To exploit this, we express \( p \) in generalized spherical coordinates \((q, \theta_1, \theta_2, \ldots, \theta_{d-1})\). Eventually, we have
\[
esimp(r, q) \exp(-n \Gamma(r, q)) d\Gamma dq,
\]
with \( \mu_d = (2\pi^{d/2})/(d\Gamma(d/2)) \) being the volume of the unit \(d\)-ball.

3 Bounds for the Uniform Distribution in a \(d\)-Ball

In this section we turn to the uniform distribution in the unit \(d\)-ball in particular and prove the following theorem.

**Theorem 1** Let \( x_n = \{x_1, x_2, \ldots, x_n\} \) be a set of \( n \) sites drawn independently from the uniform distribution on the interior of the unit \(d\)-ball. Then \( ES_n \), the expected number of simplices of the dual of the Voronoi diagram of \( x_n \), is \( \Theta(n) \).

**Proof Sketch.** Let \( U \) denote the unit \(d\)-ball, \( B \) the ball defined by the points \( z_1 \) through \( z_{d+1} \), and \( \partial B \) the surface of \( B \). We have \( g(x) = 1/\mu_d \) when \( \|x\| \leq 1 \) and \( g(x) = 0 \) otherwise. Then
\[
\Gamma = \frac{\text{vol}(B \cap U)}{\mu_d},
\]
\[
\bar{g} = \frac{\text{vol}_{d-1}(\partial B \cap U)}{\mu_d},
\]
\[
esimp \leq \text{vol}(\text{conv}(\partial B \cap U)).
\]

We divide the domain of integration of the integral of (2.2) into eight regions corresponding to the possible patterns of intersection of the two balls \( B \) and \( U \), and compute \( \Gamma, \bar{g}, \) and \( esimp \) for each case. A relatively simple case is Case 1 (Figure 1); the most difficult is perhaps Case 6 (Figure 2).

Rather lengthy calculations show that the contribution of Case 1 dominates, and that
\[
ES_n \sim \frac{d! \mu_d \nu_d n^{d+1}}{(d+1)}
\]
where \( [14] \) \( \nu_d = \frac{\Gamma((d+1)/2)\Gamma(d/2)^{d+1}}{\sqrt{\pi d! \Gamma(d/2)\Gamma((d+1)/2)^d}}. \)

Applying (3.1) for \( d = 2 \), we obtain \( ES_n \sim 2n \). This is confirmed by well-known combinatorial results. We
also have

\[ E_{S_n} \sim \frac{24\pi^2}{35}n \approx 6.77n \text{ for } d = 3; \]

\[ E_{S_n} \sim \frac{286}{9}n \approx 31.78n \text{ for } d = 4. \]

These values are not obviously inconsistent with the values 6.31 and 25.6 found empirically by Avis & Bhattacharya for (rather small) samples of 1000 points chosen from the unit hypercube [1, Table 1]; it is reasonable to conjecture that (3.1) in fact holds for point sets chosen from a uniform distribution on any convex body.

4 A Fast Algorithm for the Unit d-Ball

It is immediate from Theorem 1 and the connection to convex hulls mentioned in the introduction that the Voronoi diagram of random points from a d-ball can be constructed in \(O(n^3)\) time on average by either the shelling algorithm or the gift-wrapping algorithm. In this section, we describe a faster algorithm requiring only \(O(n)\) time on average.

Our algorithm enumerates the \(d\)-simplices of the Voronoi dual. It is similar to Maus' planar algorithm [12]: it employs standard bucketing techniques, and its operation in \(\mathbb{R}^d\) corresponds to the operation of the gift-wrapping algorithm for convex hulls in \(\mathbb{R}^{d+1}\). It will be convenient to call the \(d\)-simplices of the Voronoi dual cells and the \((d-1)\)-simplices facets (since they are facets of the cells); likewise, we will call an empty \(d\)-sphere defined by the vertices of a cell a cell sphere and a \((d-1)\)-sphere defined by the vertices of a facet a facet sphere. The algorithm proceeds by repeatedly finding a new cell adjacent to a known facet. Except for facets that are also facets of the \((d\)-dimensional) convex hull, every facet belongs to exactly two cells. We maintain a dictionary of facets for which only one cell is known. At each step a facet is removed from the dictionary and its unknown cell (if it exists) is found by searching for the unknown \((d+1)\)st vertex (the site search). The remaining facets of the new cell are searched for in the dictionary (facet searches). Each that is found is deleted, since both of its cells are already known. Each that is not found is inserted so that its unknown cell will be searched for in some later step.

The facet dictionary is organized as a linear array of \(n\) buckets; a random facet falls into a particular bucket with probability \(1/n\). Within each bucket facets may be organized in a balanced search tree to insure good (logarithmic) worst-case performance, but a simple linear
function $\text{Find}_{\text{site}}(\mathcal{F}, \mathcal{H})$
- $\mathcal{U}$ is the unit $d$-ball.
- $\text{Site}_{\text{lvl}}(z, \mathcal{F})$
  - is the signed distance of the center of the $d$-ball
  defined by $z$ and $\mathcal{F}$ above the hyperplane of $\mathcal{F}$.
- $\text{Box}_{\text{lvl}}(B, \mathcal{F}) = \min_{y \in \mathcal{F}(y, \mathcal{H})} \text{Site}_{\text{lvl}}(y, \mathcal{F})$.
- $q$ is a priority queue of boxes ordered by $\text{Box}_{\text{lvl}}$.

$\text{Insert}(\text{Box}(\text{center of facet sphere of } \mathcal{F}), q);$
ans := nil;
while ($q \neq \emptyset$) 
  if ($\text{Box}_{\text{lvl}}(\text{Find}_{\text{min}}(q), \mathcal{F}) < \text{Site}_{\text{lvl}}(\text{ans}, \mathcal{F})$) do
    curBox := $\text{Delete}_{\text{min}}(q)$;
    for $z \in \text{curBox}$ do
      if ($z \in \mathcal{H}$) 
        then $\text{ans} := z$;
      for newBox adjoining $\text{curBox}$ do
        if ($\text{newBox} \cap \mathcal{H} \neq \emptyset$) 
          then $\text{Insert}(\text{newBox}, q)$;
  return $\text{ans}$;
end $\text{Find}_{\text{site}}$

Figure 3: The site-searching routine.

Pseudo-code for the site-searching function $\text{Find}_{\text{site}}$ is found in Figure 3. To speed the site searches, we partition the hypercube $[-1, 1]^d$ into approximately $2^d \mu_d$ hypercubic “boxes” of side $[(\mu_d/n)^{1/d}]$ and volume about $\mu_d/n$. Boxes intersecting the unit ball will contain in expectation at most one site each. On a particular call to $\text{Find}_{\text{site}}$, let $\mathcal{F}$ and $\mathcal{H}$ be the actual arguments (a known facet and a halfspace to search), let $\mathcal{U}$ be the unit $d$-ball, and let $B$ be the cell ball of the unknown cell if it exists. If the cell exists, it is necessary and sufficient to examine those boxes intersecting $(B \cap \mathcal{H} \cap \mathcal{U})$ to determine it. If it does not exist, it is necessary and sufficient to examine the boxes intersecting $(\mathcal{H} \cap \mathcal{U})$ to determine this (Figures 4, 5). The function $\text{Find}_{\text{site}}$ examines almost exactly those boxes.

$\text{Site}_{\text{ctr}}(z, \mathcal{F})$ is the center of the $d$-sphere determined by the point $z$ and the $(d-1)$ vertices of the facet $\mathcal{F}$. If $\mathcal{H}$ is the halfspace $(z, \mathcal{F}) > 0$, then the goal of the site search is to find the site $z \in \mathcal{H}$ for which $\text{Site}_{\text{lvl}}(z, \mathcal{F}) = (a, \text{Site}_{\text{ctr}}(z, \mathcal{F}))$ is minimized. The priority queue is primed by inserting the box containing the center of the facet sphere of $\mathcal{F}$. When a box is removed from the priority queue and its contents are examined, the $2d$ boxes that share a facet with it are entered into the queue for possible examination later. The boxes in the priority queue are ordered by the quantity

Figure 4: What $\text{Find}_{\text{site}}$ must search when successful.

Figure 5: What $\text{Find}_{\text{site}}$ must search when unsuccessful.
If I... ideally, we would like to have

\[ \text{Boz rq} \times (3) = \min_{y \in B} \text{Site}(y, 3) \]

(where the minimum is taken over all points, not sites, in \( B \)) but this quantity apparently requires \( \Omega(3^d) \) time to compute. Instead, we approximate this in \( O(d) \) time.

We define

\[ \text{Boz \_lu}(B, 3) = \min_{y \in S(B)} \text{Site}(y, 3), \]

where \( S(B) \) is the circumsphere of the hypercube \( B \).

This definition satisfies

\[ \text{Boz \_lu}(B, 3) \leq \min_{y \in (B \cap \mathcal{H})} \text{Site}(y, 3); \]

thus it guarantees that the correct site will be found, while perhaps causing a few boxes to be examined unnecessarily. The computation involves projecting the center of the box onto the hyperplane defined by \( \mathcal{F} \), solving a quadratic equation, then choosing one of its roots. (See Figure 6, where \( K \) is the center of the box \( B \), and \( \text{dist}(C, P) \) is the value of \( \text{Boz \_lu} \).

The priority queue operations \( \text{Insert}, \text{Find \_min}, \) and \( \text{Delete \_min} \) can be implemented so that only \( O(\log n) \) time is required for each, but a naive linked-list implementation in which each operation requires time proportional to the length of the list suffices for the purposes of our average-case analysis.

5 Analysis of the Algorithm

In this section we show that all the facet and sites searches made by Algorithm A in the previous section can be completed in \( O(n) \) time on average.

Lemma 2. The facet searches can be completed in \( O(n) \) expected time.

Proof Sketch. To manage the facet dictionary, we maintain \( n \) buckets and hash facets into buckets by computing the exclusive-or of the binary representations of the indices of the sites defining the facet. Within each bucket we maintain a linked list of the facets in the bucket; each search or insertion in the bucket takes time proportional to the length of this list. It is not hard to see that this scheme hashes an equal number of the \( d \)-subsets of \( \{1, 2, \ldots, n\} \) to each bucket, but we must show that those \( d \)-subsets actually defining facets are not correlated in a way that causes them to be hashed to a small number of buckets.

Let \( C(n, d) = \binom{n}{d} \), let \( \mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{C(n,d)} \) be the \( d \)-subsets of \( X_n \), and let \( F_i \) represent the condition "\( F_i \) is a dual facet". The conditional probability \( \Pr(\mathcal{F} | F_i) \) depends only on \( |\mathcal{D}_i \cap \mathcal{D}_j| \). If \( |\mathcal{D}_i \cap \mathcal{D}_j| = m > 0 \), then there are at most \( C(n-d, m-1)/C(m, m-1) = O(n^{m-1}) \) ways to choose the \( m \) elements of \( \mathcal{D}_i \) from \( \mathcal{D}_j \) so that \( \mathcal{D}_i \) and \( \mathcal{D}_j \) fall into the same bucket. This holds because choosing the first \( m-1 \) elements fixes the \( m \)th. On the other hand, there are \( C(n-d, m) = \Theta(n^m) \) ways to choose the \( m \) elements of \( \mathcal{D}_i \) overall. Thus only a fraction \( O(1/n) \) of the facets fall into the same bucket as \( \mathcal{D}_i \). Roughly, this shows that any correlation is at the level of constant factors only.

We now turn to the search for the \((d+1)\)st site completing a cell with a known facet. We call a site search "successful" if a site is found, and "unsuccessful" if no site is found because the facet lies on the boundary of the convex hull.

Lemma 3. The successful site searches can be completed in \( O(n) \) expected time.

Proof Sketch. If we define the distance between a point \( x \) and a set \( Y \) by

\[ \text{dist}(x, Y) = \min_{y \in Y} \text{dist}(x, y), \]

all boxes with circumspheres intersecting \( B \cap \mathcal{H} \cap U \) are contained by the set

\[ A = \{ x | \text{dist}(x, B \cap U) \leq \sqrt{d(\mu_d/n)^{1/d}} \}. \]

The use of a priority queue in \( \text{Find \_site} \) guarantees that no box is examined unless its circumsphere is contained
by A. Assuming the naive linked-list implementation, the cost of the each priority queue operation is at most proportional to the total number of boxes examined. The cost of examining the sites in a box is $O(1)$ in expectation. The expected total cost of the site search is therefore proportional to the square of the number of boxes examined, or

$$ecost = O((n \cdot \text{vol} A)^2). \tag{5.2}$$

If we write $C_n$ for the total cost of all successful site searches needed to compute the Voronoi diagram of $X_n$, we have

$$EC_n \leq \binom{n}{d+1} \int_{\mathbb{R}^d} \cdots \int_{\mathbb{R}^d} ecost(x_1, \ldots, x_{d+1})$$

$$(1 - \Gamma)^n_{d+1} g(x_1) \cdots g(x_{d+1}) dx_1 \cdots dx_{d+1};$$

the $(d+1)$-fold integral represents an upper bound on the expected cost of a successful site search to complete the cell $x_1 \ldots x_{d+1}$. Proceeding as in §2, we eventually obtain

$$EC_n = O(1) \cdot \int_0^\infty \int_0^\infty q^{d-1} r^{-d} g^{d+1}(r, q)$$

$$\text{esimp}(q, r) \exp(-n \Gamma(q, r)) ecost(q, r) \, dr \, dq.$$

We continue as in §2, dividing the domain of integration into eight regions; $ecost$ is calculated differently in each region according to the geometry of the situation. □

Finally, we consider the unsuccessful site searches for facets of the convex hull.

Lemma 4 The unsuccessful site searches can be completed in $o(n)$ expected time.

Proof Sketch. In the same way that the method of §2 for determining the expected number of dual simplices was modified to derive the cost of the successful searches, published methods for finding the expected number of convex-hull facets [9,15,8] can be modified to estimate the expected cost of the unsuccessful searches. Details are left for a more complete exposition, but it can be shown that $EU_n = O(n^{1-1/(d^2+d)}) = o(n)$. □

Lemmas 2, 3, and 4 together imply the following theorem.

Theorem 5 Let $X_n = \{X_1, X_2, \ldots, X_n\}$ be a set of $n$ sites drawn independently from the uniform distribution on the interior of the unit $d$-ball. Then for fixed $d$, Algorithm A constructs the Voronoi diagram of $X_n$ in $O(n)$ time on average.

6 Discussion

The algorithm presented is clearly optimal in the average-case sense when dimension is fixed, and is asymptotically faster than any other known. If a balanced-tree implementation of priority queues is used, the running time of this algorithm is $O(S_n n \log n)$, only a factor of $\Theta(\log n)$ worse than the standard gift-wrapping algorithm. Worst-case performance can be improved to $O(n S_n)$ if the use of buckets is abandoned on any site search that examines $\sqrt{n}$ buckets. It should not be difficult to show that this occurs so infrequently that average performance is not affected.

It is easy to show that linear performance is preserved if the distribution is “quasi-uniform” in the unit $d$-ball, i.e., if its density bounded above and below by a positive constant everywhere in the $d$-ball. It is an open question whether the same approach yields an $O(n)$ algorithm — or even linear bounds on $S_n$ — for other distributions.

References

[1] D. Avis and B. K. Bhattacharya. Algorithms for computing $d$-dimensional Voronoi diagrams and their duals. In F. P. Preparata, editor, Advances in Computing Research: Computational Geometry, pages 159–180, Greenwich, Conn., 1983. JAI Press Inc.

[2] B. Bhattacharya. Worst-case analysis of a convex hull algorithm. Simon Fraser U., 1982.

[3] A. Bowyer. Computing Dirichlet tessellations. Computer J., 24(2):162–166, 1981.

[4] K. Q. Brown. Geometric Transforms for Fast Geometric Algorithms. PhD thesis, Carnegie-Mellon U., 1979.

[5] W. Browstow, J.-P. Dussault, and B. L. Fox. Construction of Voronoi polyhedra. J. Computational Physics, 29:81–97, 1978.

[6] D. R. Chaud and S. S. Kapur. An algorithm for convex polytopes. Journal of the ACM, 17(1):78–86, January 1970.
[7] R. A. Dwyer. A faster divide-and-conquer algorithm for constructing Delaunay triangulations. *Algorithmica*, 2(2):137–151, 1987.

[8] R. A. Dwyer. On the convex hull of random points in a polytope. *J. Appl. Prob.*, 26(4):888–899, 1989.

[9] B. Efron. The convex hull of a random set of points. *Biometrika*, 52:331–342, 1965.

[10] J. L. Finney. A procedure for the construction of Voronoi polyhedra. *J. Computational Physics*, 32:137–143, 1979.

[11] F. N. Gilbert. Random subdivisions of space into crystals. *Ann. Math. Stat.*, 33:958–972, 1962.

[12] A. Maus. Delaunay triangulation and the convex hull of n points in expected linear time. *BIT*, 24:151–163, 1984.

[13] J. L. Meijering. Interface area, edge length, and number of vertices in crystal aggregates with random nucleation. *Philips Res. Rep.*, 8:270–290, 1963.

[14] R. E. Miles. Isotropic random simplices. *Adv. Appl. Prob.*, 3:353–382, 1971.

[15] H. Raynaud. Sur l’enveloppe convexe des nuages des points aléatoires dans $\mathbb{R}^n$, I. *J. Appl. Prob.*, 7:35–48, 1970.

[16] R. Seidel. The complexity of Voronoi diagrams in higher dimensions. In *Proc. 20th Annual Allerton Conference on Communication, Control, and Computing*, pages 94–95. University of Illinois at Urbana-Champaign, October 1982.

[17] R. Seidel. Constructing higher-dimensional convex hulls at logarithmic cost per face. In *Proc. 18th ACM Symp. on Theory of Computing*, pages 404–413. ACM, May 1986.

[18] R. Seidel. On the number of faces in higher-dimensional Voronoi diagrams. In *Proc. 3rd Ann. Symp. on Computational Geometry*, pages 181–185. ACM, June 1987.

[19] G. Swart. Finding the convex hull facet by facet. *J. Algorithms*, 6:17–48, 1985.

[20] M. Tanemura, T. Ogawa, and N. Ogita. A new algorithm for three-dimensional Voronoi tessellation. *J. Computational Physics*, 51(2):191–207, August 1983.

[21] A. H. Thiessen. Precipitation averages for large areas. *Monthly Weather Review*, 39:1082–1084, July 1911.

[22] G. Voronoï. Nouvelles applications des paramètres continus à la théorie des formes quadratique, deuxième mémoire: recherches sur les paralléloèdres primitifs. *J. reine u. angew. Math.*, 134(1):198–287, May 1908.

[23] D. F. Watson. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. *Computer J.*, 24(2):167–172, 1981.

[24] E. Wigner and F. Seitz. On the constitution of metallic sodium. *Physical Reviews*, 43:804–810, 1933.