AN OPTIMAL CONVEX HULL ALGORITHM FOR POINTS SETS IN ANY FIXED DIMENSION

Bernard Chazelle

CS-TR-336-91

June 1991

An Optimal Convex Hull Algorithm for Points Sets in Any Fixed Dimension

BERNARD CHAZELLE
Department of Computer Science
Princeton University
Princeton, NJ 08544

Abstract: We present a deterministic algorithm for computing the convex hull of n points in E^d in optimal $O(n^{\lfloor d/2 \rfloor})$ time, for $d > 3$. This result settles an open question of long standing: optimal solutions were previously known only in two and three dimensions or alternatively by allowing randomization. The algorithm involves a fairly elaborate dynamic search process, whose fine points are clarified by using an analogy with statistical thermodynamics: this allows us to uncover some unexpected phenomena relating to the convergence of the algorithm. By duality, the algorithm can be used to construct the full lattice structure of the feasible set of n linear constraints in optimal $O(n \log n + n^{\lfloor d/2 \rfloor})$ time, for any fixed d. As an immediate corollary, we obtain an algorithm for computing the Voronoi diagram of n points in d-space in optimal $O(n \log n + n^{\lceil d/2 \rceil})$ time, which is also a new result.

The author wishes to acknowledge the National Science Foundation for supporting this research in part under Grant CCR–9002352.

1. Introduction

This paper provides a deterministic algorithm for computing the convex hull of n points in d-space in optimal $O(n^{\lfloor d/2 \rfloor})$ time, for any $d > 3$. This result settles an open question of long standing. Since optimal algorithms are known for the two and three-dimensional cases [18], the complexity of the problem is now completely elucidated when the dimension is fixed. By duality, the algorithm can be used to construct the full lattice structure of the feasible set of n linear constraints in optimal $O(n \log n + n^{\lfloor d/2 \rfloor})$ time. As a corollary, we obtain an algorithm for computing the Voronoi diagram of n points in optimal $O(n \log n + n^{\lceil d/2 \rceil})$ time, which is also a new result.

The convex hull problem has had an intriguing history. The cases $d = 2, 3$ have long been solved, but the problem has been elusive in higher dimensions. Ten years ago, Seidel gave an optimal algorithm for the case where d is even [20] (incidentally, the parity condition should be a hint of how "strangely" convex hulls can behave). Later, Seidel showed how to compute a convex hull incrementally by reporting each face at logarithmic cost (modulo quadratic overhead) [21], thus providing a quasi-optimal output-sensitive algorithm. (Note that the issue of output-sensitivity is not addressed by our result.) On the probabilistic front, Clarkson and Shor [6] gave an incremental randomized method with optimal expected time. An elegant simplification of that algorithm, and especially of its analysis, was proposed recently by Seidel [22].

But what about the deterministic case? Derandomizing randomized incremental algorithms is still, by and large, unconquered territory. Interestingly, this comes in sharp constrast with the largely successful derandomization of geometric algorithms based on random sampling [1,3,4,12,13,14]. The reader familiar with the recent work on hyperplane cuttings [3,14] or shallow cuttings [15] might wonder why such powerful divide-and-conquer tools cannot be brought to bear on the convex hull problem. Without getting too deep into what are fairly technical issues, let us only mention one major obstacle, which in our opinion, also happens to be the central difficulty in the convex hull problem; one might call it the "curse of odd dimensionality:" By the upper bound theorem, the facial complexity of the convex hull of n points is $O(n^{\lfloor d/2 \rfloor})$. Therefore, if the points are in general position, the total complexity of the faces incident upon a given vertex is on average a fraction $O(1/n)$ of the total, which gives an upper bound of $O(n^{\lfloor d/2 \rfloor - 1})$. This simple fact is the key to the efficiency of the randomized convex hull algorithms of Clarkson/Shor and Seidel. But what about the worst-case incidence complexity of a given vertex? It is the same as the maximum complexity of a convex hull of $n-1$ points in $(d-1)$-space, which in odd dimension, is still $O(n^{\lfloor d/2 \rfloor})$. Dually, this means that in odd dimension a single facet of an n-facet polyhedron might contribute a fixed fraction of the overall facial complexity of the polyhedron. By contrast, in an arrangement of n hyperplanes, no hyperplane can contribute more than a fraction $O(1/n)$ of the overall complexity. This difference between polyhedra and arrangements rules out any naive extension of our recent cutting construction [3] to the case of shallow cuttings [15] (i.e., cuttings whose action is restricted to the vicinity of a single cell of an arrangement).

We turn our sights in a different direction. We begin with a simple observation: By making randomized incremental methods into dynamic versions of random sampling-based algorithms, we can apply Raghavan and Spencer's method of *conditional probabilities* [19,23], and derive a deterministic convex hull algorithm. The catch is that the algorithm, although polynomial, is hopelessly slow (much slower than any naive convex hull algorithm). To speed it up, we need to introduce all kinds of data structuring tricks. But most important, we substitute *estimations* for exact computations. Specifically, we lump together sequences of computations whenever we can approximate their outputs cheaply with some reasonable precision. It is important to note that the errors which we introduce into the computation, although possibly very large, cannot affect the correctness of the final output, but only the time it takes to produce it. Why is that so? The Raghavan-Spencer method is used only for *sequencing* the input, i.e., for determining the order in which the points should be processed. Errors can only hurt insofar as they might lead to a less effective sequencing which slows down the algorithm. The probabilistic algorithms of Clarkson/Shor and Seidel guarantee the existence of a sequencing for which the running time of the algorithm is $O(n^{\lfloor d/2 \rfloor})$. Whereas randomization gives us good sequencing for free, we must compute it in the deterministic case, and that can be very costly. Our idea is to simulate the computation of a good sequencing but allow errors in order to speed up the process. Of course, this degrades the quality of the sequencing, but we can calibrate the error tolerance so that the resulting sequencing, although perhaps far from optimal, still leads to an efficient convex hull algorithm. To control the damage caused by the errors we use the theory of range spaces of finite VC-dimension, whose power, we should note, rests heavily on the fact that the dimension of the ambient space is bounded.

After this overview of the philosophy behind the algorithm, let us get down to specifics. We begin with the method of conditional probabilities. We must introduce a slight twist in its implementation, which might be interesting in its own right. Let us briefly review the method in its standard form. Given a (uniformly distributed) random sample of size r among n objects, let $A_1, \ldots, A_m$ be a set of "good" events, which might be true or false depending on the sample. If $\sum_i \text{Prob}[\bar{A}_i] < 1$, then obviously the probability that all events are good for some sample is nonzero. Now, commit one object p_1 into the sample and pick the other $r-1$ randomly. Given that first pick p_1, the sum of conditional probabilities $\pi(p_1) = \sum_i \text{Prob}[\bar{A}_i \,|\, p_1]$ might or might not be less than 1. But by definition the average value of $\pi(p_i)$ over all n objects is exactly $\sum_i \text{Prob}[\bar{A}_i] < 1$. Therefore, for at least one of the p_i's, we have $\pi(p_i) < 1$, and given that pick there is hope of completing the sample successfully. The standard derandomization strategy is to pick the p_{i_1} that minimizes $\pi(p_i)$ and iterate in this fashion. After r picks there's no randomness left, so $\sum_i \text{Prob}[\bar{A}_i \,|\, p_{i_1}, \ldots, p_{i_r}] < 1$ means that the sample $\{p_{i_1}, \ldots, p_{i_r}\}$ makes all good events true. Of course, we don't need the absolute minimum, but any p_j such that $\pi(p_j) \leq \sum_i \text{Prob}[\bar{A}_i]$ will do. Suppose we can compute any partial sum $\sum_i \pi(p_i)$ at the same cost as a single $\pi(p_i)$. Then, it is advantageous to home in on a good p_{i_1} by binary search in the following manner. Divide up the set of objects into two groups of roughly equal size, S_1 and S_2. Clearly, we can eliminate the group that maximizes the quantity $\sum_{p_i \in S_j} \pi(p_i)/|S_j|$, for $j = 1, 2$. Iterating on this process allows us to zero in on a good pick in roughly $\log n$ stages.

We will use this variant of the Raghavan-Spencer method in our convex hull algorithm. What makes our work difficult is that we make errors every step of the way, and we must be careful to bound the drift away from the average. The algorithm involves a complex dynamic search process, which it is illuminating to interpret in the language of statistical thermodynamics. This allows us to answer the question: How random-looking is the partial sample $p_1, \ldots, p_k$ computed at step k? We show in that in some cases the partial sample might deviate from a random one in some remarkable ways, for example, by being much more "favorable" than a random sample could ever be.

A recurring theme of this work is to design a slow, but extremely "fault-tolerant" algorithm, and then speed it up by trading exact computations for approximations and estimations. We expect that other applications of this design paradigm will be found. The theory of range spaces of finite VC-dimension is used for two purposes: one is to set the grounds for divide-and-conquer, and the other is to build efficient estimators. In particular, we use the fact that ε-approximations can be used to efficiently estimate how many vertices in a hyperplane arrangement lie inside a given simplex. We recall a useful result established in [3]. Let H be a set of n hyperplanes in E^d. Given $r > 0$, we say that $R \subseteq H$ is a $(1/r)$-*approximation* for H if, for any relatively open line segment e, the densities in R and H of the hyperplanes crossing e differ by less than $1/r$, i.e.,

$$\left|\frac{A}{|R|} - \frac{B}{|H|}\right| < \frac{1}{r},$$

where A (resp. B) is the number of hyperplanes of R (resp. H) meeting e. The subset R is called a $(1/r)$-*net* if any time the segment e intersects more than n/r hyperplanes of H, then at least one of them belongs to R. (The original definitions of these notions are abstract and nongeometric [10,24] but for our purposes it is more convenient to specialize them to the case of line segments.) By a result of Matoušek [12] it is possible to compute a $(1/r)$-approximation of size $O(r^2 \log r)$ in time $O(n(r^2 \log r)^{2d})$. A $(1/r)$-net of size $O(r \log r)$ can be obtained within the same time bound.

Lemma 1.1. [3] *Let H be a set of n hyperplanes in E^d and suppose that we have available a $(1/r)$-approximation for H of size $O(r^2 \log r)$. Then, given any relatively open simplex s of dimension j, the number of vertices inside s that belong to the j-dimensional arrangement of H defined within the affine space spanned by s can be estimated in time $r^{O(1)}$ with an absolute error of at most $2jn^j/r$.*

2. The Convex Hull Algorithm

As is well known, computing the convex hull of n points is equivalent, by duality, to computing the intersection of n halfspaces [8]. We will work on the intersection problem for convenience. Let H be a set of n hyperplanes (assumed to be in general position for simplicity), and for each $h \in H$, let h^+ denote one of the closed halfspaces bounded by h. Given $R \subseteq H$, we use the notation R^+ to refer to the polyhedron $\bigcap_{h \in R} h^+$. Our goal is to compute an explicit description of the facial structure of H^+. We can check in linear time whether H^+ is empty, and if not, find a point O inside [3,5,7,16]. Obviously, we can assume that H^+ is not empty and that O does not lie on any hyperplane of H.

Because optimal algorithms already exist in two and three dimensions we can assume that $d > 3$. Our basic aim is to derandomize the probabilistic incremental algorithms of [6,22]. These algorithms insert each hyperplane into the current halfspace intersection (in dual form in the case of [22]), one at a time, in random order. In the worst case, the m-th hyperplane to be inserted may cut the current polyhedron in a cross-section of size $O(m^{\lfloor (d-1)/2 \rfloor})$. If this happens too often and d is odd, then the resulting algorithm may fall short of optimal. If we randomize the order of the input, however, the expected size of the cross-section should not exceed the average facet size (over all facets) of an m-facet polyhedron, which is $O(m^{\lfloor d/2 \rfloor -1})$. Summing up those sizes over all insertions gives the optimal bound of $O(n^{\lfloor d/2 \rfloor})$. This is, in essence, why the randomized version of the naive insertion algorithm works.

To derandomize it is tricky because random permutations are hard to produce deterministically and thus finding a good sequencing of the input might not be so easy. To begin with, how do we define a good permutation of the input, anyway? Given an ordering of the hyperplanes, let H_i be the set formed by the first i of them. Using the pseudo-metric induced by O and the arrangement $\mathcal{A}(H)$ formed by H, we define the notion of a "distance" from H_i^+ to O. The polyhedron that we seek, $H^+ = H_n^+$, is at distance 0, and in general, the distance of the polyhedron H_i^* to O reflects how far its vertices are from O in the pseudo-metric of $\mathcal{A}(H)$. If the sequencing is truly random then the (expected) distance of H_i^+ to O, for $i = 1, 2$, etc., decreases as a rate fast enough to keep the costs of computing the H_i^*'s within control. (The relation between the cost of computing the polyhedron H_i^*, given H_i, and its distance to O will become clear later.) A good sequencing is one that keeps the rate comparable to (or better than) the random case. Simple-minded methods such as the greedy one (insert the next hyperplane that maximizes the rate of decrease) are too local in nature to hold much promise. On the other hand, to explore the search space exhaustively is hopelessly expensive. The method of conditional probabilities offers a compromise between these two approaches by injecting a small amount of consideration for the future into the selection criterion. The idea is to keep track at any time of how many ways we have of completing H_i into H and produce a good sequencing. We average out a certain measure of "goodness" over all possible completions, and call the resulting quantity the *energy* of the system. Intuitively, the greater the energy the more likely a random completion will give a good sequencing. Our strategy is thus very natural: we pick as the next hyperplane the one that maximizes the energy, or at least any one that does not decrease it. (The latter corresponds to our weakened variant of the method of conditional probabilities.) The energy can be regarded as a means to "summarize" the future in a concise manner. Unfortunately, to evaluate it requires computing, among other things, the full arrangement $\mathcal{A}(H)$. This is where the theory of range spaces of finite VC-dimension comes into play by giving us the tools to approximate the energy both cheaply and accurately.

There are two additional difficulties which we must mention here. The first one is that the estimation tools we build can work only within a given range of input sizes. Therefore, they must be recomputed every time the size of H_i exceeds a certain range. To deal with this problem in a clean fashion, we break up the construction of the sequencing into a logarithmic number of separate phases, each of which can be carried out without having to rebuild the estimators.

The second difficulty is more serious. Suppose that H_i has been computed and we want to find the next hyperplane to be inserted so as to form H_{i+1}. Let $f(n)$ be the time to check how good a candidate for insertion a given hyperplane is. Clearly, the whole algorithm can run in no better than $O(n^2 f(n))$ time. So, in order to achieve optimal complexity in dimensions 4 and 5, it is imperative that the (amortized) value of $f(n)$ be constant. Intuitively, the candidacy of a new hyperplane depends on its "contribution" to the arrangement $\mathcal{A}(H)$, which is determined by $O(n^{d-1})$ facial features. Thus, to evaluate the candidacy of a new hyperplane in constant time is not so easy and requires special data structuring tricks. There are other snags in the way. Without being too technical, let us mention one of them which, although it surfaces only at the very end of our description of the algorithm, requires us to prepare for it very early on.

Recall that by analogy with the randomized algorithms we aim to insert hyperplanes one at a time, making sure that the contribution of any new hyperplane h to the current intersection H_i^+ is not too big. In this way, we can insert h by tracing explicitly its intersection with H_i^+. Actually, that any work related to the insertion of h should occur only within its "sphere of influence" is as crucial to our algorithm as it is to the Clarkson/Shor and Seidel methods. But in our case to abide by that principle is difficult for the following, rather frustrating, reason. Recall that the algorithm is driven by a certain energy function, which we must update after each insertion. At stage i, this function is of the form $\sum_v f(i,v)$, where the summation extends over all vertices v of $\mathcal{A}(H)$. The parameter $f(i,v)$ depends both on the "geometry" of v within $\mathcal{A}(H)$ and H_i^+ (e.g., its distance to O, how many hyperplanes of H_i pass through it), but also on the time coordinate i. After an insertion, as expected, $f(i,v)$ needs to be updated if v lies in the vicinity of the new hyperplane. However, because $f(i,v)$ also depends on i in a rather complicated manner, technically it must be updated for *all* vertices v. Since we cannot afford to do that we must somehow find shortcuts. We explore this problem further in Section 3. Borrowing elementary ideas from statistical thermodynamics we interpret the insertion of the various hyperplanes as a cooling process. The increment from i to $i+1$ corresponds to a drop in temperature which, of course, affects every particle (vertex) in the system. For an algorithmic standpoint, to deal with the difficulty of global changes we must use persistent data structures.

2.1. Getting Started.† We need to introduce a little notation. Let s be a simplex (closed or not) of any dimension. Given $X \subseteq H$, let $X(s)$ denote the set of hyperplanes in X that meet the relative interior of s without enclosing it. Most often, X is H itself and we concern ourselves with the size of $H(s)$. For this reason, we introduce the shorthand $V_s = |H(s)|$. Another important quantity, denoted H_s, is the number of vertices of $\mathcal{A}(H)$ that lie within the relative interior of s.

Given $R \subseteq H$, a standard triangulation, called *canonical*, of the arrangement $\mathcal{A}(R)$ is obtained by first triangulating recursively the $(d-1)$-dimensional cross-section of the arrangement made by each hyperplane, and then for each cell of the arrangement, lifting all the k-simplices on its boundary $(k = 0, \dots, d-1)$ toward a chosen vertex (except for the simplices decomposing the faces incident upon the vertex in question). When it comes to triangulating the polyhedron R^+, however, we must be more specific about the choice of lifting vertices. We define the *geode* of R as the particular triangulation of R^+ obtained in the following recursive manner. For $k = 2, 3, \dots, d$, in that order,

† To preserve the flow of the presentation, the proofs of all the lemmas have been moved to an Appendix. Also, take notice that all logarithms are to the base 2. Finally, to facilitate the calculations, we shall use $b_0, c_0, b_1, c_1, \dots$ as an unbounded supply of constants. Typically, the b_i's will be local (and reusable) while the c_i's will be global: all will be assumed large enough to satisfy all the inequalities in which they appear (warning: this assumption will not be restated). Dependencies among these constants will exist and will be mentioned systematically.

triangulate each k-face f of H^+ like this: If $k < d$, let $v_1, \ldots, v_m$ be the vertices of f and let v_1 be the one that minimizes the quantity V_e, where $e = Ov_i$ $(1 \le i \le m)$. Lift toward v_1 the triangulation of each j-face $(j < k)$ of R^+ that lies within f but is not incident upon v_1. For $k = d$, simply lift toward O the triangulation of ∂R^+ just obtained. An easy inductive argument based on the Upper Bound Theorem shows that the size of the geode is $O\big(|R|^{\lfloor d/2 \rfloor}\big)$. Our next result motivates our particular choice of triangulation.

Lemma 2.1. *Given the geode of $R \subseteq H$, for any constant c large enough, $\sum_s V_s^c \le c^c \sum_e V_e^c$, where the first sum is taken over all d-dimensional simplices s of the geode and the second one over all segments e connecting O to the vertices of the geode.*

Our strategy for computing H^+ is to take a nested sequence of "pseudo-random" samples

$$R_0 \subset R_1 \subset R_2 \subset \cdots \subset H$$

and compute the geode of each R_i in that order, using the geode of R_{i-1} to help us in that endeavor. Each R_i^+ provides a better approximation of H^+ than the previous one and, more important, its geode helps to "localize" the computation into pockets where we can concentrate our efforts. We should point out that although R_i contains R_{i-1}, the geode of R_i^+ is not a refinement of that of R_{i-1}^+. In particular, we do not use geodes to break down the problem into subproblems to be solved recursively. We always keep all the pieces of the puzzle together: a proper analogy might be to say that we begin with a fuzzy picture of H^+, and that the entire computation is aimed at increasing the resolution of that picture until all blurness is gone. As the resolution increases it is important to "localize" the computation so as to limit the number of hyperplanes which need to be compared against each other. We choose the size r_i of each sample R_i so that $r_0, r_1, \ldots$ follow a geometric progression. Thus, we need roughly $\log n$ phases to complete the computation.

For a sample R_i to be good, the vertices of its geode should be as "close" to O as possible in the pseudo-metric defined by the arrangement $\mathcal{A}(H)$; this makes sense since, after all, the vertices of H^+ are those at distance zero from O in that metric. To capture this notion of proximity in a single number, we introduce a penalty function, which leads to the notion of a semicutting. We say that $R \subseteq H$ forms a *semicutting* (for H) if

$$\sum_e V_e^{c_1} \le c_0 r^{\lfloor d/2 \rfloor} \left(\frac{n}{r}\right)^{c_1},$$

where $r = |R|$ and the sum ranges over all segments e connecting O to the vertices of R^+. Looking at the distribution of hyperplanes crossing the edges e, it appears that a semicutting expresses an upper bound on the c_1-th moment of that distribution. Expectedly, this implies a bound on all lower moments. Indeed, for any $c \le c_1$,

$$\sum_e V_e^c \le 2c_0 r^{\lfloor d/2 \rfloor} \left(\frac{n}{r}\right)^c.$$

We shall use this property throughout our discussion without mentioning it again. To see why it holds, let $t_e = rV_e/n$, and observe that

$$\sum_e t_e^c \le c_0' r^{\lfloor d/2 \rfloor} + \sum_e t_e^{c_1} \le (c_0 + c_0') r^{\lfloor d/2 \rfloor},$$

where $c_0' r^{\lfloor d/2 \rfloor}$ is an upper bound on the number of vertices in R^+.

Our goal now is to build a semicutting R_i and its geode, for each $i = 0, 1, 2$, etc. When $r_i = |R_i|$ gets to be big enough so as to differ from n by only a constant factor, there will be so few hyperplanes crossing each simplex of the geode on average that we will be able to complete the computation of H^+ directly by using any naive method. To obtain R_0, we could start with any subset of H and adjust c_0 accordingly. But for technical reasons (actually, just in order to simplify further calculations) we wish to ensure that

$$\sum_e V_e^{c_1} \le \left(\frac{n}{c_1}\right)^{c_1}. \tag{2.1}$$

We choose R_0 as a $(1/\rho_0)$-net for H. The net R_0 has size $r_0 = O(\rho_0 \log \rho_0)$ and can be computed in $O(n)$ time, provided that ρ_0 is a constant [12]. By definition,

$$\sum_e V_e^{c_1} \le b_0 r_0^{\lfloor d/2 \rfloor} \left(\frac{n}{\rho_0}\right)^{c_1} \le b_1 (\rho_0 \log \rho_0)^{\lfloor d/2 \rfloor} \left(\frac{n}{\rho_0}\right)^{c_1},$$

which satisfies (2.1) for $\rho_0 = \rho_0(c_1)$ large enough. The only implication on r_0 is that it should exceed some constant depending on c_1.

The boundary of H^+ can be interpreted as the 0-th level of $\mathcal{A}(H)$ with respect to O. Predictably, levels in arrangements will play a great role in our analysis, so we close this subsection with a useful technical result about them. We say that a point p is at *level* k if the relative interior of the segment Op intersects exactly k hyperplanes of H. Let $f_k(H)$ denote the number of vertices of $\mathcal{A}(H)$ at level k. Clarkson and Shor [6] have shown that the sum $f_{\le k}(H) = \sum_{0 \le i \le k} f_i(H)$ is in $O(n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil})$. Using the same proof technique, we extend their result in the following manner. Let R be a subset of r hyperplanes in H and, for any $j \le d$, let $f_k(H, R, j)$ denote the number of vertices of $\mathcal{A}(H)$ at level k formed by the intersection of a j-face of R^+ and j hyperplanes of $H \setminus R$. (Note that the level function remains unchanged.)

Lemma 2.2.

$$f_{\le k}(H, R, j) = O\left(\left(r + \frac{n}{k+1}\right)^{\lfloor d/2 \rfloor} (k+1)^j\right).$$

2.2. Preprocessing the Semicutting. Let us assume that a semicutting R_i of size r_i has already been computed, along with its geode and a list of the hyperplanes of H meeting each simplex of the geode. We will show how to use that information to obtain a semicutting R_{i+1} of size $r_{i+1} > r_i$. To avoid overburdening the notation, we set $i = 1$. The sample R_2 is computed by forming the disjoint union of R_1 with a well-chosen subset $\bar{R}_2 \subseteq H$ of size $\bar{r}_2 = r_2 - r_1$. We can easily verify that a random $\bar{R}_2$ works well with high probability, and that the method of conditional probabilities can be employed to make the construction deterministic. We must now devise an approximation scheme to speed up the computation. The basic idea is to take each simplex s in the geode of R_1 and subdivide the portion of the arrangement of H within s into tiny simplices, which we can then treat as "fat" vertices. We need to specify how these tiny simplices are chosen and how to estimate their "fatness" (i.e., how many vertices of $\mathcal{A}(H)$ they enclose). We begin by assuming that the size $\bar{r}_2$ of $\bar{R}_2$ is neither too big nor too small:

$$c_1 r_1 \leq \bar{r}_2 \leq c_1^2 r_1 \leq \frac{n}{c_1}. \tag{2.2}$$

Let s be an (open) d-dimensional simplex of the geode of R_1. If few hyperplanes cross s, we say that s is light and we construct the arrangement of $H(s)$ in full. (We can afford to do that.) Otherwise, we construct a certain net Σ_s for $H(s)$, which we triangulate canonically. The vertices within any given face of Σ_s are all within roughly the same distance from O. To be able to count how many there are, we use ε-approximations. More precisely, for each simplex σ of the portion of the triangulation of Σ_s within s, we build an approximation for $H(\sigma)$. To specify these constructions, we need two parameters, ρ_s and ν_s, which we define below.

$$\rho_s = c_1 \left\lceil \frac{\bar{r}_2 V_s}{n - r_1} \right\rceil$$

and

$$\nu_s = c_0^2 (\rho_s \log \rho_s)^{d+\sqrt{c_1}}.$$

Definition. *If $V_s < \rho_s^2$, then the d-dimensional simplex s is light. Otherwise, it is heavy.*

The heavy case: Assume for the time being that the d-dimensional simplex s is heavy. Then, we compute a $(1/\rho_s)$-net Σ_s for $H(s)$ of size $O(\rho_s \log \rho_s)$, which we can do in time $V_s \rho_s^{O(1)}$ [12]. Let B be the set of hyperplanes bounding s and let $\mathcal{T}_s$ denote the portion within the closure of s of the canonical triangulation of $\mathcal{A}(\Sigma_s \cup B)$. (Note that this includes the boundary of s.) The simplices of $\mathcal{T}_s$ are relatively open and fall into $d+1$ categories: $\mathcal{T}_s(d)$ consists of all those simplices with no intersection with hyperplanes of R_1, and for $0 \leq j < d$, $\mathcal{T}_s(j)$ includes the simplices that intersect exactly $d-j$ hyperplanes of R_1. Note that the simplices of $\mathcal{T}_s(j)$ are of dimension at most, but not necessarily equal to, j. For a given $j \geq 0$, the union $\bigcup_s \mathcal{T}_s(j)$ might not be disjoint (because we take the closures of the simplices s), but by making some arbitrary conventions to remove duplicates, we can easily make it disjoint.

Given a simplex σ of $\mathcal{T}_s$, let $k_l(\sigma)$ (resp. $k_h(\sigma)$) be an upper (resp. lower) bound on the level of any point in σ. It is important for the efficiency of the algorithm to limit the number of different values of $k_l(\sigma)$ to a very small range. (This is to counter the global effect of temperature changes mentioned earlier.) We achieve this by making $k_l(\sigma)$ a power of two. Let $k(\sigma)$ be the number of hyperplanes separating O and σ *without* intersecting σ: if this number is 0, we set $k_l(\sigma) = 0$; else we have (logarithms to the base 2)

$$k_l(\sigma) = 2^{\lfloor \log k(\sigma) \rfloor}.$$

We also set

$$k_h(\sigma) = k(\sigma) + V_\sigma.$$

Note that

$$k_h(\sigma) \le V_s \qquad \text{and} \qquad V_\sigma \le \frac{dV_s}{\rho_s} \le \frac{n - r_1}{\bar{r}_2}. \tag{2.3}$$

For any point $v \in \sigma$, if $k(v)$ denotes the number of hyperplanes in H cutting the relative interior of Ov, we have

$$\frac{k(v)}{2} - \frac{n - r_1}{2\bar{r}_2} < k_l(\sigma) \le k(v) \le k_h(\sigma) \le k(v) + \frac{n - r_1}{\bar{r}_2}. \tag{2.4}$$

Next, for each simplex $\sigma \in \mathcal{T}_s$ such that $\nu_s < V_\sigma^{1/3}$, we compute a $(1/\nu_s)$-approximation A_σ for $H(\sigma)$ of size $O(\nu_s^2 \log \nu_s)$ in time $V_\sigma \nu_s^{O(1)}$ [12]. By Lemma 1.1, we know that the number of vertices formed by the approximation A_σ within σ can be used to approximate the number H_σ of vertices of $\mathcal{A}(H)$ in σ with an absolute error of at most $2j'V_\sigma^{j'}/\nu_s$, where $j' \le d$ is the dimension of σ. This means that we can estimate H_σ from above by H_σ^*, where $H_\sigma^* \le H_\sigma + 4dV_\sigma^{j'}/\nu_s$. If $\sigma \in \mathcal{T}_s(j)$, we have $j \ge j'$, therefore from (2.2,2.3),

$$0 \le H_\sigma^* - H_\sigma \le \frac{4d}{\nu_s}\Big(\frac{n - r_1}{\bar{r}_2}\Big)^j. \tag{2.5}$$

Actually, the lemma says that we can estimate the number of vertices in the j'-dimensional arrangement formed by H within σ, which might be greater than the number of vertices of $\mathcal{A}(H)$ in σ. To resolve this discrepancy, we simply observe that if σ lies within hyperplanes of H then the two quantities are the same and the estimation is valid. Otherwise, we know by the general position of H that the number of vertices of $\mathcal{A}(H)$ in σ must be 0.

If $\nu_s \ge V_\sigma^{1/3}$, then we compute the portion of $\mathcal{A}(H(\sigma))$ within σ, which takes $O(V_\sigma^{j'})$ time. This gives us H_σ directly, to which we set H_σ^*, thus making inequality (2.5) hold vacuously.

The light case: Assume now that s is light. In that case, we compute the arrangement of $H(s)$ in $O(V_s^d)$ time and keep in storage its portion within s. As a result, we can set Σ_s to be the full set $H(s)$. For each simplex $\sigma \in \mathcal{T}_s$, we dispense with the $(1/\nu_s)$-approximation altogether. By default, we can still write $H_\sigma^* = H_\sigma$, so that (2.5) holds vacuously. (We do this to avoid having to distinguish between light and heavy simplices all the time.) This concludes the preprocessing.

Now that we have most of the approximation tools needed to get the incremental computation of R_2 started, let us show that the errors introduced by using fat vertices are tolerable. Let $\bar{R}_2$ be a random sample of $\bar{r}_2$ hyperplanes chosen among the $n - r_1$ hyperplanes of $H \setminus R_1$. Given a vertex v of $\mathcal{A}(H)$, let $p(v)$ be the probability that v is a vertex of R_2^+, i.e., that the d hyperplanes defining v are in R_2 and no hyperplane cutting the relative interior of Ov is in R_2. The *energy* of the system is defined as the expectation of $\sum_e V_e^{c_1}$, where the sum extends over all segments e connecting O to the vertices of R_2^+. It is equal to $\sum_v p(v)k(v)^{c_1}$, which can be shown to be at most on the order of $r_2^{\lfloor d/2 \rfloor}(n/r_2)^{c_1}$, where $r_2 = |R_2|$. Thus, the method of conditional probabilities could be used to compute R_2, but this would require evaluating the energy $\sum_v p(v)k(v)^{c_1}$ exactly and maintaining it under dynamic conditionings. As it turns out, this is not a problem for the vertices within light simplices, but it is far too expensive for the others. Instead, we compute a coarse approximation of the energy by regarding the H_σ vertices within each $\sigma \in \mathcal{T}_s$ as one single "fat" vertex of weight H^*_σ and approximating $k(v)$, where $v \in \sigma$, from above and from below by $k_h(\sigma)$ and $k_l(\sigma)$, respectively.

Our estimation of the energy will be given by a quantity Φ, which is easy to compute. We write $\Phi = \sum_s \Phi(s)$, where s ranges over all d-dimensional simplices of the geode of R_1. If s is light, then we set $\Phi(s) = \sum_v p(v)k(v)^{c_1}$, where v runs through all the vertices of $\mathcal{A}(H)$ within s (minus the exclusions meant to avoid double counting mentioned earlier). If s is heavy, on the other hand, we set $\Phi(s) = \sum_{0\le j\le d} \Phi(s,j)$, with

$$\Phi(s,j) = \sum_{\sigma\in\mathcal{T}_s(j)} H^*_\sigma k_h^{c_1}(\sigma)\binom{n-r_1-k_l(\sigma)-j}{\bar{r}_2-j}\Big/\binom{n-r_1}{\bar{r}_2}.$$

Note that because of (2.1,2.2) all the binomial coefficients are well defined. Since we used only one-sided approximations it is obvious that Φ is an upper bound on the actual energy $\sum_v p(v)k(v)^{c_1}$. What is less immediate is that Φ is actually a fairly tight upper bound. From (2.1,2.2,2.4) we have

$$k_l(\sigma) \le V_s \le d \times \max_e V_e \le \frac{dn}{c_1} < \frac{n}{2} - r_2, \tag{2.6}$$

where e runs over all segments Ov, with v a vertex of R_1^+. With this inequality in hand, we can prove the following result.

Lemma 2.3. *For $c_0 = c_0(c_1)$ and $c_2 = c_2(c_1)$ large enough, the energy Φ is at most $c_2 r_2^{\lfloor d/2\rfloor}(n/r_2)^{c_1}$.*

The lemma shows that the approximation scheme is not too coarse, since Φ provides a fairly tight bound on the worst-case expectation of $\sum_e V_e^{c_1}$ (where the sum extends over all segments e connecting O to the vertices of R_2^+). The usefulness of Φ is that it can be calculated very efficiently.

2.3. The Error Analysis of the Conditioning Step. We compute $\bar{R}_2$ deterministically in an incremental fashion. Suppose that we have already committed the set $R \subseteq H \setminus R_1$ to $\bar{R}_2$, meaning that we must only find $\bar{r}_2 - r$ hyperplanes in H to complete the set $\bar{R}_2$, where $r = |R|$. To find the next element to be added to R, we might try out every candidate hyperplane $h \in H \setminus (R_1 \cup R)$, and choose the one that minimizes a certain function,

$$\Xi(R \cup \{h\}) = \phi(R \cup \{h\}) + \psi(R \cup \{h\}),$$

where ϕ is the normalized (estimated) energy of the system and ψ is a function used to ensure that not too many sample hyperplanes meet a given simplex s of the geode of R_1. The introduction of ψ is a technicality which is not essential to the intuition behind the algorithm. As we mentioned in the opening section of this paper, we shall use a variant of this selection criterion. Instead of picking the one hyperplane h that minimizes $\Xi(R \cup \{h\})$, we pick any one of them that makes $\Xi(R \cup \{h\})$ no greater than its average over all candidate hyperplanes.

We need an additional piece of notation: Given $X \subseteq H$ and a simplex σ of arbitrary dimension, the number of hyperplanes of $H \setminus X$ that meet the relative interior of σ without containing it can be written as $|(H \setminus X)(\sigma)|$; for convenience we use the shorthand $V_\sigma(X)$. Note that $V_\sigma = V_\sigma(\emptyset)$. Now, let s be a d-dimensional simplex of the geode of R_1. Conditioning upon the inclusion $R \subseteq \bar{R}_2$, let $\psi(R)$ be the (conditional) expectation of the sum

$$\frac{1}{c_0 c_3 r_1^{\lfloor d/2 \rfloor}} \sum_s |\bar{R}_2(s)|^{c_1},$$

where c_3 depends only c_1. We observe that $\psi(R)$ is easy to evaluate since it can be expressed as

$$\psi(R) = \frac{1}{c_0 c_3 r_1^{\lfloor d/2 \rfloor}} \sum_s \sum_{0 \le k \le V_s(R)} (|R(s)|+k)^{c_1} \binom{n - r_1 - r - V_s(R)}{\bar{r}_2 - r - k} \binom{V_s(R)}{k} \Big/ \binom{n - r_1 - r}{\bar{r}_2 - r}, \quad (2.9)$$

where $\binom{a}{b} = 0$ if $b < 0$; note that because of (2.1,2.2), $b \le a$.

Lemma 2.4. *For an appropriate $c_3 = c_3(c_1)$, we have $\psi(\emptyset) < 1/4$.*

The upper bound of $1/4$ gives us a little maneuvering room, within which we can relax the evaluation of $\psi(R)$ and allow round-off errors. However, the errors involved in evaluating the expectation of $\sum_e V_e^{c_1}$ are, as we already saw, of a totally different kind: they stem from clumping together bunches of vertices. To complete the specification of $\Xi(R) = \phi(R) + \psi(R)$, we define the normalized estimated energy $\phi(R)$ by the equation,

$$\Phi(R) = 8 c_2 r_2^{\lfloor d/2 \rfloor} \left(\frac{n}{r_2}\right)^{c_1} \phi(R),$$

where $\Phi(R)$ is an estimation of the conditional expectation of $\sum_e V_e^{c_1}$. It is an approximation of the energy of the system after insertion of R. Accordingly, we have $\Phi(\emptyset) = \Phi$. The formal definition of $\Phi(R)$ will be given shortly. If s is light, again all calculations are performed exactly, so let us assume that s is heavy until indicated otherwise.

Definition. *Given $\sigma \in \mathcal{T}_s$, let $\mathcal{T}_\sigma(R)$ be the portion within σ of the canonical triangulation of the arrangement formed by $R(\sigma)$ and the hyperplanes bounding σ (recall that $R(\sigma)$ denotes the set of hyperplanes in R meeting σ without containing it). We define $\mathcal{T}_s(R, j)$ as consisting of the faces of $\mathcal{T}_\sigma(R)$ that intersect exactly $d - j$ hyperplanes of $R_1 \cup R$, over all $\sigma \in \mathcal{T}_s$. By extension, $\mathcal{T}_s(R)$ is the set of faces in the triangulations $\mathcal{T}_\sigma(R)$ for all $\sigma \in \mathcal{T}_s$.*

Note that since each σ might be triangulated independently, there is no guarantee that $\mathcal{T}_s(R)$ is itself a cell complex. It is, however, a partition of s into simplices of all dimensions. Consistently with our notation, we have $\mathcal{T}_s(j) = \mathcal{T}_s(\emptyset, j)$ and $\mathcal{T}_s = \mathcal{T}_s(\emptyset)$. Any $\sigma' \in \mathcal{T}_s(R, j)$ lies within a unique $\sigma \in \mathcal{T}_s$, so we can extend the definition of k_h and k_l, with the convention that $k_l(\sigma') = k_l(\sigma)$ and $k_h(\sigma') = k_h(\sigma)$.

We now introduce the technical concept of a *witness set.* If $k(\sigma) = 0$, the witness set is empty. Otherwise, among the $k(\sigma)$ hyperplanes separating O and σ without intersecting σ, choose $k_l(\sigma)$ of them arbitrarily once and for all. That choice determines the witness set of σ. Note that more than half of the candidates end up in the witness set. We define $\delta(R,\sigma)$ to be 1 if R does not intersect the witness set of σ and 0 otherwise. Naturally, we set $\delta(R,\sigma') = \delta(R,\sigma)$. Using witness sets is the price we must pay for replacing $k(\sigma)$ by its approximation $k_l(\sigma)$. The function δ is used in defining the estimated energy $\Phi(R)$: its role is to ensure that $\Phi(R)$ can still be regarded as a higher moment of a random variable.

To define $H^*_{\sigma'}$, again we distinguish between two cases. If $\nu_s \geq V_\sigma^{1/3}$, then we compute $H^*_{\sigma'} = H_{\sigma'}$ directly, otherwise we use the $(1/\nu_s)$-approximation A_σ for $H(\sigma)$ previously computed. By Lemma 1.1, this allows us to estimate the number $H_{\sigma'}$ of vertices of $\mathcal{A}(H)$ in σ' with an absolute error of at most $2dV_\sigma^{j'}/\nu_s$, where j' is the dimension of σ'. (Notice that the subscript of V_σ is not primed.) Since $j \geq j'$, this means that we can estimate $H_{\sigma'}$ from above by $H^*_{\sigma'}$, with (σ being the simplex of $\mathcal{T}_s$ containing σ')

$$0 \leq H^*_{\sigma'} - H_{\sigma'} \leq \frac{4dV_\sigma^j}{\nu_s}. \tag{2.10}$$

Returning to the general step in the incremental construction of R_2, we consider a random choice of $\bar{R}_2$ and we denote by R the subset already committed to $\bar{R}_2$ at that stage. Given a vertex v of $\mathcal{A}(H)$, let $p(R,v)$ be the conditional probability that $v \in R_2^+$, i.e., the d hyperplanes defining v are in the disjoint union $R_2 = R_1 \cup \bar{R}_2$ and no hyperplane cutting the relative interior of Ov is in R_2, given that $R \subseteq \bar{R}_2$. We define the estimated energy after insertion of R,

$$\Phi(R) = \sum_s \Phi(R,s),$$

where s ranges over all d-dimensional simplices of the geode of R_1. If s is light, then $\Phi(R,s) = \sum_v p(R,v)k(v)^{c_1}$, where v runs through all the vertices of $\mathcal{A}(H)$ within s (minus the usual exclusions between simplices to avoid overcounting). If s is heavy, then we break down $\Phi(R,s)$ along dimensions $0,\ldots,d$: $\Phi(R,s) = \sum_{0\leq j\leq d}\Phi(R,s,j)$, where

$$\Phi(R,s,j) = \sum_{\sigma'\in\mathcal{T}_s(R,j)} \delta(R,\sigma')H^*_{\sigma'}k_h^{c_1}(\sigma')\binom{n-r_1-r-k_l(\sigma')-j}{\bar{r}_2-r-j}\Big/\binom{n-r_1-r}{\bar{r}_2-r}.$$

The energy of the system after insertion of R is equal to the conditional expectation of $\sum_e V_e^{c_1}$, given $R \subseteq \bar{R}_2$, where the sum extends over all segments e connecting O to the vertices of R_2^+. This energy is equal to $\sum_v p(R,v)k(v)^{c_1}$, which is at most $\Phi(R)$. This follows from the fact that again only upper bounds are used in the definition of $\Phi(R,s,j)$. Note also that $\Phi(R)$ is actually an upper bound on the conditional expectation of $\sum_e V_e^{c_1}$, where the sum extends over all segments e connecting O to the vertices of $\mathcal{A}(R_2)$ in R_1^+ whose corresponding witness sets do not intersect R_2; this includes the vertices of R_2^+ but also possibly many others. After insertion of $\bar{R}_2$ no randomness is left, so we have,

Lemma 2.5. *The estimated energy $\Phi(\bar{R}_2)$ is an upper bound on $\sum_e V_e^{c_1}$, where the sum extends over all segments e connecting O to the vertices of R_2^+.*

Let $C = H \setminus (R_1 \cup R)$. Ideally, we would like to prove that

$$\frac{1}{|C|} \sum_{h \in C} \Phi(R \cup \{h\}) = \Phi(R),$$

which is the necessary condition for applying the method of conditional probabilities. Of course, this does not hold in our case because errors accumulate, and all we can hope for is that the average value of $\Phi\big(R \cup \{h\}\big)$, over all $h \in C$, does not exceed $\Phi(R)$ by too much. So, our immediate goal is to prove an upper bound on

$$\Delta(R) = -\Phi(R) + \frac{1}{n - r_1 - r} \sum_{h \in C} \Phi\big(R \cup \{h\}\big),$$

in order to bound the drift away from the average caused by the estimations made at every step of our incremental selection of $\bar{R}_2$.

Lemma 2.6. *For $c_0 = c_0(c_1)$ and $r < \bar{r}_2$,*

$$\Delta(R) \leq \frac{1}{c_0 \bar{r}_2} \Big(\frac{n}{r_1}\Big)^{\sqrt{c_1}} \sum_s |R(s)|^d V_s^{c_1 - \sqrt{c_1}}.$$

Remark: Let us interpret the lemma informally, in a way that is technically incorrect but intuitively on the mark. On the average, V_s should be on the order of n/r_1 and $|R(s)|$ should be a constant, so the upper bound might look like $(a/c_0\bar{r}_2) r_1^{\lfloor d/2 \rfloor} (n/r_1)^{c_1}$, where a is a constant dependent only on c_1. Since there are $\bar{r}_2$ steps and c_0 can be made arbitrarily large *independently* of c_1, the final error is bounded by $\varepsilon r_1^{\lfloor d/2 \rfloor} (n/r_1)^{c_1}$, where ε can be chosen arbitrarily small. This makes the relative error smaller than any fixed positive constant. The presence of strange exponents such as $c_1 - \sqrt{c_1}$ is a technicality motivated by our later use of Hölder's inequality.

Recall that the algorithm starts off with R empty and fills it in, one element at a time, until $r = |R| = \bar{r}_2$. To add one new element to R, the rule is that out of all the hyperplanes in $C = H \setminus (R_1 \cup R)$ we choose one, h_0, such that

$$\Xi\big(R \cup \{h_0\}\big) \leq \frac{1}{|C|} \sum_{h \in C} \Xi\big(R \cup \{h\}\big).$$

We will show later how to find h_0. Because of our upper bound on $\Delta(R)$ (Lemma 2.6) we can show that if we abide by our selection rule the upward drift of $\Xi(R)$ is slow enough. In what might appear as a slight contradiction to what we said earlier, the proof relies on the fact that R_1 is a semicutting. This seems to indicate that in order to keep the energy function bounded it is important to have it bounded at previous steps. Note that this would not be necessary in a perfectly randomized algorithm. The subtle point here is that we need the induction not to bound the energy function per se, but to prove that the error in its estimation is bounded. From Lemma 2.1 and the fact that R_1 is a semicutting, we have

$$\sum_s V_s^{c_1} \leq c_1^{c_1} \sum_e V_e^{c_1} \leq c_0 c_1^{c_1} r_1^{\lfloor d/2 \rfloor} \Big(\frac{n}{r_1}\Big)^{c_1}. \tag{2.18}$$

It is now easy to establish the following inequality.

Lemma 2.7. *For any r such that $0 \leq r \leq \bar{r}_2$,*

$$\Xi(R) < \frac{1}{2} - \frac{\bar{r}_2 - r}{8\bar{r}_2}.$$

The lemma implies that $\Xi(\bar{R}_2) < 1/2$, which has the double implication:

$$\sum_s |\bar{R}_2(s)|^{c_1} < c_0 c_3 r_1^{\lfloor d/2 \rfloor}, \tag{2.19}$$

$$\Phi(\bar{R}_2) < 4c_2 r_2^{\lfloor d/2 \rfloor} \Big(\frac{n}{r_2}\Big)^{c_1}.$$

From Lemma 2.5, this shows that setting $c_0 = c_0(c_1, c_2)$ big enough ensures that R_2 is a semicutting, which achieves our first goal.

2.4. Computing the Next Sample. Now that the error analysis is behind us, we must describe how to find $\bar{R}_2$ efficiently. One has to be quite careful, lest truly enormous running times befall us. From now on, to simplify our notation, every step of the complexity analysis will be correct only up to within constant additive and multiplicative factors. Suppose that R_1^+ is given to us, along with the list of hyperplanes of $H \setminus R_1$ separating each vertex of R_1^+ from O. With this information, it is routine to compute the geode and the lists of hyperplanes crossing its faces in time $r_1^{\lfloor d/2 \rfloor} + \sum_s V_s$. (Again, recall that we have set $i = 1$ for notational convenience.) By collecting the various mentions of running times made in Section 2.2, we easily find that all further preprocessing (computing all Σ_s's and the various $(1/\nu_s)$-approximations) can be accomplished in time

$$\sum_s V_s(\rho_s \nu_s)^{O(1)} \leq \sum_s \Big(1 + \Big(\frac{r_2 V_s}{n}\Big)^{b_1\sqrt{c_1}}\Big) V_s, \tag{2.20}$$

where b_1 is independent of c_1. For simplicity, we have overlooked the costs associated with light simplices and trivial $(1/\nu_s)$-approximations: these will be accounted for later. Recall that computing R_0, and hence its geode, takes $O(n)$ time.

Let us now turn to the incremental construction of R and evaluate the cost of adding one more element to that set. From now on, the symbol R will be used exclusively to denote a member of the set sequence leading from $\emptyset$ to $\bar{R}_2$. Inductively, we assume that not only the geode of R_1 is available, but also that for each heavy d-dimensional simplex s the following information has been computed:

(i) The net Σ_s and all $(1/\nu_s)$-approximations for s; recall that these structures do not depend on R but only on R_1.

(ii) The triangulation $\mathcal{T}_\sigma(R)$ of each $\sigma \in \mathcal{T}_s$.

(iii) If s is light, the entire portion of the arrangement formed by H within s is available explicitly.

Computing $\Xi(R)$ involves summing up various quantities associated with each simplex s. For a given s, one of these quantities, denoted $\psi(R,s)$, depends on $|R(s)|$ and $V_s(R)$ and is needed only for the evaluation of $\psi(R)$. The others enter the expression of $\Phi(R,s) = \sum_{0 \leq j \leq d} \Phi(R,s,j)$. Recall that

$$\Phi(R,s,j) = \sum_{\sigma' \in \mathcal{T}_s(R,j)} \delta(R,\sigma') H^*_{\sigma'} k_h^{c_1}(\sigma') \binom{n - r_1 - r - k_l(\sigma') - j}{\bar{r}_2 - r - j} \Big/ \binom{n - r_1 - r}{\bar{r}_2 - r}.$$

As it turns out, it is very unfortunate that the binomial coefficients should depend on r. Intuitively, one would hope that adding a new hyperplane would cause work only in the vicinity of that hyperplane. But because r must be increased by one, every binomial coefficient must be updated, and technically every single feature in $\bigcup_s \mathcal{T}_s(R)$ must be looked at. This could be a nightmare scenario, if we didn't use some special data structuring tricks. To begin with, we break up $\Phi(R,s)$ into terms with the same binomial coefficients. We write

$$\Phi(R,s) = \sum_{j,k} F(R,s,j,k) B(r,j,k),$$

with

$$B(r,j,k) = \binom{n - r_1 - r - k - j}{\bar{r}_2 - r - j} \Big/ \binom{n - r_1 - r}{\bar{r}_2 - r},$$

and

$$F(R,s,j,k) = \sum \delta(R,\sigma') H^*_{\sigma'} k_h^{c_1}(\sigma'),$$

where the sum extends over all $\sigma' \in \mathcal{T}_s(R,j)$ such that $k_l(\sigma') = k$. An entry $B(r,j,k)$ is 0 if its binomial coefficients are out of range. We can make (j,k) run over the same range for any s by setting $F(s,j,k) = 0$ appropriately. It is now clear that

$$\Xi(R) = \sum_{j,k} \sum_s f(R,s,j,k) B(r,j,k),$$

where $f(R,s,j,k)$ is $F(R,s,j,k)$ scaled by a factor independent of r,s,j,k; for convenience, we extend the sum to $(j,k) = (-1,-1)$, with $f(R,s,-1,-1) = \psi(R,s)$ and $B(r,-1,-1) = 1$.

Unfortunately, we cannot afford to test $\Xi(R \cup \{h\})$ for each h every time we want to add one more element to R. This is why we modified the Raghavan-Spencer strategy by seeking not the minimum value of $\Xi(R \cup \{h\})$, but any one that does not exceed the average. The entire construction of $\bar{R}_2$ revolves around a particular family of matrices $\{ M(R) : |R| = 0, 1, \ldots, \bar{r}_2 \}$, each of them indexed by (h,j,k), where $h = 1, \ldots, n - r_1$ designates a hyperplane of $H \setminus R_1$ and, as usual, (j,k) indexes the $B()$'s. We look at (j,k) as indexing the elements of a single row by implicitly mapping all the (j,k)'s injectively to an interval of integers: the maximum number of such indices is denoted

μ; since $k_l(\sigma')$ is a power of two, $\mu = O(\log n)$. By abuse of notation, h will be used both as a row index and as a hyperplane. If $h \in R_1 \cup R$, then $M(R)[h,j,k] = 0$, otherwise

$$M(R)[h,j,k] = \sum_s f\big(R \cup \{h\}, s, j, k\big).$$

Note that $\Xi\big(R \cup \{h\}\big)$ is the inner product of the h-th row of $M(R)$ with the row indexed r of the matrix B. Let us break up the interval $1, \ldots, n - r_1$ into at most $\sqrt{n}$ intervals $\Lambda_1, \ldots, \Lambda_p$, of length less than $\sqrt{n}$. We define the row-vector (indexed by (j,k))

$$M_i^+(R) = \sum_{h \in \Lambda_i} M(R)[h,j,k].$$

To find the next h to be included into R, we first compute the index i_0 that minimizes, over all i, the average value of $\Xi\big(R \cup \{h\}\big)$ over Λ_i, i.e.,

$$\frac{1}{|\Lambda_i|} \sum_{j,k} M_i^+(R)[j,k] \times B(r,j,k).$$

Next, we check each index in Λ_{i_0} and set the new hyperplane h_0 to the index i that minimizes $\Xi\big(R \cup \{h\}\big)$ over Λ_{i_0}, i.e.,

$$\sum_{j,k} M(R)[i,j,k] \times B(r,j,k).$$

We can easily check that this maneuver is consistent with our selection strategy and homes in on a good hyperplane. But how costly are the computations? The key to efficiency is to evaluate the matrices in an incremental fashion and make them into persistent data structures. To do so, we define the row-vector

$$M(R)[0,j,k] = \sum_s f(R,s,j,k),$$

and express $M(R)[h,j,k]$ in terms of the operator Δ, where

$$\Delta M(R)[h,j,k] = M(R)[h,j,k] - M(R)[0,j,k].$$

We also define $\Delta_i^+ M(R)$ by summing up the rows of $\Delta M(R)$ indexed by Λ_i.

For a given s, let $t(s)$ be the maximum time, over each R computed by the incremental algorithm, spent computing all the values in $\bigcup_{j,k}\{f(R,s,j,k)\}$. For convenience, we also include in $t(s)$ the cost of computing $\mathcal{T}_s(R)$ from scratch (but not the net Σ_s and the $(1/\nu_s)$-approximations within s). Two simple observations form the basis of our analysis. One is that for a given pair (R,s), we can compute the vector $f(R,s,j,k)$, assumed to be initialized to 0, in $t(s)$ time; the initialization takes μ time. The other observation is that if $h \in H \setminus (R_1 \cup R)$ does not meet s then the vectors $f\big(R \cup \{h\}, s, j, k\big)$ and $f(R,s,j,k)$ are identical.

Thus, we can compute the entire row $M(R)[0,j,k]$ in time at most $\mu + \sum_s t(s)$. The purpose of using $\Delta M(R)$ is that any row of the difference matrix can be evaluated by looking only at the neighborhood of h across the triangulations $\big\{\, \mathcal{T}_s(R) \;:\; s \in J(h) \,\big\}$, where $J(h)$ denotes the set of simplices s that h meets. More precisely, it can be evaluated by restricting ourselves to the symmetric difference between $\bigcup_{s\in J(h)} \mathcal{T}_s(R)$ and $\bigcup_{s\in J(h)} \mathcal{T}_s(R\cup\{h\})$. Of course, this entails updating $\bigcup_{s\in J(h)} \mathcal{T}_s(R)$ into $\bigcup_{s\in J(h)} \mathcal{T}_s(R\cup\{h\})$. We call the complete set of operations (computing the new triangulations in the vicinity of h and then evaluating f at the proper places) "tracing the incidences of h over each $s \in J(h)$." In general, we update the triangulations for the sole purpose of evaluating a row of $\Delta M(R)$, in which case we immediately "undo" those updates right after the evaluation is completed, so as not to corrupt the geometric structures. Because of our use of $t(s)$ in the complexity analysis, every time we update some triangulation within s, we might as well recompute $\mathcal{T}_s(R)$ entirely from scratch. The cost of setting up row h of $\Delta M(R)$ is at most

$$\mu + \sum_{s\in J(h)} t(s).$$

Thus, the time for computing the row $M(R)[0,j,k]$ and initializing $\Delta M(R)$ and $\Delta_i^+ M(R)$ for $R=\emptyset$ is at most

$$n\mu + \sum_s t(s) + \sum_{h\in H\setminus R_1} \sum_{s\in J(h)} t(s),$$

and hence, no more than

$$n\mu + \sum_s t(s)(V_s+1).$$

After precomputing all possible $B(r,j,k)$'s in $O(nr_2)$ time, we apply our selection algorithm and find the first hyperplane of $H\setminus R_1$ to be included into R. More generally, given $\Delta M(R)$, we find the next $h_0 \in H\setminus(R_1\cup R)$ to be added to R in time at most $\mu\sqrt{n}$. Summing up over the entire growth of R from $\emptyset$ to $\bar{R}_2$, we find a total "discovery" cost (excluding updates) of at most

$$nr_2 + \bar{r}_2\mu\sqrt{n}.$$

After adding h_0 to R, the next step is to update $\Delta M(R)$, meaning to convert it into $\Delta M(R\cup\{h_0\})$. But first we must compute $M(R\cup\{h_0\})[0,j,k]$. We observe that the difference

$$M(R\cup\{h_0\})[0,j,k] - M(R)[0,j,k]$$

can be computed by tracing the incidences of h_0, therefore setting up the row $M(R\cup\{h_0\})[0,j,k]$ takes time at most $\mu+\sum_{s\in J(h_0)} t(s)$. Since h_0 is not just a candidate but it is the elected hyperplane, we do not "undo" the new triangulations created while tracing the incidences of h_0 but we leave them in place.

We now turn to the updating of the matrix $\Delta M(R)$ itself, which is the most subtle part of the algorithm. First, we *mark* row h_0 as a way to indicate that the corresponding row in $M(R)$ is now

0. From our previous observations, we derive

$$\begin{aligned}
&\Delta M(R \cup \{h_0\})[h,j,k] - \Delta M(R)[h,j,k] \\
&= \sum_s \big(f(R \cup \{h_0, h\}, s, j, k) - f(R \cup \{h_0\}, s, j, k)\big) - \sum_s \big(f(R \cup \{h\}, s, j, k) - f(R, s, j, k)\big) \\
&= \sum_{s \notin J(h_0)} \big(f(R \cup \{h_0, h\}, s, j, k) - f(R \cup \{h_0\}, s, j, k)\big) - \sum_{s \notin J(h_0)} \big(f(R \cup \{h\}, s, j, k) - f(R, s, j, k)\big) \\
&+ \sum_{s \in J(h_0)} \big(f(R \cup \{h_0, h\}, s, j, k) - f(R \cup \{h_0\}, s, j, k)\big) - \sum_{s \in J(h_0)} \big(f(R \cup \{h\}, s, j, k) - f(R, s, j, k)\big) \\
&= \sum_{s \in J(h_0) \cap J(h)} \big(f(R \cup \{h_0, h\}, s, j, k) - f(R \cup \{h_0\}, s, j, k) + f(R, s, j, k) - f(R \cup \{h\}, s, j, k)\big).
\end{aligned}$$

Let $N(h_0)$ be the set of hyperplanes h such that $J(h) \cap J(h_0) \neq \emptyset$. From the identity above, it is clear that if h is not in $N(h_0)$ then the row $\Delta M(R \cup \{h_0\})[h,j,k]$ is exactly the same as $\Delta M(R)[h,j,k]$. Otherwise, it must be recomputed by tracing the incidences of h over the simplices $s \in J(h_0) \cap J(h)$. For any such h, we can find which entries of the row $\Delta M(R \cup \{h_0\})[h,j,k]$ must be updated in time $\sum_s t(s)$, where s runs over $J(h_0) \cap J(h)$. We must be very careful not to scan the entire row systematically but to visit *only* those entries that need updating. (As it turns out, failure to do so would make the algorithm nonoptimal in dimensions 4 and 5.) Counted over all rows, the time for these operations is at most

$$n - r_1 + \sum_{h \in N(h_0)} \sum_{s \in J(h_0) \cap J(h)} t(s).$$

Finally, we must update each $\Delta_i^+ M(R)$, which does not take longer than the time to update $\Delta M(R)$, added to the time μ needed to reflect the effect of marking row h_0. Over all R's, this gives a total update time of at most

$$\bar{r}_2(n - r_1 + \mu) + \sum_{h_0} \sum_{h \in N(h_0)} \sum_{s \in J(h_0) \cap J(h)} t(s),$$

where h_0 ranges over all $\bar{r}_2$ hyperplanes inserted into R. Once $\bar{R}_2$ is available, we can retrieve R_2^+ directly from our various triangulations, along with the lists of hyperplanes crossing the edges joining O to the vertices of R_2^+. This takes an additional $r_2^{\lfloor d/2 \rfloor} + \sum_e V_e$ time, which is at most $n r_2^{\lfloor d/2 \rfloor - 1}$, because R_2 is a semicutting. To summarize, the entire construction of $\bar{R}_2$ takes time at most

$$n r_2^{\lfloor d/2 \rfloor - 1} + n\mu + \sum_s t(s)(V_s + 1) + n r_2 + \bar{r}_2 \mu \sqrt{n} + \bar{r}_2(n - r_1 + \mu) + \sum_{h_0} \sum_{s \in J(h_0)} t(s)(V_s + 1),$$

which, because $d > 3$, is no more than

$$\mu n^{3/2} + n r_2^{\lfloor d/2 \rfloor - 1} + \sum_s \big(|\bar{R}_2(s)| + 1\big) t(s)(V_s + 1). \tag{2.21}$$

If s is light, then $t(s) \le V_s^d$, which is either constant or less than ρ_s^{2d}. Suppose that s is heavy. There are at most on the order of $(\rho_s \log \rho_s)^d(|\bar{R}_2(s)|+1)^d$ simplices σ' in $\mathcal{T}_s(R)$. For each of them, estimating the number $H^*_{\sigma'}$ of vertices inside it takes time at most $\nu_s^{O(1)}$, if $\nu_s < V_\sigma^{1/3}$, where σ is the simplex of $\mathcal{T}_s$ containing σ'. Otherwise, it takes time at most $V_\sigma^d \le \nu_s^{3d}$. Note that both $k_l(\sigma')$ and $k_h(\sigma')$ are available from the preprocessing. Thus, again up to within additive and multiplicative constant factors,

$$t(s) \le |\bar{R}_2(s)|^d + \left(\frac{r_2 V_s}{n}\right)^{b_2\sqrt{c_1}}(|\bar{R}_2(s)|+1)^d, \tag{2.22}$$

where b_2 is independent of c_1. From (2.20–2.22) and $\mu = O(\log n)$ we find that the time $T(r_2)$ to compute R_2^+ is at most

$$n^{3/2}\log n + nr_2^{\lfloor d/2\rfloor-1} + \left(\frac{r_2}{n}\right)^{b_1\sqrt{c_1}}\sum_s V_s^{1+b_1\sqrt{c_1}}$$

$$+\sum_s(|\bar{R}_2(s)|+1)^{d+1}(V_s+1) + \left(\frac{r_2}{n}\right)^{b_2\sqrt{c_1}}\sum_s(|\bar{R}_2(s)|+1)^{d+1}(V_s+1)^{1+b_2\sqrt{c_1}}.$$

By Cauchy's inequality and (2.18,2.19), we have

$$\sum_s(|\bar{R}_2(s)|+1)^{d+1}(V_s+1) \le \sqrt{\sum_s(|\bar{R}_2(s)|+1)^{2d+2}} \times \sqrt{\sum_s(V_s+1)^2} < b_3 n r_1^{\lfloor d/2\rfloor-1}$$

and

$$\sum_s(|\bar{R}_2(s)|+1)^{d+1}(V_s+1)^{1+b_2\sqrt{c_1}}$$

$$\le \sqrt{\sum_s(|\bar{R}_2(s)|+1)^{2d+2}} \times \sqrt{\sum_s(V_s+1)^{2+2b_2\sqrt{c_1}}} < b_4 r_1^{\lfloor d/2\rfloor}\left(\frac{n}{r_1}\right)^{1+b_2\sqrt{c_1}}.$$

It follows from (2.18) that

$$T(r_2) \le n^{3/2}\log n + nr_2^{\lfloor d/2\rfloor-1} + \left(\frac{r_2}{r_1}\right)^{b_1\sqrt{c_1}} nr_1^{\lfloor d/2\rfloor-1} + \left(\frac{r_2}{r_1}\right)^{b_2\sqrt{c_1}} nr_1^{\lfloor d/2\rfloor-1}.$$

Because of (2.2) we finally derive

$$T(r_2) \le n^{3/2}\log n + nr_2^{\lfloor d/2\rfloor-1}.$$

Because $d > 3$ and the r_i's follow a geometric progression, summing up over all $r_1, r_2, \ldots$, we find a running time of $O(n^{\lfloor d/2\rfloor})$.

Remark: Don't let the inequality for T fool you: When $d = 2, 3$, there is an extra term nr_2, which implies that the algorithm is quadratic in two and three dimensions.

2.5. Completing the Construction. Because of condition (2.2) we can apply the general recursive step only as long as $\bar{r}_i$ is less than about n/c_1. Since on the other hand, r_i can grow exponentially fast, we are not forced to stop until r_i differs from n by a constant factor. What remains for us to do then is to take the last semicutting R_i and complete the computation. We just follow our usual routine of inserting new hyperplanes h_0 into a set R initially set to $\emptyset$. This time, however, we operate without following any selection criterion, and without using any data structure, besides the lists of hyperplanes $H(s)$ associated with the simplices s of the geode of R_i. By computing the full arrangements within each simplex, we trivially complete the computation in time $\sum_s (V_s + 1)^{b_1}$ (b_1 independent of v_1), which, because of Lemma 2.1 and the fact that R_i is a semicutting, is $r_i^{\lfloor d/2 \rfloor}(n/r_i)^{b_1} = O(n^{\lfloor d/2 \rfloor})$. In conjunction with the two and three-dimensional convex hull algorithms of Preparata and Hong [18], this completes the proof of our main result.

Theorem 2.8. *It is possible to compute the convex hull of n points in d-space deterministically in* $O(n \log n + n^{\lfloor d/2 \rfloor})$ *time, which is optimal.*

There are several ways of interpreting a Voronoi diagram of n points in d-space as a convex hull or an intersection of halfspaces in $(d+1)$-space [2,8,9]. We have the immediate corollary:

Theorem 2.9. *The Voronoi diagram of n points in d-space can be computed in time* $O(n \log n + n^{\lceil d/2 \rceil})$, *which is optimal.*

5. Searching as a Process in Statistical Thermodynamics

In order to cope with the various obstacles we encountered along the way, we had to modify the method of conditional probabilities in several ways: For example, we replaced the greedy selection criterion by a "kinder and gentler" one. The notion of a witness set was introduced to limit the number of distinct possible distances, but this seems a rather unnatural artifice. More serious still, the errors we accumulate at each conditioning step all but destroy the most basic properties of conditional probabilities. We can make an analogy with statistical thermodynamics to help us conceptualize and build some intuition for what is happening. This will also give us an opportunity to quantify more discriminatingly how the search zeroes in on its target. In particular, it will allow us to answer the natural question: How random-looking are the samples R_i computed by our algorithm? In the Clarkson-Shor algorithm, a sample R_i is a prefix of a random permutation and therefore is itself random. This implies, in particular, that the vertices of R_i^+ lie at an average distance roughly $n/|R_i|$ from O. Is that always true of our deterministic samples? The answer is a resounding no. There can actually be some rather remarkable phenomenon occurring: Measure the speed *V-det* at which the deterministic R_i^+ shrinks toward O, as i increases, and compare it against the speed *V-rand* of its random counterpart. Depending on the shape of the arrangement $\mathcal{A}(R)$ it might be that the two speeds follow each other fairly closely. But we can easily construct a case where *V-det* exceeds *V-rand* by a strikingly huge amount to begin with, only to even up much later. This is quite surprising, especially since our modification of the Raghavan-Spencer method makes the search mimic its randomized counterpart even more closely: indeed, recall that in our variant the conditional expectations might remain equal to the initial expectation at all steps of the selection process. We will explain how such a huge deviation from a random behavior is at all possible.

But first, we must build our "thermodynamics" model.† For simplicity we will concentrate on the problem of finding a semicutting R of size r, assuming that we have all the needed ε-nets and ε-approximations. Recall that for R to be a semicutting, a certain higher-order moment of the distance distribution between O and the vertices of R^+ must be bounded from above. This moment, denoted $E(R)$, is the *energy* of the system. We are faced with an optimization problem, in which a certain function $E(R)$ must be minimized over all $R \subseteq H$ of a given size. Of course, minimization is not quite the right term, because for our purposes simply reaching as low as roughly the level of the mean value of $E(R)$ is good enough. Moreover a random sample R works fine with at least a fixed probability.

A. Three Classical Approaches. The most simplistic attitude toward optimization is greed. In our context, the *greedy* strategy would be to add hyperplanes into the initially empty set R, one by one, always selecting the hyperplane that minimizes the gradient of the energy $E(R)$. This "instant gratification" strategy is likely to fail because it does not worry about the long-term consequences of a move: Thus, the search might begin on a promising, steep downhill course, only to later find itself "stuck" in a plateau for a long time.

Simulated annealing [11,17] attempts to remedy this problem by randomizing the direction of the next move in the hope that a random walk will guide us more surely toward the global minimum. It also allows for uphill (i.e., locally bad) moves in order to keep the search from getting stuck at local minima. The *temperature* of the system controls how likely we are to accept a bad move as a function of how uphill it goes. By analogy with the thermal motion of atoms, the probability of acceptance is chosen so as to make the system evolve into a Boltzmann distribution: in that distribution, the probability of a variation of energy of ΔE is equal to $e^{-\Delta E/kt}$, where k is Bolztmann's constant and t is the temperature. Once a temperature is set it is not lowered again until the physical system reaches equilibrium. (Translation: keep the random walk going until it attains its stationary distribution.) The soundness of the scheme is predicated on the assumption that deep wells in the graph of E in configuration space should have wide openings: thus, randomizing the moves at a given temperature makes it more likely to fall into a wide well (which is true), and hence, one that contains the global minimum (which is more a matter of faith).

The method of conditional probabilities is both deterministic and greedy. Its range of application is limited by the fact that it requires the ability to estimate the density of good targets reachable from any point. It is greedy in the sense that it opts for a move that, roughly speaking, maximizes the estimated density of good targets over the subspace of configurations reachable from the current point (without backtracking). In the combinatorial setting in which it is usually defined, the search can be modeled as the traversal of a "greedy" path in the tree of all possible sample choices: the nodes are labeled after their associated conditional probabilities.

† The analogy should not be taken too literally. It is used mostly to build up intuition for what is happening. As such, we believe that it serves a valuable purpose.

B. A Thermodynamic Process. Think of the vertices of the arrangement $\mathcal{A}(H)$ as particles of a gas in equilibrium in a closed environment. At any given time, imagine that the full set R has been chosen by Nature, but that our experimental knowledge of R is only partial; therefore, according to the ensemble method of statistical mechanics, we postulate that any R compatible with our current knowledge is equally probable. From now on, we will use R to denote the state of our current knowledge, i.e., the subset of hyperplanes which the search has already committed at a given time.

Each hyperplane in R has a specific physical effect on the whole system, but hyperplanes outside R have no influence. On a macroscopic level, the system has a well-defined temperature and energy at any given time. The initial temperature is r. Once a hyperplane h is added into R, the temperature t drops by one, and this transition causes changes in the thermodynamical state of the system in a manner specific to h. Thus, searching for R amounts to cooling down the system until it freezes (zero temperature), at which point R becomes entirely known to us. Quantifying the uncertainty in our knowledge of the system in terms of its entropy is unlikely to reveal much about its dynamic structure. Energy considerations will prove much more useful.

At nonzero temperature, the energy is known only thermodynamically, i.e., statistically by its mean over all remaining allowable configurations of the system. We can now state the fundamental link between the method of conditional probabilities and statistical thermodynamics: Selecting one more hyperplane into R may move the system to a different mean energy level, but *the average over all possible selections is exactly the same as the mean energy before the transition.* As often is the case in physics, however, the reality is a little more complex than in the case of an idealized environment. In particular, to take into account the error terms of the conditioning step, we must assume that in in order to be activated a transition requires a small transfer of energy from the outside. These are minor inputs of external energy, which taken over the entire cooling, do not amount to more than a fixed fraction of the original mean energy of the system (Lemmas 2.6–2.7). Notice that energy is a discrete quantity. (That was to be expected, right? Quantum mechanics tells us, by way of Schrödinger's equation, that the energy states of a system are the eigenvalues of its Hamiltonian operator, and hence, are discrete. So, everything fits beautifully, doesn't it?) †

At the microscopic level, each particle is in one of $d+1$ *phases* at any time: initially, all particles are in phase 0, and any decrease in temperature can only force a particle to move up to a higher phase or to stay in the same phase. The phase of a particle corresponds to the number of hyperplanes known to be in R that pass through it. The particles on the boundary of R^+ (which includes more than just the vertices of R^+) form what we call the *shrinking shell*: it consists of all those particles in a nonzero phase. The energy of any particle is in one of several quantum states, again known only by its statistical average. At each temperature transition, the particles incident upon the newly added hyperplane move up to a phase one higher than the one they are currently in; the others remain in the same phase. It is now easy to interpret our selection strategy:

Pick any move as long as it does not increase the mean energy of the system beyond the amount necessary to activate the transition.

† This is a joke.

In the worst case, therefore, ignoring the activation energy, the original mean energy is conserved and simply redistributed among the particles. Observe that the original Raghavan-Spencer selection criterion is greedy and picks the move that minimizes the mean energy. We now want to look more closely at how the redistribution of energy takes place. In particular, we are interested in the energetic contribution of the shell. How does it vary? Is it ever dominant, and if yes, when is it so?

C. The Attila-the-Hun Effect. At temperature t, where should we expect the shell to be? Since $r-t$ hyperplanes have already been selected, by analogy with the random case, we might expect the average distance of a vertex of the shell to O to be around $n/(r-t+1)$. This seems a reasonable rule of thumb, especially if most of the hyperplanes in H are incident upon H^+. If the arrangement makes it impossible to form polyhedra with large numbers of facets, however, then this estimate might be completely off, as we shall discuss later. This is what we call the *jolt effect,* which is a rather interesting phenomenon in itself. But for the time being let us assume that the radius of the shell is about $n/(r-t+1)$.

In the first few transitions, the shell has too few vertices to contain much energy. As it shrinks toward the center O, however, the shell accumulates particles and slowly gathers energy as a result. Without loss of generality, assume that the initial mean energy is conserved throughout the cooling, which is the worst case scenario. Then, with regard to the system as a whole, the shell appears to be sucking in energy from the other particles. Let us analyze this intriguing phenomenon in greater detail. (Warning: all estimates in this section will be given up to within a constant factor.)

The particles enclosed by the shell but not on it form the *core* of the system: all particles in the core are in phase 0. Particles outside the shell and the core are devoid of any energy, so the total mean energy of the system consists of (1) the core energy and (2) the shell energy. A particle in phase ϕ and at distance k from the center O has mean energy roughly

$$k^{c_1}\left(\frac{t}{n}\right)^{d-\phi}e^{-tk/n}.$$

If we ignore the fact that the shell acts as a barrier, we find that the core energy is roughly

$$\sum_{k\geq 0} n^{\lfloor d/2\rfloor}k^{\lceil d/2\rceil-1+c_1}\left(\frac{t}{n}\right)^{d}e^{-tk/n},$$

which is on the order $t^{\lfloor d/2\rfloor}(n/t)^{c_1}$. Of course, this is not quite correct because particles beyond the shell are inactive. This means that the summation should not extend past $n/(r-t+1)$. Therefore, our previous assessment is correct provided that $n/t < n/(r-t+1)$, i.e., the temperature exceeds about $r/2$. In the worst case, the total mean energy is conserved thoughout the process, and stands as roughly $r^{\lfloor d/2\rfloor}(n/r)^{c_1}$. Then, prior to the halfway mark, that is, as long as the temperature is as high as $r/2$ (we use 2 as a generic constant here), the core energy is dominant in the whole system. Then, around the halfway markpoint, shell and core energies break even, and from that point on, the shell becomes dominant and keeps stealing energy from the core until nothing is left and the core freezes.

The shell is almost always greatly outpopulated by the core, so how can it manage to steal so much energy? The answer is to be found in the phase transitions. Particles at higher phases have exponentially more energy. Specifically, to move up one phase gives the particle a multiplicative energy boost of n/t. Therefore, although small in size, the shell contains high-energy particles. Recall that a core particle at distance k stores an amount of energy on the order of $E_t = k^{c_1}(t/n)^d e^{-tk/n}$. As t drops by one, assuming that the particle stays in the same phase, we have (for k not too large),

$$\frac{E_{t-1}}{E_t} = \left(1-\frac{1}{t}\right)^d e^{k/n} = 1 + \frac{k}{n} - \frac{d}{t} + O\left(\frac{1}{t^2} + \frac{k^2}{n^2}\right).$$

Up to within a constant factor, we see that the energy of the particle increases until the temperature drops to roughly dn/k, and decreases thereafter. As a whole, the core energy remains about constant between temperatures r and $r/2$ (remember that we do not care about constant factor variations). During the second part of the cooling, however, the core energy vanishes to 0 polynomially fast.

We conclude that after the halfway mark, not only the shell collects the energy of the particles that it sweeps over, but it also steals energy from the core. This is what we might call the "*Attila-the-Hun effect.*" As the shell shrinks toward the center, it effectively kills the particles that its sweeps over and steals their energy for itself; at the same time, it sucks in and absorbs energy from the core, all the while constantly squeezing the core toward the center (actually toward a ball of radius n/r).

D. The Jolt. If very few hyperplanes of R are incident upon R^+, for any R, then a remarkable phenomenon occurs: the shell is given an initial jolt and shrinks extremely fast at the beginning. Then the shrinking slows down until the "random" shell eventually catches up with it. (This is the shell that corresponds to the randomized version of the algorithm: its radius is $n/(r-t+1)$ at temperature t.) Thus, the shell greatly outperforms its randomized counterpart at the early stages. How is that possible? The whole algorithm revolves around the notion of ε-nets, for which randomization usually helps. Furthermore, our variant of the Raghavan-Spencer method makes it mimic the randomized algorithm even more closely, so how can one explain this *jolt effect*? To understand this mystery, let us consider a specific example: We set $d=2$ and $c_1=3$, meaning that we are in two dimensions and the energy is defined in terms of the third moment of the distance-to-center distribution. The set H consists of n lines tangent to the bottom half of a fixed circle and placed at regular intervals around it. If we choose H^+ to be the upper envelope of the lines, then the shell and its random counterpart behave in a similar fashion. The interesting case is to choose H^+ to be the lower envelope. Note that H^+, or for that matter any R^+, is just a simple wedge. To simplify calculations, we will use a continuous approximation of the energy in terms of the incomplete gamma function.

There are about x vertices of $\mathcal{A}(H)$ at distance x from O, therefore at temperature r, the energy of the system is

$$E_0 = \int_0^n x\left(\frac{r}{n}\right)^2 e^{-rx/n}x^3\,dx.$$

Using integration by parts (or even better, Mathematica), it is easy to find that

$$E_0 = 24\Big(\frac{n}{r}\Big)^3 - \frac{e^{-r}}{n}\Big(4n^4 + \frac{24n^4}{r^3} + \frac{24n^4}{r^2} + \frac{12n^4}{r} + n^4 r\Big).$$

At temperature t, the shell is a wedge whose supporting lines pass through, say, the kth leftmost point and the l-th rightmost point of contact on the half-circle. Without much loss of generality, assume that we have perfect symmetry, i.e., $k = l$. If the first $r-t$ lines were picked at random then we should have $k = l = k_t$ on average, where $k_t = n/(r-t+1)$. But a calculation shows that if it were the case the mean energy would have to rise to a level much higher than E_0, which is ruled out by our energy conservation policy. So, what is k_t? If we were trying to minimize the energy of the system at every step, as in the original Raghavan-Spencer method, then we might expect a much smaller value of k_t, on the grounds that we would thus be trying to be more clever than a random pick. But recall that by staying at the same energy level we are actually trying to keep conditional expectations equal to the original expectation. This can be interpreted as an attempt to mimic the randomized algorithm as closely as possible. So, how much lower than $n/(r-t+1)$ should we expect to find k_t? Probably not much, right? Wrong. We can actually show that

$$k_t = \frac{n}{t}\log\Big(\frac{r^3}{r^3 - t^3}\Big),$$

which means that as soon as the temperature drops, k_t immediately plummets to about a multiplicative factor of $\log r$ off its final value of n/r, while in the random case, k_t barely begins to drop. Here is the detailed calculation:

$$E_t = E_t^{(0)} + E_t^{(1)} + E_t^{(2)},$$

where $E_t^{(i)}$ is the mean energy of the particles in phase i. We have

$$\begin{aligned} E_t^{(0)} &= \int_0^{k_t} x\Big(\frac{t}{n}\Big)^2 e^{-tx/n}x^3\,dx + \int_{k_t}^{2k_t}(2k_t - x)\Big(\frac{t}{n}\Big)^2 e^{-tx/n}x^3\,dx \\ &= 24\Big(\frac{n}{t}\Big)^3 + 2e^{-2\alpha}\Big(\frac{n}{t}\Big)^3\big(12 + 18\alpha + 12\alpha^2 + 4\alpha^3 - e^{\alpha}(24 + 18\alpha + 6\alpha^2 + \alpha^3)\big), \end{aligned}$$

where $\alpha = tk_t/n$. We also have

$$E_t^{(1)} = \int_{k_t}^{2k_t} \frac{2t}{n} e^{-tx/n}x^3\,dx = 2e^{-2\alpha}\Big(\frac{n}{t}\Big)^3\big(e^{\alpha}(6 + 6\alpha + 3\alpha^2 + \alpha^3) - 6 - 12\alpha - 12\alpha^2 - 8\alpha^3\big)$$

and

$$E_t^{(2)} = 8\alpha^3 e^{-2\alpha}\Big(\frac{n}{t}\Big)^3.$$

It follows that

$$E_t = 24\Big(\frac{n}{t}\Big)^3 + 6e^{-2\alpha}\Big(\frac{n}{t}\Big)^3\big(2 + 2\alpha - e^{\alpha}(6 + 4\alpha + \alpha^2)\big).$$

By conservation of energy, we have $E_t = E_0$. It is not possible to obtain a closed form for this equation, but we can approximate its solution by writing it as

$$\Big(\frac{n}{r}\Big)^3 = \Big(1 - \frac{1}{e^{\alpha}}\Big)\Big(\frac{n}{t}\Big)^3,$$

which gives the approximate value

$$k_t = \frac{n}{t}\log\Big(\frac{r^3}{r^3 - t^3}\Big).$$

To appreciate the magnitude of the jolt at the very beginning of the cooling, assume that $r - t$ is a small constant. Then, as claimed earlier, k_t is about $(n/r)\log r$, while in the randomized case, k_t is still about n.

Here is a probabilistic game which illustrates, in simpler form, what the jolt effect is really all about. A lottery produces a sequence of 10 random reals between 0 and 100: the maximum value is your gain. That is, if the numbers are, say, $20, 34, 12, 89, 45, 62, 49, 32, 38, 72$, then you go home with 89 dollars. It is easy to see that your expected gain is \$90.9. Suppose now that someone offers you an alternative option: you are guaranteed a minimum gain of 70 dollars, no matter what, but as a penalty, you can play with only 9 random numbers, as opposed to 10. If the maximum of these 9 numbers exceeds 70, you walk home with that amount, otherwise you pocket \$70. Which game should you play? Your intuition might tell you that in game I, the first number will average to \$50, and hence, should play basically no role in determining your final gain (since it is to be around \$90.9, anyway). So, switching to game II might seem advantageous. This reasoning is false. As it turns out, the safety net value at which you should switch to game II is not \$70 but \$78.68 or above. The reason the break-even point is so far above 50 is that we must compensate for the fact that derandomizing the first pick reduces its variance to 0. Clearly, the high expected gain is due to the large (upper part of the) variance of the uniform distribution. Another way to see the relationship with the higher moments of the distribution is to recall that the L_c metric (the "moment" metric) converges toward the L_∞ metric (the "max" metric), as c goes to infinity.

6. Concluding Remarks

We are hopeful that this work will open a new opportunity for the powerful theory of ε-nets by providing the tools for speeding up algorithms via approximate computations. The moral of this story is that ε-nets and ε-approximations are not to be used solely for divide-and-conquer purposes. Specifically, we suspect that the techniques in this paper can improve the deterministic computation of shallow cuttings, k-levels, k-th order Voronoi diagrams, etc. More generally, it will be interesting to find out whether the same techniques can be used to derandomize the other probabilistic incremental algorithms used in computational geometry. Many examples seem to fall squarely in the framework we build in this paper, but some others seem more challenging: the latter category includes, for example, Clarkson and Shor's algorithm for computing the diameter of a set of points in 3-space [6].

Acknowledgments: I wish to thank L. Lovász, J. Matoušek, and P. Raghavan for several helpful discussions.

Appendix

Lemma 2.1. *Given the geode of $R \subseteq H$, for any constant c large enough, $\sum_s V_s^c \leq c^c \sum_e V_e^c$, where the first sum is taken over all d-dimensional simplices s of the geode and the second one over all segments e connecting O to the vertices of the geode.*

Proof: Because it cannot pass through O, any hyperplane of H meeting a relatively open simplex s of nonzero dimension without containing it must intersect (without containing) the relative interior of at least one of the segments connecting its vertices to O. Thus, it suffices to prove the inequality obtained by substituting W_s for V_s, where W_s is the number of hyperplanes crossing at least one of those segments outside its endpoints. We will show by induction on k that, for any k-face f of R^+, the sum $\sum_s W_s^c$, denoted A_f, where s ranges over the faces of the geode lying within the closure of f, is at most

$$d^k (2^c+1)^k \sum_{e \in E} V_e^c,$$

where E is the set of edges joining O to the vertices of f. The case $k = 0$ is obvious, so let assume that $k > 0$.

We observe that by our choice of the lifting vertex, we have $A_f \leq (2^c + 1) \sum_g A_g$, where g ranges over all the $(k-1)$-faces of R^+ incident upon f. The term 1 comes from the contribution of the faces incident upon f, while the term 2^c accounts (conservatively) for the effect of the lifting vertex to the contribution of the geode faces within f. By induction, we have

$$\sum_g A_g \leq \mu d^{k-1} (2^c+1)^{k-1} \sum_{e \in E} V_e^c,$$

where μ represents the maximum multiplicity of a segment e in the counting. But the general position of H, and hence of R, ensures that a segment e is incident upon d hyperplanes, and therefore, k $(k-1)$-faces incident upon f. It follows that $\mu \leq d$, which completes the inductive proof. The case $k = d$, where f is the interior of R^+, gives the lemma. ∎

Lemma 2.2.

$$f_{\leq k}(H,R,j) = O\left(\left(r + \frac{n}{k+1}\right)^{\lfloor d/2 \rfloor} (k+1)^j\right).$$

Proof: Let $s = \lfloor n/(2k+2) \rfloor$; the lemma is trivial if k is on the order of n, so we can assume that s is large enough. Let S be a random sample of $H \setminus R$ obtained by picking each hyperplane independently with probability s/n, and let V be the number of vertices of the polyhedron $R^+ \cap S^+$ that are formed by the intersection of a j-face of R^+ with j hyperplanes of $H \setminus R$. Using elementary tail estimates for the binomial distribution, we find that the expected value of V is $O\big((r+s)^{\lfloor d/2 \rfloor}\big)$. It is also equal to

$$\sum_{k \geq 0} f_k(H,R,j) \left(\frac{s}{n}\right)^j \left(1 - \frac{s}{n}\right)^k \geq f_{\leq k}(H,R,j) \left(\frac{s}{n}\right)^j e^{-2ks/n} \geq e^{-1} \left(\frac{s}{n}\right)^j f_{\leq k}(H,R,j),$$

from which the lemma follows. ∎

Lemma 2.3. *For $c_0 = c_0(c_1)$ and $c_2 = c_2(c_1)$ large enough, the energy Φ is at most $c_2 r_2^{\lfloor d/2 \rfloor}(n/r_2)^{c_1}$.*

Proof: Let us first concentrate on the case where s is heavy.

$$\Phi(s,j) \le \sum_{\sigma \in \mathcal{T}_s(j)} H_\sigma^* k_h^{c_1}(\sigma)\Big(\frac{\bar{r}_2}{n-r_1-k_l(\sigma)}\Big)^j \Big(1-\frac{k_l(\sigma)}{n-r_1}\Big)^{\bar{r}_2}.$$

From (2.5,2.6) we derive that

$$\Phi(s,j) \le \sum_{\sigma \in \mathcal{T}_s(j)} b_1 \left(\frac{1}{\nu_s}\Big(\frac{n-r_1}{\bar{r}_2}\Big)^j + H_\sigma\right) k_h^{c_1}(\sigma)\Big(\frac{\bar{r}_2}{n}\Big)^j e^{-k_l(\sigma)\bar{r}_2/(n-r_1)}.$$

The r.h.s. consists of two parts: one, $A(s,j)$, depends on H_σ and the other, $B(s,j)$, is the additive error term arising from approximating H_σ as H_σ^*. From (2.4) it follows that

$$\begin{aligned}\sum_s A(s,j) &= \sum_s \sum_{\sigma \in \mathcal{T}_s(j)} b_1 H_\sigma k_h^{c_1}(\sigma)\Big(\frac{\bar{r}_2}{n}\Big)^j e^{-k_l(\sigma)\bar{r}_2/(n-r_1)} \\ &\le b_1\sqrt{e}\Big(\frac{\bar{r}_2}{n}\Big)^j \sum_s \sum_{v \in \mathcal{T}_s(j)} \Big(k(v)+\frac{n-r_1}{\bar{r}_2}\Big)^{c_1} e^{-\frac{1}{2}k(v)\bar{r}_2/(n-r_1)},\end{aligned}$$

where by abuse of notation, "$v \in \mathcal{T}_s(j)$" means that v is a vertex of $\mathcal{A}(H)$ that lies in some $\sigma \in \mathcal{T}_s(j)$. Expanding the sum along levels of $\mathcal{A}(H)$, we have

$$\sum_s A(s,j) \le b_1\sqrt{e}\Big(\frac{\bar{r}_2}{n}\Big)^j \sum_{k\ge 0}\Big(k+\frac{n-r_1}{\bar{r}_2}\Big)^{c_1} f_k(H,R_1,j)e^{-\frac{1}{2}k\bar{r}_2/(n-r_1)}.$$

Splitting the sum at $k=k_0$, where k_0 is on the order of $(n-r_1)/\bar{r}_2$, we easily find from Lemma 2.2 that, up to within a constant factor (dependent on c_1), the first part is dominated by

$$\Big(\frac{\bar{r}_2}{n}\Big)^j \Big(\frac{n-r_1}{\bar{r}_2}\Big)^{c_1}\Big(r_1+\frac{n\bar{r}_2}{n-r_1}\Big)^{\lfloor d/2\rfloor}\Big(\frac{n-r_1}{\bar{r}_2}\Big)^j.$$

To handle the second part, we use Lemma 2.2 but now we apply summation by parts beforehand. In other words, using the identity

$$\sum_{k_0}^n u_k v_k = \sum_{k_0}^n (u_k-u_{k+1})\sum_{k_0}^k v_i + u_{n+1}\sum_{k_0}^n v_i,$$

we derive a similar upper bound. It follows that

$$\sum_s A(s,j) \le b_2\Big(\frac{\bar{r}_2}{n}\Big)^j\Big(\frac{n-r_1}{\bar{r}_2}\Big)^{c_1+j}\Big(r_1+\frac{n\bar{r}_2}{n-r_1}\Big)^{\lfloor d/2\rfloor},$$

for $b_2=b_2(c_1)$. From (2.2) we find that, for $b_3=b_3(c_1)$,

$$\sum_s A(s,j) \le b_3 r_2^{\lfloor d/2\rfloor}\Big(\frac{n}{r_2}\Big)^{c_1}, \tag{2.7}$$

It follows from (2.2,2.3) that the error term $B(s,j)$ satisfies the inequality

$$\begin{aligned} B(s,j) &= \sum_{\sigma \in \mathcal{T}_s(j)} \frac{b_1}{\nu_s}\Big(\frac{n-r_1}{\bar r_2}\Big)^j k_h^{c_1}(\sigma)\Big(\frac{\bar r_2}{n}\Big)^j e^{-k_l(\sigma)\bar r_2/(n-r_1)} \\ &\le \frac{b_1}{\nu_s}\sum_{\sigma\in\mathcal{T}_s(j)} V_s^{c_1} \le \Big(\frac{1}{\nu_s}\Big) b_1 c_1 (\rho_s \log \rho_s)^d V_s^{c_1} \le \frac{b_4}{c_0^2}\Big(\frac{n}{r_1}\Big)^{\sqrt{c_1}} V_s^{c_1-\sqrt{c_1}}, \end{aligned}$$

for $b_4 = b_4(c_1)$. Since R_1 is a semicutting it follows from Lemma 2.1 and (2.2) that

$$\sum_s B(s,j) \le \frac{b_5}{c_0} r_2^{\lfloor d/2 \rfloor}\Big(\frac{n}{r_2}\Big)^{c_1} \le r_2^{\lfloor d/2\rfloor}\Big(\frac{n}{r_2}\Big)^{c_1}, \tag{2.8}$$

for $b_5 = b_5(c_1)$ and $c_0 = c_0(c_1)$ big enough. We have assumed so far that s is heavy. If it is light, then the bounds (2.7–2.8) hold by default since no approximations are introduced in the specification of $\Phi(s,j)$. The lemma follows. ∎

Lemma 2.4. *For an appropriate $c_3 = c_3(c_1)$, we have $\psi(\emptyset) < 1/4$.*

Proof: Let $k_0 > 0$ be a parameter to be set later. The expectation of $|\bar R_2(s)|^{c_1}$ is at most $k_0^{c_1} + A$, where

$$A = \sum_{k>k_0} k^{c_1}\binom{n-r_1-V_s}{\bar r_2 - k}\binom{V_s}{k}\Big/\binom{n-r_1}{\bar r_2}.$$

We find that

$$A \le \sum_{k>k_0}\Big(\frac{\bar r_2}{n-r_1-\bar r_2+k}\Big)^k\Big(1-\frac{V_s}{n-r_1}\Big)^{\bar r_2-k}\Big(\frac{eV_s}{k}\Big)^k k^{c_1} \le \sum_{k>k_0}\Big(\frac{b_1\bar r_2 V_s}{nk}\Big)^k k^{c_1}.$$

Setting $k_0 = 2b_1\bar r_2 V_s/n$ implies that

$$A \le \sum_{k\ge 0} 2^{-k}k^{c_1} = b_2(c_1).$$

Thus, the expectation of $\sum_s |\bar R_2(s)|^{c_1}$ is at most $\sum_s b_3\lceil \bar r_2 V_s/n\rceil^{c_1}$, with $b_3 = b_3(c_1)$. From Lemma 2.1 and the fact that R_1 is a semicutting, we have

$$\sum_s V_s^{c_1} \le c_1^{c_1}\sum_e V_e^{c_1} \le c_0 c_1^{c_1} r_1^{\lfloor d/2\rfloor}\Big(\frac{n}{r_1}\Big)^{c_1}.$$

The lemma now follows trivially from (2.2) and the Upper Bound Theorem. ∎

Lemma 2.6. *For $c_0 = c_0(c_1)$ and $r < \bar r_2$,*

$$\Delta(R) \le \frac{1}{c_0\bar r_2}\Big(\frac{n}{r_1}\Big)^{\sqrt{c_1}}\sum_s |R(s)|^d V_s^{c_1-\sqrt{c_1}}.$$

Proof: Proving the lemma is not so easy because the faces of neighboring dimensions interact together in the counting argument, and we must deal with all of them at once. Let s be a d-dimensional simplex of the geode of R_1, and assume for the time being that s is heavy. Given $\sigma \in \mathcal{T}_s$ and $h \in C$, let V be the set of vertices of $\mathcal{A}(H)$ that lie in σ and in the intersection of exactly j hyperplanes of C. We define $H_{\sigma,j-1}(h)$ (resp. $H_{\sigma,j}(h)$) as the number of vertices of V that lie in h (resp. not in h). As usual, we use the star superscript to refer to approximations, so $H^*_{\sigma,j-1}(h)$ and $H^*_{\sigma,j}(h)$ denote the *estimated* values of these quantities, as they are appear in $\Phi(R,s)$. Specifically, $H^*_{\sigma,j-1}(h) = 0$ if $j = 0$; otherwise

$$H^*_{\sigma,j-1}(h) = \sum_{\sigma' \in S_1(j-1,\sigma)} H^*_{\sigma'}, \tag{2.11}$$

where $S_1(j-1,\sigma)$ is the set of all simplices of $\mathcal{T}_s(R \cup \{h\}, j-1)$ that lie in $\sigma \cap h$. Note that these simplices are at most $(j-1)$-dimensional: they lie in the intersection of h with $d-j$ hyperplanes of $R_1 \cup R$. From (2.10) we derive

$$0 \le H^*_{\sigma,j-1}(h) - H_{\sigma,j-1}(h) \le \frac{4d}{\nu_s} \sum_{\sigma' \in S_1(j-1,\sigma)} V_\sigma^{j-1} \le \frac{c_1}{\nu_s} |R(\sigma)|^d V_\sigma^{j-1},$$

therefore from (2.3),

$$0 \le H^*_{\sigma,j-1}(h) - H_{\sigma,j-1}(h) \le \frac{c_1}{\nu_s} |R(\sigma)|^d \Big(\frac{n-r_1}{\bar{r}_2}\Big)^{j-1}. \tag{2.12}$$

Similarly, we have

$$H^*_{\sigma,j}(h) = \sum_{\sigma' \in S_2(j,\sigma)} H^*_{\sigma'}, \tag{2.13}$$

where $S_2(j,\sigma)$ is the set of all simplices of $\mathcal{T}_s(R \cup \{h\}, j)$ lying in $\sigma \setminus h$. The same derivations show that

$$0 \le H^*_{\sigma,j}(h) - H_{\sigma,j}(h) \le \frac{c_1}{\nu_s} |R(\sigma)|^d \Big(\frac{n-r_1}{\bar{r}_2}\Big)^{j}. \tag{2.14}$$

We also define

$$H^*_{\sigma,j}(R) = \sum_{\sigma' \in S_3(j,\sigma)} H^*_{\sigma'}, \tag{2.15}$$

where $S_3(j,\sigma)$ is the set of all simplices of $\mathcal{T}_s(R,j)$ lying in σ. Note that for any $h \in C$ that does *not* meet σ, we have $H^*_{\sigma,j}(R) = H^*_{\sigma,j}(h)$. The following quantity,

$$A_\sigma(j) = \binom{n - r_1 - r - k_l(\sigma) - j - 1}{\bar{r}_2 - r - j - 1} \Big/ \binom{n - r_1 - r - 1}{\bar{r}_2 - r - 1},$$

defined for $r < \bar{r}_2$, is useful to express some of the terms involved in the expression for $\Phi(R,s,j)$. If any term in the binomial coefficient is negative then we set $A_\sigma(j) = 0$. Note that because of (2.1,2.2) the coefficients cannot be of the form $\binom{a}{b}$, where $a < b$. Next, we introduce some additional intermediate quantities based on $A_\sigma(j)$ and $A_\sigma(j-1)$:

$$T_{\sigma,j}(h) = A_\sigma(j-1) H_{\sigma,j-1}(h) + A_\sigma(j) H_{\sigma,j}(h),$$

$$T^*_{\sigma,j}(h) = A_\sigma(j-1)H^*_{\sigma,j-1}(h) + A_\sigma(j)H^*_{\sigma,j}(h).$$

Obviously, $T^*_{\sigma,j}(h)$ is an estimation of $T_{\sigma,j}(h)$ from above. If s is light, then all calculations are performed exactly, and for this reason, we define all starred quantities to be equal to their non-starred counterparts. In view of the following inequalities,

$$\sum_{h\in C\cap H(\sigma)} H_{\sigma,j-1}(h) \le jH^*_{\sigma,j}(R),$$

$$\sum_{h\in C\cap H(\sigma)} \left(H_{\sigma,j}(h) + H_{\sigma,j-1}(h)\right) \le |C\cap H(\sigma)|H^*_{\sigma,j}(R) = V_\sigma(R)H^*_{\sigma,j}(R),$$

the elementary algebraic manipulations below lead to (2.16):

$$\sum_{h\in C\cap H(\sigma)} T_{\sigma,j}(h) \le A_\sigma(j)V_\sigma(R)H^*_{\sigma,j}(R) + \sum_{h\in C\cap H(\sigma)} \left(A_\sigma(j-1) - A_\sigma(j)\right)H_{\sigma,j-1}(h),$$

therefore

$$\left(n - r_1 - r - k_l(\sigma) - V_\sigma(R)\right)A_\sigma(j)H^*_{\sigma,j}(R) + \sum_{h\in C\cap H(\sigma)} T_{\sigma,j}(h)$$

is at most

$$\left(n - r_1 - r - k_l(\sigma)\right)A_\sigma(j)H^*_{\sigma,j}(R) + \left(A_\sigma(j-1) - A_\sigma(j)\right)jH^*_{\sigma,j}(R),$$

and hence,

$$\left(n - r_1 - r - k_l(\sigma) - V_\sigma(R)\right)A_\sigma(j)H^*_{\sigma,j}(R) + \sum_{h\in C\cap H(\sigma)} T_{\sigma,j}(h) \le (\bar{r}_2 - r)A_\sigma(j-1)H^*_{\sigma,j}(R). \quad (2.16)$$

Note that if the simplex s is light then (2.16) still holds. To derive an upper bound on $\Delta(R)$ we break it down into simpler components. To do so, we define

$$\Delta'(R,s,j) = -\Phi(R,s,j) + \frac{1}{n - r_1 - r}\sum_{h\in C}\Phi'(R\cup\{h\},s,j),$$

where

$$\Phi'(R\cup\{h\},s,j) = \sum_{\sigma\in\mathcal{T}_s} \delta(R\cup\{h\},\sigma)T^*_{\sigma,j}(h)k_h^{c_1}(\sigma).$$

Note that although Φ and Φ' are different, they are related in the following way: Because of (2.11,2.13) and the fact that $S_1(d,\sigma) = \emptyset$ and

$$\mathcal{T}_s(R\cup\{h\},j) = \bigcup_{\sigma\in\mathcal{T}_s} \left(S_1(j,\sigma)\cup S_2(j,\sigma)\right),$$

we derive

$$\begin{aligned}\sum_{0\le j\le d}\Phi'(R\cup\{h\},s,j) &= \sum_{1\le j\le d}\sum_{\sigma\in\mathcal{T}_s}\sum_{\sigma'\in S_1(j-1,\sigma)}\delta(R\cup\{h\},\sigma')A_{\sigma'}(j-1)H^*_{\sigma'}k_h^{c_1}(\sigma')\\ &\qquad+\sum_{0\le j\le d}\sum_{\sigma\in\mathcal{T}_s}\sum_{\sigma'\in S_2(j,\sigma)}\delta(R\cup\{h\},\sigma')A_{\sigma'}(j)H^*_{\sigma'}k_h^{c_1}(\sigma')\\ &= \sum_{0\le j\le d}\sum_{\sigma\in\mathcal{T}_s}\sum_{\sigma'\in S_1(j,\sigma)}\delta(R\cup\{h\},\sigma')A_{\sigma'}(j)H^*_{\sigma'}k_h^{c_1}(\sigma')\\ &\qquad+\sum_{0\le j\le d}\sum_{\sigma\in\mathcal{T}_s}\sum_{\sigma'\in S_2(j,\sigma)}\delta(R\cup\{h\},\sigma')A_{\sigma'}(j)H^*_{\sigma'}k_h^{c_1}(\sigma')\\ &= \sum_{0\le j\le d}\sum_{\sigma'\in\mathcal{T}_s(R\cup\{h\},j)}\delta(R\cup\{h\},\sigma')A_{\sigma'}(j)H^*_{\sigma'}k_h^{c_1}(\sigma')\\ &= \sum_{0\le j\le d}\Phi(R\cup\{h\},s,j).\end{aligned}$$

It follows that

$$\Delta(R)=\sum_{s,j}\Delta'(R,s,j),$$

where the sum extends over all d-dimensional simplices s of the geode of R_1 and all j between 0 and d. We have

$$\Phi(R,s,j)=\sum_{\sigma'\in\mathcal{T}_s(R,j)}\delta(R,\sigma')\Big(\frac{\bar r_2-r}{n-r_1-r}\Big)A_{\sigma'}(j-1)H^*_{\sigma'}k_h^{c_1}(\sigma'),$$

and therefore

$$\Phi(R,s,j)=\sum_{\sigma\in\mathcal{T}_s}\delta(R,\sigma)\Big(\frac{\bar r_2-r}{n-r_1-r}\Big)A_\sigma(j-1)H^*_{\sigma,j}(R)k_h^{c_1}(\sigma). \tag{2.17}$$

From (2.15) the contribution within σ to $\sum_{h\in C}\Phi'(R\cup\{h\},s,j)$ of each of the $n-r_1-r-k_l(\sigma)-V_\sigma(R)$ hyperplanes of C that avoid both σ and its witness set is $\delta(R,\sigma)A_\sigma(j)H^*_{\sigma,j}(R)k_h^{c_1}(\sigma)$, therefore we find that $\sum_{h\in C}\Phi'(R\cup\{h\},s,j)$ is equal to

$$\sum_{\sigma\in\mathcal{T}_s}\Big(\big(n-r_1-r-k_l(\sigma)-V_\sigma(R)\big)A_\sigma(j)H^*_{\sigma,j}(R)+\sum_{h\in C\cap H(\sigma)}T^*_{\sigma,j}(h)\Big)\delta(R,\sigma)k_h^{c_1}(\sigma).$$

Thus, in light of (2.16–2.17), we derive

$$\Delta'(R,s,j)\le\sum_{\sigma\in\mathcal{T}_s}\frac{\delta(R,\sigma)k_h^{c_1}(\sigma)}{n-r_1-r}\sum_{h\in C\cap H(\sigma)}\big(T^*_{\sigma,j}(h)-T_{\sigma,j}(h)\big).$$

From (2.2) we easily derive

$$A_\sigma(j)\le\Big(\frac{\bar r_2-r}{n-r_1-r}\Big)^j,$$

and from (2.3),

$$\Delta'(R,s,j)\le\frac{V_s^{c_1}}{n-r_1-r}\sum_{\sigma\in\mathcal{T}_s}\delta(R,\sigma)\sum_{h\in C\cap H(\sigma)}M(\sigma,h),$$

where

$$M(\sigma,h)=\big(H^*_{\sigma,j-1}(h)-H_{\sigma,j-1}(h)\big)\Big(\frac{\bar r_2-r}{n-r_1-r}\Big)^{j-1}+\big(H^*_{\sigma,j}(h)-H_{\sigma,j}(h)\big)\Big(\frac{\bar r_2-r}{n-r_1-r}\Big)^{j}.$$

If s is light, $M(\sigma,h)=0$. Otherwise, from (2.2,2.12,2.14) we derive $M(\sigma,h)\le c_1 2^d|R(\sigma)|^d/\nu_s$, therefore from (2.3),

$$\begin{aligned}\sum_{0\le j\le d}\Delta'(R,s,j)&\le\frac{(d+1)c_1 2^d V_s^{c_1}}{n-r_1-r}\Big(\frac{n-r_1}{\bar r_2}\Big)\sum_{\sigma\in\mathcal{T}_s}\frac{\delta(R,\sigma)|R(\sigma)|^d}{\nu_s}\\&\le\frac{b_1|R(s)|^d(\rho_s\log\rho_s)^d V_s^{c_1}}{c_0^2\bar r_2(\rho_s\log\rho_s)^{d+\sqrt{c_1}}}\\&\le\frac{b_1}{c_0^2\bar r_2}\Big(\frac{n}{\bar r_2}\Big)^{\sqrt{c_1}}|R(s)|^d V_s^{c_1-\sqrt{c_1}},\end{aligned}$$

for $b_1=b_1(c_1)$, which proves the lemma. ∎

Lemma 2.7. *For any r such that $0\le r\le\bar r_2$,*

$$\Xi(R)<\frac{1}{2}-\frac{\bar r_2-r}{8\bar r_2}.$$

Proof: We proceed by induction on r. Note that the lemma is true initially ($r=0$) by virtue of Lemmas 2.3 and 2.4. Assuming that it is true at step $r<\bar r_2$, we know that $\psi(R)<1/2$, and therefore,

$$\sum_s|R(s)|^{d\sqrt{c_1}}\le\mathbf{E}\sum_s|\bar R_2(s)|^{c_1}\le c_0c_3r_1^{\lfloor d/2\rfloor}.$$

By (2.18) and Hölder's inequality, this implies that

$$\sum_s|R(s)|^dV_s^{c_1-\sqrt{c_1}}\le\Big(\sum_s|R(s)|^{d\sqrt{c_1}}\Big)^{1/\sqrt{c_1}}\Big(\sum_s V_s^{c_1}\Big)^{1-1/\sqrt{c_1}}\le c_0b_1r_1^{\lfloor d/2\rfloor}\Big(\frac{n}{r_1}\Big)^{c_1-\sqrt{c_1}},$$

for some constant $b_1=b_1(c_1)$. From Lemma 2.6, it then follows that

$$\begin{aligned}\sum_{h\in C}\phi(R\cup\{h\})&\le(n-r_1-r)\phi(R)+\frac{1}{8c_0c_2r_2^{\lfloor d/2\rfloor}(n/r_2)^{c_1}}\Big(\frac{n-r_1-r}{\bar r_2}\Big)\Big(\frac{n}{r_1}\Big)^{\sqrt{c_1}}\sum_s|R(s)|^dV_s^{c_1-\sqrt{c_1}}\\&\le(n-r_1-r)\Big(\phi(R)+\frac{1}{8\bar r_2}\Big),\end{aligned}$$

for $c_2=c_2(c_1)$ large enough. Since, trivially, $\sum_{h\in C}\psi(R\cup\{h\})=(n-r_1-r)\psi(R)$, the new pick $h_0\in C$ satisfies

$$\Xi(R\cup\{h_0\})\le\frac{1}{n-r_1-r}\sum_{h\in C}\Xi(R\cup\{h\})\le\Xi(R)+\frac{1}{8\bar r_2}.$$

This implies that

$$\Xi(R\cup\{h_0\})<\frac{1}{2}-\frac{\bar r_2-(r+1)}{8\bar r_2},$$

which completes the proof. ∎

REFERENCES

1. Agarwal, P.K. *Partitioning arrangements of lines I: An efficient deterministic algorithm,* Disc. Comput. Geom. 5 (1990), 449–483.
2. Brown, K.Q., *Voronoi diagrams from convex hulls,* Inf. Proc. Lett. 9 (1979), 223–228.
3. Chazelle, B., *Cutting hyperplanes for divide-and-conquer,* Tech. Rep., CS–TR–335–91, June 1991, Princeton University.
4. Chazelle, B., Friedman, J. *A deterministic view of random sampling and its use in geometry,* Combinatorica 10 (1990), 229–249.
5. Clarkson, K.L. *Linear programming in* $O(n3^{d^2})$ *time,* Inf. Proc. Lett. 22 (1986), 21–24.
6. Clarkson, K.L., Shor, P.W. *Applications of random sampling in computational geometry, II*, Disc. Comput. Geom. 4 (1989), 387–421.
7. Dyer, M.E. *On a multidimensional search technique and its application to the Euclidean one-centre problem,* SIAM J. Comput. 15 (1986), 725–738.
8. Edelsbrunner, H. *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, Germany, 1987.
9. Edelsbrunner, H., Seidel, R. *Voronoi diagrams and arrangements,* Disc. Comput. Geom. 1 (1986), 25–44.
10. Haussler, D., Welzl, E. *Epsilon-nets and simplex range queries,* Disc. Comput. Geom. 2, (1987), 127–151.
11. Kirkpatrick, S., Gelatt, Jr., C.D., Vecchi, M.P. *Optimization by simulated annealing,* Science 220 (1983), 671–680.
12. Matoušek, J. *Approximations and optimal geometric divide-and-conquer,* Proc. 23rd Ann. ACM Symp. Theory of Comput. (1991), 505–511.
13. Matoušek, J. *Efficient partition trees,* KAM Series (tech. report) 90-175, Charles University, 1990.
14. Matoušek, J. *Cutting hyperplane arrangements,* to appear in Disc. Comput. Geom., 1991. Also, in Proc. 6th Ann. ACM Symp. Comput. Geom. (1990), 1–9.
15. Matoušek, J. *Reporting points in halfspaces,* manuscript, 1991.
16. Megiddo, N. *Linear programming in linear time when the dimension is fixed,* J. ACM 31 (1984), 114–127.
17. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E. J. Chem. Phys. 21 (1953), 1087.
18. Preparata, F.P., Hong, S.J. *Convex hulls of finite sets of points in two and three dimensions*, Comm. ACM 20 (1977), 87–93.
19. Raghavan, P. *Probabilistic construction of deterministic algorithms: approximating packing integer programs*, Proc. 27th Ann. IEEE Symp. on Foundat. of Comput. Sci. (1986), 10–18.
20. Seidel, R. *A convex hull algorithm optimal for point sets in even dimensions,* Univ. British Columbia, tech. Rep. 81–14, 1981.

21. Seidel, R. *Constructing higher-dimensional convex hulls at logarithmic cost per face,* Proc. 18th Ann. ACM Symp. Theory Comput. (1986), 404–413.

22. Seidel, R. *Linear programming and convex hulls made easy,* Proc. 6th Ann. ACM Symp. Comput. Geom. (1990), 211–215.

23. Spencer, J. *Ten lectures on the probabilistic method*, CBMS-NSF, SIAM, 1987.

24. Vapnik, V. N., Chervonenkis, A. Ya. *On the uniform convergence of relative frequencies of events to their probabilities,* Theory Probab. Appl. 16 (1971), 264–280.