PRODUCT RANGE SPACES, SENSITIVE SAMPLING, AND DERANDOMIZATION*

HERVÉ BRÖNNIMANN†¶, BERNARD CHAZELLE‡¶, AND JIŘÍ MATOUŠEK§

Abstract. We introduce the concept of a *sensitive ε-approximation* and use it to derive a more efficient algorithm for computing ε-nets. We define and investigate *product range spaces*, for which we establish sampling theorems analogous to the standard finite VC-dimensional case. This generalizes and simplifies results from previous works. Using these tools, we give a new deterministic algorithm for computing the convex hull of n points in $\mathbb{R}^d$. The algorithm is obtained by derandomization of a randomized incremental algorithm, and its running time of $O(n \log n + n^{\lfloor d/2 \rfloor})$ is optimal for any fixed dimension $d \geq 2$.

Key words. convex hull, deterministic optimal algorithm

AMS subject classifications. 52B55, 68Q20

1. Introduction.

During the last decade, randomized algorithms have been proposed as an efficient and elegant solution to several geometric problems [12, 13]. Derandomization aims at producing deterministic algorithms whose running times are within a constant factor of the running times of their randomized counterpart [9, 19, 23]. This process has successfully produced several geometric algorithms whose complexities are the best among those of the existing deterministic algorithms for the problems considered [5, 6, 7, 10]. For instance, the problem of computing the convex hull of n points in $\mathbb{R}^d$ is solved deterministically in [7] in time $O(n^{\lfloor d/2 \rfloor})$ for any $d \geq 4$, which is optimal. In section 3, we describe a new deterministic algorithm for computing the convex hull of n points in $\mathbb{R}^d$. Its running time of $O(n \log n + n^{\lfloor d/2 \rfloor})$ is optimal in any fixed dimension d. It uses a method similar to the one given in [7], but it is arguably simpler. Furthermore, there is no need to treat the two and three-dimensional cases separately, as is done in [7].

Deterministic constructions of ε-nets and ε-approximations [10, 18, 20, 21] play a key role in the derandomization of probabilistic geometric algorithms [5, 6, 7, 9, 10, 19], and they also do for our convex hull algorithm. A *range space* $\Sigma = (X, \mathcal{R})$ is a pair of a set X and a collection $\mathcal{R}$ of subsets of X [17]. An *ε-approximation* for Σ is a subset A of X such that

$$\left| \frac{|R|}{|X|} - \frac{|R \cap A|}{|A|} \right| \leq \varepsilon,$$

for every set R in $\mathcal{R}$. To be an *ε-net*, A need only intersect every set R in $\mathcal{R}$ whose size is greater than $\varepsilon|X|$. It is a classical result [17] that the existence of small subsets having these properties is linked to the finiteness of a parameter of the set system, called its *VC-dimension*. Namely, if the range space has finite VC-dimension

*A preliminary version of this paper appeared in Proc. 34. IEEE Symposium on Foundations of Computer Science (1993) pages 400–409.

†INRIA Sophia Antipolis, BP. 93, 2004 Route des Lucioles, 06902 Sophia Antipolis Cedex, France.

‡Department of Computer Science, Princeton University, Princeton, NJ 08544, USA. Supported by The Geometry Center, University of Minnesota, an STC funded by NSF, DOE, and Minnesota Technology, Inc.

§Department of Applied Mathematics, Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic. Supported by Humboldt Research Fellowship and by Czech Republic Grants GAUK 193,194 and GAČR 0194.

¶Supported by NSF Grant CCR-90-02352.

d, there exists an ε-net of size $O(d\varepsilon^{-1}\log(d\varepsilon^{-1}))$, and an ε-approximation of size $O(d\varepsilon^{-2}\log(d\varepsilon^{-1}))$ (see section 2).

In geometric applications, it is common to use the range space $(X, \mathcal{R})$ consisting of a set X of hyperplanes and the collection $\mathcal{R}$ of all the subsets of X consisting of the hyperplanes stabbed by a line segment. By definition, an ε-approximation allows to estimate how many hyperplanes separate any two points (with a level of accuracy depending on ε). It was shown in [6] that an ε-approximation can also be used to estimate how many vertices of the arrangement formed by X lie within a given simplex: this feature is essential in the recent work on point location [6], convex hull [7], and weak ε-nets for convex sets [8].

We generalize this idea by introducing the notion of a *product range space*. We discuss the problem of sampling such a space, and we explain the apparent paradox that product range spaces can be sampled even though they may have unbounded VC-dimension. We prove that the product of finite VC-dimensional range spaces can be sampled almost as efficiently as the original spaces, meaning that they admit ε-approximations and ε-nets of size polynomial in $1/\varepsilon$ and, most importantly, independent of the size of the range spaces. We specialize these sampling theorems to a geometric setting and we build tools for numerically integrating functions defined over the vertices of an arrangement of hyperplanes.

We also introduce the notion of *sensitive* sampling. Formally, we say that a subset $A \subseteq X$ is a *sensitive ε-approximation* for Σ if

$$\left| \frac{|R|}{|X|} - \frac{|R \cap A|}{|A|} \right| \le \frac{\varepsilon}{2}\Big(\sqrt{\frac{|R|}{|X|}} + \varepsilon\Big),$$

for every set R in $\mathcal{R}$. The bound on the right hand side may appear strange, but it arises naturally from what we expect of a random sample. Observe that a sensitive ε-approximation is at once both an ε-approximation and an ε^2-net. For a set system with finite VC-dimension d, we show the existence of a sensitive ε-approximation of size $O(d\varepsilon^{-2}\log(d\varepsilon^{-1}))$. We also modify an algorithm of [10] for computing ε-approximations so that it computes sensitive ε-approximations. If the underlying range space Σ has VC-dimension d then, under standard computational assumptions given in section 2, a sensitive $(1/r)$-approximation of size $O(dr^2\log(dr))$ can be computed in time $O(d)^{3d}r^{2d}\log^d(dr)|X|$. This gives an algorithm for computing a $(1/r)$-net of size $O(dr\log(dr))$ for $(X, \mathcal{R})$ in time $O(r^d\log^d r)|X|$ time, which significantly improves on the bound $O(r^{2d}\log^d r)|X|$ given in [10].

In a preliminary version of this paper, we had also claimed an $O(n\log^3 n)$-time algorithm for deterministically computing the diameter of n points in 3-space. This part contained an error (kindly pointed out to us by E. Ramos). In the meantime, a simpler deterministic algorithm with the same time complexity was found by Amato, Goodrich, and Ramos [3]. They use ideas akin to ours, but they use sampling over a different geometric range space $(X, \mathcal{R}')$ that contains ours: given a set X of hyperplanes, a range in $\mathcal{R}'$ consist of all the hyperplanes of X that intersect a given simplex of any dimension. We refer to their paper for a correct description of the algorithm. For further developments in the derandomization of geometric algorithms, we also refer to [4, 5].

2. Terminology and Sampling Theorems. In this section we consider general range spaces. First we review standard definitions and facts [17, 20]. A *range space* is a set system (or equivalently a hypergraph), whose elements are called *points* and sets are called *ranges*. Let $\Sigma = (X^*, \mathcal{R}^*)$ be a (possibly infinite) range space. If $Y \subseteq X^*$, we denote by $(Y, \mathcal{R}^*|_Y)$ the *subspace induced by* Y, where $\mathcal{R}^*|_Y = \{ R \cap Y : R \in \mathcal{R}^* \}$. A subset $Y \subseteq X^*$ is *shattered* (by $\mathcal{R}^*$) if $\mathcal{R}^*|_Y = 2^Y$. The maximum size of any shattered subset of X^* is called the *VC-dimension* of Σ; note that it can be infinite. We define the *shatter function* π_Σ of Σ as follows: $\pi_\Sigma(m)$ is the maximum possible number of sets in the subsystem of $(X^*, \mathcal{R}^*)$ induced by any m-point subset of X^*. It is well-known that the shatter function of a range space of VC-dimension d is bounded $\Phi(m,d) = \binom{m}{0} + \cdots + \binom{m}{d}$, which is less that $m^d + 1$ (see for instance [2]). Conversely, if the shatter function is bounded by a polynomial, then the VC-dimension is bounded by a constant.

In practice, we usually deal with finite subsystems of a range space. Let X be a finite subset of X^*, and let $\mathcal{R}$ be a shorthand for $\mathcal{R}^*|_X$; by abuse of terminology we still call the pair $(X, \mathcal{R})$ a range space. As we mentioned earlier, given any $0 < \varepsilon < 1$, a subset $A \subseteq X$ is called an ε*-approximation* for the range space $(X, \mathcal{R})$ if

$$\left| \frac{|R|}{|X|} - \frac{|R \cap A|}{|A|} \right| \leq \varepsilon$$

for each $R \in \mathcal{R}$. A subset $N \subseteq X$ is an ε*-net* for $(X, \mathcal{R})$ if $|R| > \varepsilon|X|$ implies that $R \cap N \neq \emptyset$. An ε-approximation is also an ε-net but the converse is false in general.

For instance, consider the range space $(X^*, \mathcal{R})$ mentioned in the introduction, where X^* is a set of hyperplanes in $\mathbb{R}^d$ and $\mathcal{R}^*$ is the collection of all the subsets of X^* consisting of the hyperplanes stabbed by a given line segment. Pick a finite subset X of m hyperplanes. Then it can be easily verified that the subsets of X^* consisting of the hyperplanes stabbed by two line segments are identical, if the endpoints of the segments can be paired up so that a pair lie in the same cell of the arrangement of X. There are $O(m^d)$ such cells, therefore the shatter function $\pi_\Sigma(m)$ of $(X^*, \mathcal{R})$ is bounded above by $O(m^{2d})$. This in turn implies that the range space has finite VC-dimension.

Efficient deterministic constructions of ε-nets and ε-approximations were given in [20] for the particular range space described above; see also [10] for slightly simpler proofs that are expressed in terms of general range spaces. Let d be the VC-dimension of Σ, or for that matter, any constant such that $\pi_\Sigma(m) = O(m^d)$. We assume that the range space admits a *subsystem oracle of dimension* d, meaning that, given any $Y \subseteq X$, all the sets of $\mathcal{R}|_Y$ can be computed explicitly in time $O(|Y|^{d+1})$. This assumption is justified in practice, as can be checked for instance on the range space given above: one may simply construct the arrangement of Y, choose a point inside each cell, and each pair of points yields a segment s and a range R_s of the hyperplanes stabbed by s; moreover, each range is enumerated exactly once in this process. Given any $r > 1$, one can in time $O(d)^{3d} r^{2d} \log^d(dr)|X|$ compute a $(1/r)$-approximation for $(X, \mathcal{R})$ of size $O(dr^2 \log(dr))$ and a $(1/r)$-net of size $O(dr \log(dr))$. These time bounds are linear in $|X|$ dif r is a constant.

2.1. Sensitive Approximations. Let $\Sigma = (X, \mathcal{R})$ be a range space of dimension bounded by a constant d. It is known that if one wants to get a $(1/r)$-approximation for Σ, it suffices to pick a random sample $A \subseteq X$ of size $O(r^2 \log r)$. Such a sample, however, has still better approximation properties if we are only interested in small ranges. The fact that A is, with high probability, a $(1/t)$-net for Σ with t being almost r^2 can be seen as a manifestation of this phenomenon. If we look at the dependence of the error with which a random sample approximates a range on the size of that range, we arrive at the following definition. A subset $A \subseteq X$ is a *sensitive ε-approximation* for Σ if

$$\left| \frac{|R|}{|X|} - \frac{|R \cap A|}{|A|} \right| \leq \frac{\varepsilon}{2}\Big(\sqrt{\frac{|R|}{|X|}} + \varepsilon\Big),$$

for every set $R \in \mathcal{R}$. In [5], it is shown that a random sample of size $O(dr^2 \log(dr))$ possesses the sensitive $(1/r)$-approximation property. It follows from the definition that a sensitive ε-approximation is an ε-approximation as well as an ε^2-net. By this observation the next result gives an immediate improvement over the $O(d)^{3d} r^{2d} \log^d(dr)|X|$-time construction of a $(1/r)$-net given in [20], while at the same time keeping the same size bound.

THEOREM 2.1. *Let $(X, \mathcal{R})$ be a range space with a subsystem oracle of dimension d. Given any $r > 1$, one can in time $O(d)^{3d} r^{2d} \log^d(dr)|X|$ compute a sensitive $(1/r)$-approximation for $(X, \mathcal{R})$ of size $O(dr^2 \log(dr))$.*

COROLLARY 2.2. *Let $(X, \mathcal{R})$ be a range space with a subsystem oracle of dimension d. Given any $r > 1$, one can in time $O(d)^{3d} r^{d} \log^d(dr)|X|$ compute a $(1/r)$-net for $(X, \mathcal{R})$ of size $O(dr \log(dr))$.*

Proof. Proof of Theorem 2.1:

We begin with a restriction of Theorem 2.1 to the case where ε is very small. We then adapt a recursive construction given in [10] to generalize this result and establish the theorem.

LEMMA 2.3. *Let $(X, \mathcal{R})$ be a range space of finite VC-dimension, where $|X| = n$ is even and large enough, and let $m = |\mathcal{R}|$. In $O(nm)$ time it is possible to compute a sensitive ε-approximation for $(X, \mathcal{R})$ of size $n/2$, for $\varepsilon = 15\sqrt{\ln(6m+6)/n}$.*

Proof. We select a random sample $A_1 \subseteq X$ of expected size $n/2$ by picking every element independently with probability $1/2$. Tail estimates show that A_1 (or its complement $X \setminus A_1$) has the sensitive ε-approximation property with high probability, to be made precise below. To obtain an approximation of size exactly $n/2$, we then show how to trim some elements from the bigger of A_1 and $X \setminus A_1$ while keeping the trimmed set a sensitive ε-approximation.

Given $0 < p < 1$, let $x_1, \ldots, x_n$ be independent random variables, each equal to $p - 1$ with probability p, respectively to p with probability $1 - p$. The following tail estimate can be found in [2]:

$$\text{Prob}\left[\left|\sum_{i=1}^{n} x_i\right| > \Delta\right] < 2e^{-2\Delta^2/n}.$$

Select a subset $A_0 \subseteq X$ by picking each element of X with probability p. Define an auxiliary function

$$\Delta(x) = \sqrt{x \ln(6m+6)/2}$$

and let $\mathcal{R}' = \mathcal{R} \cup \{X\}$. Given $R \in \mathcal{R}'$, the previous tail estimate indicates that, for our choice of Δ, the following holds with a probability greater than $1 - 1/(3m+3)$:

$$\Big|\, |R \cap A_0| - p|R| \,\Big| \leq \Delta(|R|). \tag{2.1}$$

Assume that A_0 satisfies (2.1) for each $R \in \mathcal{R}'$. We call such a subset *p-good* for $(X, \mathcal{R}')$: note that a random A_0 is p-good with probability at least $2/3$. Set $p = 1/2$ and let A_1 be the larger of the two sets A_0 and $X \setminus A_0$. Trivially, A_1 consists of at least $n/2$ elements and is p-good. As an effect of adding X as a range, A_1 has also little more than $n/2$ elements, namely at most $n/2 + \Delta(n)$.

We now want to remove some elements from A_1 so that it has exactly $n/2$ elements (this exact halving will be convenient in the forthcoming algorithm). If our goal were an ε-approximation only, we could remove an appropriate number of elements quite arbitrarily (as it is done in [20]). Removing arbitrary elements could, however, destroy the sensitive ε-approximation properties for small ranges, so we choose the elements to remove more carefully — we use a suitable random sample from A_1.

Let $q = 4\Delta(n)/n$; note that the finite VC-dimension hypothesis implies $m = n^{O(1)}$, so $q < 1$ for n large enough. With probability at least 2/3, a random sample $A_2 \subseteq A_1$ (with each element chosen independently with probability q) is a q-good subset for $(A_1, \mathcal{R}'|_{A_1})$. For such an A_2, since $A_1 = X \cap A_1$ is a range in $\mathcal{R}'|_{A_1}$, we have

$$|A_2| \geq q|A_1| - \Delta(|A_1|) \geq \frac{qn}{2} - \Delta(n) \geq \Delta(n) \geq |A_1| - \frac{n}{2},$$

and therefore we can pick $|A_1| - n/2$ elements in A_2 (any of them) and remove them from A_1, thus producing a subset $A \subseteq X$ of size $n/2$.

Note that our probabilistic construction of A can be derandomized in a straightforward fashion by using the method of conditional probabilities of Raghavan and Spencer [2, 23, 24]. With a little care, this can be accomplished in $O(nm)$ time: see [3, 20] for a similar construction.

It remains to show that A is a sensitive ε-approximation for $(X, \mathcal{R})$, for a choice of $\varepsilon = 15\sqrt{\log(6m+6)/n}$. We have

$$\left| |R \cap A| - \frac{|R|}{2} \right| \leq \left| |R \cap A| - |R \cap A_1| \right| + \left| |R \cap A_1| - \frac{|R|}{2} \right| \leq |R \cap A_2| + \Delta(|R|) \leq$$

$$q|A_1 \cap R| + 2\Delta(|R|) \leq \frac{q|R|}{2} + (q+2)\Delta(|R|)\,.$$

As $q < 1$ for n large enough and $q|R|/2 = 2|R|\Delta(n)/n \leq 2\Delta(|R|)$, we obtain $\Big| |R \cap A| - |R|/2 \Big| \leq 5\Delta(|R|)$, and hence,

$$\left| \frac{|R \cap A|}{|A|} - \frac{|R|}{n} \right| \leq \frac{2}{n} 5\Delta(|R|) = \frac{10}{n}\sqrt{|R| \ln(6m+6)/2} \leq \frac{\varepsilon}{2}\sqrt{\frac{|R|}{n}},$$

which proves lemma 2.3. □

Sensitive approximations can be *refined* (Lemma 2.4) and *composed* (Lemma 2.5) in a fashion similar to standard ε-approximations [10].

LEMMA 2.4. *If A is a sensitive ε-approximation for a range space $(X, \mathcal{R})$ and B is a sensitive δ-approximation for $(A, \mathcal{R}|_A)$, then B is a sensitive $(\varepsilon + 2\delta)$-approximation for $(X, \mathcal{R})$.*

Proof. Consider a range $R \in \mathcal{R}$. For short, we write $\rho_X = |R|/|X|$, $\rho_A = |R \cap A|/|B|$, $\rho_B = |R \cap B|/|B|$. We have

$$|\rho_X - \rho_B| \leq |\rho_X - \rho_A| + |\rho_A - \rho_B| \leq \frac{\varepsilon}{2}(\sqrt{\rho_X} + \varepsilon) + \frac{\delta}{2}(\sqrt{\rho_A} + \delta)$$

by the definition of a sensitive ε-approximation. We estimate

$$\sqrt{\rho_A} \le \sqrt{\rho_X + \frac{\varepsilon}{2}\left(\sqrt{\rho_X} + \varepsilon\right)} \le \sqrt{\rho_X} + \sqrt{\frac{\varepsilon}{2}\left(\sqrt{\rho_X} + \varepsilon\right)} \le \sqrt{\rho_X} + \frac{\varepsilon/2 + \sqrt{\rho_X} + \varepsilon}{2},$$

where we used the AG-mean inequality $\sqrt{ab} \le (a+b)/2$ in the last step. With this estimate, we calculate

$$|\rho_X - \rho_B| \le \frac{\varepsilon}{2}\left(\sqrt{\rho_X} + \varepsilon\right) + \frac{\delta}{2}\left(\frac{3}{2}\sqrt{\rho_X} + \frac{3}{4}\varepsilon + \delta\right) \le \frac{\varepsilon + 2\delta}{2}\left(\sqrt{\rho_X} + \varepsilon + 2\delta\right).$$

Since this is true for any $R \in \mathcal{R}$, the proof is complete. $\square$

LEMMA 2.5. *Let $(X, \mathcal{R})$ be a range space and let $\{X_i\}_{1\le i\le m}$ be a partition of X into m equal-size subsets. If A_i is a sensitive ε-approximation for $(X_i, \mathcal{R}|_{X_i})$ and all the A_i's have the same size, then $A = \cup_i A_i$ is a sensitive ε-approximation for $(X, \mathcal{R})$.*

Proof. Consider any range $R \in \mathcal{R}$. For short, we put $\rho_{X_i} = |R \cap X_i|/|X_i|$, $\rho_{A_i} = |R \cap X_i|/|X_i|$, $\rho_X = |R|/|X|$, and $\rho_A = |R \cap A|/|A|$. Since the X_i's are disjoint, we have $\rho_X = \frac{1}{m}\sum_{i=1}^m \rho_{X_i}$ and $\rho_A = \frac{1}{m}\sum_{i=1}^m \rho_{A_i}$, where the factor $\frac{1}{m}$ accounts for the difference in the denominators of ρ_X and the ρ_{X_i}'s. Therefore,

$$|\rho_X - \rho_A| \le \frac{1}{m}\sum_{i=1}^m |\rho_{X_i} - \rho_{A_i}| \le \frac{\varepsilon}{2}\left(\frac{1}{m}\sum_{i=1}^m \sqrt{\rho_{X_i}} + \varepsilon\right) \le \frac{\varepsilon}{2}\left(\sqrt{\rho_X} + \varepsilon\right),$$

where the last inequality follows by the concavity of the square root function. $\square$

We are now ready to build a sensitive $(1/r)$-approximation, for any value of r. The algorithm is almost identical to the construction of nonsensitive ε-approximations given in [10]. We begin with a simplifying observation. If $n = |X|$ is not a power of two, let us pad X by adding up to $n-1$ artificial points so as obtain a power of two. This gives us a new range space $\Sigma' = (X \cup X_0, \mathcal{R} \cup \{X_0\})$; note that the set X_0 of artificial points is added as a range. Let A be a sensitive $(\varepsilon/6)$-approximation for this new range space. It not hard to show that $B = X \cap A$ is a sensitive ε-approximation for $(X, \mathcal{R})$. Here are the details: Given any $R \in \mathcal{R}$, we have

$$\begin{aligned}\left|\frac{|R|}{|X|} - \frac{|R\cap B|}{|B|}\right| &= \frac{|X \cup X_0|}{|X|}\left|\frac{|R|}{|X\cup X_0|} - \frac{|R\cap B|}{|B|}\cdot\frac{|X|}{|X\cup X_0|}\right| \\ &\le 2\left(\left|\frac{|R|}{|X\cup X_0|} - \frac{|R\cap B|}{|A|}\right| + \frac{|R\cap B|}{|B|}\left|\frac{|X|}{|X\cup X_0|} - \frac{|B|}{|A|}\right|\right). \end{aligned} \tag{2.2}$$

Using the sensitive $(\varepsilon/6)$-approximation property of A, we get that the difference in the first absolute value is at most $\frac{\varepsilon}{12}\left(\sqrt{|R|/|X|} + \varepsilon/6\right) \le \varepsilon/6$, and so is the difference in the second absolute value. Substituting this and the trivial estimate $|R\cap B|/|B| \le 1$ into (2.2), we obtain

$$\frac{|R\cap B|}{|B|} \le \frac{|R|}{|X|} + \frac{2\varepsilon}{3}.$$

Substituting this improved upper bound into (2.2) then yields

$$\left|\frac{|R|}{|X|} - \frac{|R\cap B|}{|B|}\right| \le \frac{\varepsilon}{2}\left(\sqrt{\frac{|R|}{|X|}} + \varepsilon\right),$$

which shows that B is a sensitive ε-approximation. This allows us to assume that n is now a power of two.

We begin with one piece of terminology. Applying Lemma 2.3 to an even-sized subset $Y \subseteq X$ is called *halving* Y. Note that this results in a sensitive $h(|Y|)$-approximation, where

$$h(t) = 15\sqrt{\log(6t^d + 6)/t}.$$

It is easy to see that if $t = \Omega(d \log d)$, we have $h(t) < 1$ and $h(2t) \leq \frac{3}{4}h(t)$. Moreover, it is also not hard to show that $h(dr^2 \log(dr)) = O(1/r)$ for any $r > 1$. We are now ready to describe the algorithm for constructing a sensitive $(1/r)$-approximation for $(X, \mathcal{R})$.

To begin with, we divide up the set X into subsets of size 2^k (for some appropriate parameter k), and we associate each subset with the leaves of a complete binary tree. Next, we process the tree bottom-up level by level. At each internal node, we merge together the two sets associated with its children and, if the level of the node is not divisible by $d + 2$ (leaves being at level 0), we halve the union. The resulting set is *associated* with the node in question. Once the tree is completely processed, i.e., the set associated with its root has been computed, we say that the first phase is over, and we move on to the second phase. We take the set associated with the root and we keep halving it until its size is equal to $c_1 dr^2 \log(dr)$, for some appropriate constant $c_1 > 0$.

The union of all the sets associated with the nodes at level i constitutes a sensitive ε_i-approximation for some ε_i, which we call the *error* at level i. During the first $d+1$ levels, each individual set remains of size 2^k (since halving and merging alternate). Note that halving is applied to sets of size 2^{k+1}. By Lemmas 2.4 and 2.5, the total error after the first $d + 1$ levels is $2(d + 1)h(2^{k+1})$. At level $d + 2$, no additional error is incurred since we skip the halving step. The next $d + 1$ steps are similar to the first batch of $d + 1$, except that the size of the individual sets has now doubled: thus, the total error incurred up to level $2d + 3$ is $2(d + 1)\left(h(2^{k+1}) + h(2^{k+2})\right)$. From level to level, the error follows a geometrically decreasing series, so the total error incurred at the end of the first phase is $O(d) \times h(2^k)$. If we choose $2^k = c_2 d^3 r^2 \log(dr)$, for some constant c_2 large enough, this makes the error at most $1/2r$. In the second phase, the error is still bounded by a geometrically increasing series whose last term is $O(h(c_1 dr^2 \log(dr)))$, meaning that the additional error contributed by the second phase is $O(h(c_1 dr^2 \log(dr)))$. Choosing c_1 large enough keeps this error under $1/2r$. Combining the two phases shows that the final set, which is of the desired size $O(d)r^2 \log(dr)$, is a sensitive $(1/r)$-approximation for $(X, \mathcal{R})$.

What is the running time of the algorithm? In the first halving step, we apply Lemma 2.3 to $|X|/2^k$ sets of size $2^k = O(d^3)r^2 \log(dr)$ each; the total time needed for these operations is $O(d)^{3d} r^{2d} \log^d(dr)|X|$. In the next $d + 1$ halving steps, the sets remain of the same size but their numbers decrease geometrically. Thus, the cost of processing the first level is dominant. At level $d + 2$, the size of each individual set doubles, which increases the cost of applying Lemma 2.3 by a factor of 2^{d+1}. But the $(d + 2)$ previous merging steps in the previous round have reduced the total number of sets by 2^{d+2}, so the running time of the following round is at most half of the time for previous round. Similarly, in the second phase, the running time follows a geometrically decreasing sequence at each step. Thus, the total running time of the algorithm is $O(d)^{3d} r^{2d} \log^d(dr)|X|$, and the proof of Theorem 2.1 is complete. $\square$

2.2. Product Range Spaces. Let $\Sigma_1 = (X, \mathcal{R})$ and $\Sigma_2 = (Y, \mathcal{S})$ be (finite) range spaces. We define the product range space $\Sigma_1 \otimes \Sigma_2$ to be $(X \times Y, \mathcal{T})$, where $\mathcal{T}$ consists of all subsets $T \subseteq X \times Y$ such that all the *cross-sections* $S_x = \{\, y \in Y \,:\, (x,y) \in T \}$ are sets of $\mathcal{S}$, and similarly, all $R_y = \{\, x \in X \,:\, (x,y) \in T \}$ are sets of $\mathcal{R}$.

To illustrate this definition, consider the bichromatic arrangement of n red and n blue lines in $\mathbb{R}^2$. Ranges of the blue (resp. red) space consist of blue (resp. red) lines that intersect a given line segment. The product $\Sigma_1 \otimes \Sigma_2$ of the blue space by the red space is a range space $(Z, \mathcal{T})$, where Z consists of all the bichromatic intersections. A range is a subset T of Z such that the intersections in T that are incident upon any given line appear consecutively (among those of Z). For example, the bichromatic intersections that fall inside any convex set constitute a range. This suggests that the product of finite VC-dimensional spaces might not be itself of finite VC-dimension. Indeed, this can best be seen by observing that in our example, any bichromatic pairing of the lines gives a collection of n bichromatic intersections, and that *any* of its 2^n subsets is a valid range!

Theorem 2.6. *Let $\Sigma_1 = (X, \mathcal{R})$ and $\Sigma_2 = (Y, \mathcal{S})$ be two range spaces. If A (resp. B) is a δ-approximation (resp. ε-approximation) for Σ_1 (resp. Σ_2), for $0 \leq \delta, \varepsilon, \leq 1$, then $A \times B$ is a $(\delta + \varepsilon)$-approximation of $\Sigma_1 \otimes \Sigma_2$.* It is worth observing that even though an ε-approximation of a product space is of size $O(\varepsilon^{-4} \log^2 \varepsilon^{-1})$, its representation as a set product has size only $O(\varepsilon^{-2} \log \varepsilon^{-1})$.

A range space of infinite VC-dimension has, for infinitely many n, a shattered subset A of size n, and clearly an ε-approximation for the subspace induced by such A must be of size at least $(1-\varepsilon)n$. This might seem to contradict Theorem 2.6. To explain this apparent paradox, we must observe that in general a subspace of a product range space is not itself a product range space. In particular, even though the ground set contains very large shattered subsets, the subsystems induced by these subsets are not product range spaces: therefore, the fact that they cannot be sampled has no bearing on Theorem 2.6. In fact, the proper definition of a subspace in the context of product range spaces would be the product of subspaces of standard range spaces.

Proof. Proof of Theorem 2.6:

Given two range spaces of finite VC-dimension, $\Sigma_1 = (X, \mathcal{R})$ and $\Sigma_2 = (Y, \mathcal{S})$, recall that the product $\Sigma_1 \otimes \Sigma_2$ is defined as $(Z, \mathcal{T})$, where $Z = X \times Y$ and $\mathcal{T}$ consists all the subsets $T \subseteq Z$ such that for any $x \in X$ and $y \in Y$, the sets T^x and T_y are ranges of $\mathcal{S}$ and $\mathcal{R}$, respectively, where

$$T^x = \{\, y \,:\, (x,y) \in \mathcal{T} \,\},$$

$$T_y = \{\, x \,:\, (x,y) \in \mathcal{T} \,\}.$$

As we observed, $\Sigma_1 \otimes \Sigma_2$ usually does not have finite VC-dimension. For example, if Σ_1 and Σ_2 are the (infinite) range spaces defined by two secant lines and their intervals, the product space ranges include all the convex regions of the plane.

It is helpful to use a slightly different formulation of an ε-approximation. Let Prob_X be a probability distribution on X, and $\mathrm{Prob}_X[\, R \,|\, A \,]$ be the conditional probability that a random element in X belongs to R, given that it is in A. Thus, for A to be a δ-approximation for Σ_1, it is equivalent to say that, for every $R \in \mathcal{R}$,

$$|\, \mathrm{Prob}_X[\, R \,|\, A \,] - \mathrm{Prob}_X[R] \,| \leq \delta \,.$$

A simple technical observation will greatly simplify our discussion below. In essence, it is nothing else than Fubini's theorem and asserts than we can sum the probabilities first on x then on y, or first on y then on x, and obtain the same result. To put it in mathematical notation, given any $A \subseteq X$, $B \subseteq Y$, and $T \in \mathcal{T}$, we have

$$\mathrm{Prob}_Z\,[\,T\,|\,A \times B\,] = \begin{cases} \mathbf{E}_X\,[\,\mathrm{Prob}_Y\,[\,T^x\,|\,B\,]\,|\,A\,] \\ \mathbf{E}_Y\,[\,\mathrm{Prob}_X\,[\,T_y\,|\,A\,]\,|\,B\,] \end{cases}$$

where $\mathbf{E}_X[\cdot|A]$ denotes the expectation for a random element of X given that the element belongs to A, and $\mathbf{E}_Y[\cdot|B]$ the analogous conditional expectation on Y. To prove Theorem 2.6, we apply this observation twice. Recall that A (resp. B) is a δ-approximation (resp. ε-approximation) of Σ_1 (resp. Σ_2).

$$\begin{aligned} \mathrm{Prob}_Z\,[\,T\,|\,A \times B\,] &= \mathbf{E}_X\,[\,\mathrm{Prob}_Y\,[\,T^x\,|\,B\,]\,|\,A\,] \\ &= \mathbf{E}_X\,[\,\mathrm{Prob}_Y\,[T^x]\,|\,A\,] + \varepsilon' \\ &= \mathbf{E}_Y\,[\,\mathrm{Prob}_X[\,T_y\,|\,A\,]\,] + \varepsilon' \\ &= \mathbf{E}_Y\,[\,\mathrm{Prob}_X\,[T_y]\,] + \delta' + \varepsilon' \\ &= \mathrm{Prob}_Z\,[T] + \delta' + \varepsilon', \end{aligned}$$

where $|\varepsilon'| \leq \varepsilon$, $|\delta'| \leq \delta$, which completes the proof of Theorem 2.6. $\square$

A similar result exists for sensitive approximations but the formulas are a little more complicated. It is easy to show, however, that the product of a sensitive δ-approximation with a sensitive ε-approximation is a sensitive $\sqrt{2}(\delta + \varepsilon)$-approximation [5].

Finally, we should note that the product described here is associative. We can thus take the d-fold product $\Sigma \otimes \cdots \otimes \Sigma$ of a range space $\Sigma = (X, \mathcal{R})$. Theorem 2.6 may be extended straightforwardly.

COROLLARY 2.7. *If A is an ε-approximation for a range space Σ, then the d-fold Cartesian product A^d is a $(d\varepsilon)$-approximation of the d-fold product $\Sigma \otimes \cdots \otimes \Sigma$.* For sensitive approximations, the theorem can be extended similarly. It is easy to show that the d-fold product of a sensitive ε-approximation is a sensitive $(d^2\varepsilon)$-approximation [5].

Let us show for instance how to use this range space product to estimate the number of vertices of an arrangement of hyperplanes inside a convex region. The range space $\Sigma = (X, \mathcal{R})$ under consideration here is the one described above: given a set H of hyperplanes in $\mathbb{R}^d$, $\mathcal{R}$ is the collection of all the subsets of H consisting of the hyperplanes stabbed by a given line segment. We let Σ^d be the d-fold product of Σ. Of particular interest is the subset $H^{(d)}$ of H^d consisting of the d-tuples of hyperplanes of H in general position: such d-tuples intersect in a unique point of $\mathbb{R}^d$ which is a vertex of the arrangement of H. Let us denote by $V(H)$ the set of these vertices. For a convex region σ (not necessary full-dimensional), consider the arrangement of the intersections of hyperplanes of H with the affine hull of σ, and let $V(H, \sigma)$ be the set of vertices of this arrangement lying inside σ.

THEOREM 2.8. *Let H be a set of hyperplanes in general position, and A be an ε-approximation for $\Sigma = (H, \mathcal{R})$. Then, for any convex region σ of dimension j in $\mathbb{R}^d$,*

$$\left| \frac{|V(H,\sigma)|}{|H|^j} - \frac{|V(A,\sigma)|}{|A|^j} \right| \leq \varepsilon.$$

Proof. This theorem was already shown in [6] for the particular case of simplices. The proof in terms of range space products is particularly simple. We may assume that σ is d-dimensional, otherwise the result may be proved by considering the j-fold product of Σ.

Let $(h_1, \ldots, h_d)$ be a d-tuple in $H^{(d)}$, and let $f(h_1, \ldots, h_d) = h_1 \cap \ldots \cap h_d$ be the corresponding vertex of $V(H)$. Note that because the hyperplanes are in general position, f is a $d!$ to one map.

Given any convex region σ, the inverse image $T = f^{-1}(V(H, \sigma))$ is a range in Σ^d. Indeed, it suffices to prove that its sections $T^i_{(h_1, \ldots, h_d)}$ consisting of the hyperplanes h such that $(h_1, \ldots, h_{i-1}, h, h_{i+1}, \ldots, h_d)$ is in T, are exactly the hyperplanes stabbed by some segment s in $\mathbb{R}^d$. Note that $\cap_{j \neq i} h_j$ is a line in $\mathbb{R}^d$, and that it intersects σ along such a segment s. Moreover, a d-tuple $(h_1, \ldots, h_{i-1}, h, h_{i+1}, \ldots, h_d)$ is in T if and only if h is stabbed by the line segment s.

Finally, we note that the size of $T = f^{-1}(V(H, \sigma))$ is $d!$ times the number of vertices of $V(H, \sigma)$. Using corollary 2.7, we conclude that

$$\left| \frac{|V(H,\sigma)|}{|H|^d} - \frac{|V(A,\sigma)|}{|A|^d} \right| \leq \frac{d\varepsilon}{d!} \leq \varepsilon. \qquad \square$$

A similar result can be proved for sensitive approximations, with slightly worse approximation bounds. For instance, it is proved in [5] that

$$\left| \frac{|V(H,\sigma)|}{|H|^j} - \frac{|V(A,\sigma)|}{|A|^j} \right| \leq \frac{4\varepsilon}{2} \left(\sqrt{\frac{|V(H,\sigma)|}{|H|^j}} + 4\varepsilon \right).$$

for a set H of hyperplanes in general position, an ε-approximation A for $\Sigma = (H, \mathcal{R})$, and any convex region σ of dimension j in $\mathbb{R}^d$.

3. Computing Convex Hulls. We describe a new deterministic algorithm for computing the convex hull of n points. Its running time of $O(n \log n + n^{\lfloor d/2 \rfloor})$ is optimal in any fixed dimension d. Our strategy is similar to the derandomization scheme used in [7]. In particular, it is still built around Raghavan and Spencer's method of conditional probabilities [2, 23, 24]. The main difference is in the underlying probabilistic model and the maintenance of approximation tools. The result is an algorithm that is arguably simpler.

The convex hull problem is reducible, by duality, to computing the intersection of n halfspaces. This problem can be solved in optimal expected time by a randomized incremental algorithm [13]: the halfspaces are inserted in random order, and the current intersection is maintained after each insertion.

We aim at derandomizing such an algorithm. For technical reasons, we use a slightly different randomized algorithm as a basis, to be described below.

3.1. Notation and preliminaries. Let H be a fixed collection of n hyperplanes in $\mathbb{R}^d$, and let O (the *origin*) be a given point not lying on any hyperplane of H. Using simulation of simplicity if necessary [16], we may assume that the hyperplanes of H are in general position. We also may enclose $\mathbb{R}^d$ in a box which contains all the vertices of the arrangement of H. The set H_0 of hyperplanes bounding this box is added to H. In the sequel, when we let R be a subset of H, we assume that R is a subset of H that contains H_0. This ensures that we always deal with polytopes and not polyhedra, and thus clears the issue of unboundedness.

Let R be a subset of H. We let $R^\cap$ denote the closure of the cell enclosing O of the arrangement of R. In the sequel, the word 'simplex' always means a *relatively open* simplex. Similarly, the word 'face' always means a *relatively open* face of a polytope or of an arrangement.

Given a simplex s, let $R_{|s}$ denote the subset of hyperplanes of R that intersect s but do not contain it.

Given a polytope P (such as $R^\cap$), we let $V(P)$ denote the set of vertices of P. If R is a set of hyperplanes, we let $V(R)$ be the set of vertices of the arrangement of R. For a simplex s (not necessary full-dimensional), consider the arrangement of the intersections of hyperplanes of R with the affine hull of s, and let $V(R,s)$ be the set of vertices of this arrangement lying inside s.

For a vertex $v \in V(H)$, we define the *conflict list* of v, denoted $H_{|Ov}$, to be the set of hyperplanes separating v from the origin. The *level of* v, denoted by n_v, is the size of $H_{|Ov}$. Similarly we define the *conflict list* for a simplex s, denoted by $H_{|Os}$, as the set of hyperplanes of H intersecting the relative interior of the convex hull of $s \cup \{O\}$. We note that the conflict list of a simplex is the union of the conflict lists of its vertices. We set $n_s = |H_{|Os}|$ (note that but n_s and $|H_{|s}|$ may be different for simplices s on the boundary of $R^\cap$).

In our algorithm, we use a special kind of triangulation for $R^\cap$, called the *geode of* R and denoted by $\mathcal{G}(R)$. Formal definitions and important properties are given in [7]. The geode consists of a triangulation of the boundary of $R^\cap$ along with a central lifting of that triangulation towards the origin O. The triangulation is defined recursively and is similar to the so-called *bottom vertex triangulation* [11]. To triangulate a face f of dimension k of $\partial R^\cap$, we first triangulate its faces of dimension $\leq k-1$ and then we lift these triangulations to the *apex* of f, where the apex is the vertex contained in this face with the smallest conflict list (ties being broken using some systematic rule, such as taking the vertex with lexicographically smallest coordinate vector). The resulting geode contains $O(n^{\lfloor d/2 \rfloor})$ simplices, and the choice of the apex leads to the following property.

Lemma 3.1 ([7]). *For any integer c, and any $R \subseteq H$, there is a constant $b = b(c)$ such that for any $c' \leq c$, we have*

$$\sum_{s \in \mathcal{G}(R)} n_s^{c'} \leq b \sum_{v \in V(R^\cap)} n_v^{c'} \tag{3.1}$$

Proof. The proof is the same as in [7], but the result there is stated for $H_{|s}$ instead of n_s (even though the proof itself is actually stated in terms of n_s). We recall it for completeness.

We let $\bar{f}$ denote the closure of a face f of $R^\cap$. We prove by induction that, for any k-face of $R^\cap$, the sum $\sum_{s \in \mathcal{G}(R) \cap \bar{f}} n_s^c$, denoted A_f, is at most

$$k!(2^c+1)^k \sum_{v \in V(R^\cap) \cap \bar{f}} n_v^c \, .$$

The lemma follows from the case $k = d$, where f is the interior of $R^\cap$.

The case $k = 0$ is immediate, as the 0-faces of $R^\cap$ are precisely its vertices. Assume the induction hypothesis is true for some $k - 1 < d$. We observe that, by choosing the apex of f as the lifting vertex for the geode (or the origin, if $k = d$), $A_f \leq (2^c + 1) \sum_g A_g$, where g ranges over all the $(k-1)$-faces of $R^\cap$ 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 lifting g toward the apex w (or $w = O$ if $k = d$; note that $n_w \leq n_s$ for any s contained in the closure of g, by definition of w). By our general position assumption, a vertex cannot belong to more than k $(k-1)$-faces, so we cannot count it more than k times in the above sum ranging on g. Substituting for A_g with the induction hypothesis gives the result. □

In our algorithm and analysis, various constants will appear (dependent on d, as a rule). To avoid complicated implicit dependencies between them, we express most constants as functions of two basic parameters C and c. For all estimates to work, one first chooses c as a sufficiently large constant, and then C as a still much larger constant. The $O()$ notation in the proofs may hide constants dependent on c (and d), but *not on* C; where the hidden constant does depend on C, we use the $O_C()$ notation.

We need an estimate for the higher moments of the binomial distribution:

LEMMA 3.2. *Let $X = X_{[1,n]} = X_1 + X_2 + \ldots + X_n$, where the X_i are independent random variables, each attaining value* 1 *with probability p and value 0 with probability $1-p$. Then for any natural number c, $\mathbf{E}[X^c] \leq (c+np)^c$.* This must be part of the folklore, but since we haven't discovered an explicit reference, we include a short proof.

Proof. We prove more generally that $\mathbf{E}[(X+a)^c] \leq (c+np+a)^c$, for all natural numbers n, c, and a. The inequality is clear if $n=0$, or $c=0,1$. Assuming this is true for some c and for all n and a, we derive by induction that

$$\begin{aligned}
\mathbf{E}[(X+a)^{c+1}] &= \sum_{i=1}^{n} p\mathbf{E}[(X+a)^c | X_i = 1] + a\mathbf{E}[(X+a)^c] \\
&= np\mathbf{E}[(X_{[1,n-1]} + a + 1)^c] + a\mathbf{E}[(X+a)^c] \\
&\leq np(c + (n-1)p + a + 1)^c + a(c + np + a)^c \\
&\leq (c + 1 + np + a)^{c+1}. \qquad \square
\end{aligned}$$

3.2. The underlying randomized algorithm. In our underlying randomized algorithm, hyperplanes are inserted in rounds. In the first round, a suitable constant number c of hyperplanes are chosen (arbitrarily, not necessarily at random) and inserted. Suppose that after the $(j-1)$st round, a set $R \subseteq H$ has been inserted, $|R| = r$. We assume a suitable representation of $\mathcal{G}(R)$, the geode of R. We also keep the conflict list of every simplex $s \in \mathcal{G}(R)$.

In the jth round, we fix a probability

$$p = \frac{2}{3}\,\frac{r}{n-r}$$

and we choose a random sample S from $H \setminus R$ by picking each hyperplane of $H \setminus R$ into S randomly and independently with probability p. For each simplex $s \in \mathcal{G}(R)$, we compute the portion of the arrangement of S lying within s, then we isolate the portion of $(R \cup S)^\cap$ within s from it, and we glue these pieces together, obtaining the facial lattice of $(R \cup S)^\cap$. Using the conflict lists of the vertices, we finally compute the geode $\mathcal{G}(R \cup S)$ and the conflict lists of its simplices.

The expected number of hyperplanes in S is $\frac{2}{3}r$, thus the size of R increases geometrically between rounds and the expected number of rounds is $O(\log n)$. When the number of hyperplanes in R exceeds n/c, we insert all the remaining hyperplanes of $H \setminus R$ (in a manner similar to adding a new sample S) and finish.

The work in the jth round of this algorithm is at most proportional to

$$\sum_{s\in\mathcal{G}(R)} \left(|S_{|Os}|^d + n_s\right) \tag{3.2}$$

(see also [7] for a more detailed description of the required computations and of their time complexity). Intuitively, we should expect the sizes n_s of the conflict lists to be about n/r and each $S_{|Os}$ to have about constant size. This is not quite true of *all* simplices in the randomized algorithm (and even less so in the derandomized version). However, as was observed by Clarkson in a somewhat different context [12], the averages of $|S_{|Os}|^c$ and of $\left(n_s \frac{r}{n}\right)^c$ over all simplices of $\mathcal{G}(R)$ are expected to be bounded by a constant in the randomized algorithm, for any constant c, and this is what we will also aim at in the derandomized version. From this point of view, we might appropriately call the quantities $|S_{|Os}|$ and $n_s \frac{r}{n}$ *quasi-constant*. To simplify the notation, we introduce the symbols

$$q_s = n_s \frac{r}{n} + 1\,,$$
$$r_s = |S_{|Os}| + 1\,.$$

The basic property of a quasi-constant quantity x_s is that its moment of order c is $\sum_{s\in\mathcal{G}(R)} x_s^c = O_C(1) r^{\lfloor d/2\rfloor}$, for a constant C that will be determined later. Note that this also implies the same property for the moments of order $c' \le c$, by a routine application of Hölder's inequality.

The rest of this section is devoted to proving that the above quantities q_s and r_s are expected to be quasi-constant.

Following [7], we say that the geode of R is a *semicutting* if

$$\sum_{v\in V(R^\cap)} n_v^c \le N \stackrel{\text{def}}{=} C r^{\lfloor d/2\rfloor} \left(\frac{n}{r}\right)^c .$$

Again, a routine application of Hölder's inequality shows that

$$\sum_{v\in V(R^\cap)} n_v^{c'} \le C r^{\lfloor d/2\rfloor} \left(\frac{n}{r}\right)^{c'}$$

for any $0 \le c' \le c$. Note that the definition of semicutting involves only the conflict lists of the vertices and not of the simplices of the geode. This has no bearing on the analysis, however; recall from Lemma 3.1 that the particular triangulation of $R^\cap$ we use, the geode, is chosen in such a way that we also have

$$\sum_{s\in\mathcal{G}(R)} n_s^{c'} \le b \sum_{v\in V(R^\cap)} n_v^{c'}$$

for any $0 \le c' \le c$ and for some constant $b = b(c')$. Note that this implies that $\sum_{s\in\mathcal{G}(R)} q_s^{c'} = O_C(1) r^{\lfloor d/2\rfloor}$, and hence that q_s is quasi-constant.

Inductively, we assume that the geode of R built in the previous rounds of the algorithm is a semicutting. For the sample S, we postulate the following conditions:

C1. $r/2 \le |S| \le r$.
C2. $\sum_{s\in\mathcal{G}(R)} r_s^c \le C^2 r^{\lfloor d/2\rfloor}$.
C3. The geode of $R \cup S$ is a semicutting.

As we will see shortly, the randomized algorithm yields these properties with high probability[1].

Corresponding to these properties, we introduce three functions measuring the quality of the sample S. We put

$$F_1(S) = \frac{1}{4r}\left(|S| - \frac{2}{3}r\right)^2,$$
$$F_2(S) = \frac{1}{C^2 r^{\lfloor d/2 \rfloor}} \sum_{s \in \mathcal{G}(R)} r_s^c,$$
$$F_3(S) = \frac{1}{N} \sum_{v \in V((R \cup S)^\cap)} n_v^c .$$

Further we define the quantities

$$\mathcal{E}_j = \mathbf{E} F_j(S)$$

for $j = 1, 2, 3$, where the expectation is taken with respect to a random choice of S (it implicitly depends on R, which we consider fixed). We put $\mathcal{E} = \mathcal{E}_1 + \mathcal{E}_2 + \mathcal{E}_3$. The quantities $\mathcal{E}$ and $\mathcal{E}_j$ will be referred to as *energy* (for reasons more apparent later). We now bound $\mathcal{E}$.

LEMMA 3.3. *We have $\mathcal{E}_1 \le 1/6$.*

Proof. It is immediate that $\mathcal{E}_1 = \dfrac{\mathbf{var}\,|S|}{4r} = \left(\dfrac{n-r}{4r}\right) p(1-p) \le \dfrac{1}{6}$. $\square$

LEMMA 3.4. *If the geode of R is a semicutting, then $\mathcal{E}_2 = O(1/C)$.*

Proof. Consider a simplex $s \in \mathcal{G}(R)$. The contribution of every hyperplane $h \in H_{|Os}$ to $|S_{|Os}|$ is a $0/1$ random variable attaining value 1 with probability p, and so by Lemma 3.2 we have $\mathbf{E}\,|S_{|Os}|^c \le (c + p|H_{|Os}|)^c \le (c + pn_s)^c = O(q_s^c)$. Summing over all the simplices $s \in \mathcal{G}(R)$ shows that the expectation of $\sum_{s \in \mathcal{G}(R)} r_s^c = O(\sum_{s \in \mathcal{G}(R)} q_s^c)$. Using (3.1), the latter expression is $O(Cr^{\lfloor d/2 \rfloor})$, which proves the lemma. $\square$

The next lemma concerns $\mathcal{E}_3$. Unlike the previous lemma, it does not assume that the geode of R is a semicutting, thus, no matter how 'bad' R might be, a random S guarantees that the geode of $R \cup S$ is a semicutting (with high probability). This robustness property will be crucial: in the derandomized version, the computed R won't presumably be as good as a true random sample would be, but the error will not propagate between rounds, as each new round alone would suffice to produce a semicutting for *any* R provided that S is random or imitates a random sample well enough.

LEMMA 3.5. *Let $R \subset H$ be arbitrary, $c \le |R| = r \le n/c$, and let S be a random sample from $H \setminus R$ obtained by choosing each hyperplane independently with probability p. Then*

$$\mathcal{E}_3 = \frac{1}{N} \sum_{v \in V(H) \cap R^\cap} p^{d_v} (1-p)^{n_v} n_v^c = O(1/C)\,, \tag{3.3}$$

where d_v denotes the number of hyperplanes of $H \setminus R$ passing thru v.

[1]The reader might wonder why we look at high moments when the complexity of the randomized algorithm only involves n_s and the dth power of r_s. The reason is that in the derandomization, we need auxiliary computations whose complexity is a larger polynomial in q_s and r_s.

Proof. For a vertex $v \in V(H) \cap R^\cap$, the probability of appearing as a vertex in $V((R \cup S)^\cap)$ is equal to $p^{d_v}(1-p)^{n_v}$, and thus the middle sum in (3.3) is the expectation of

$$\sum_{v \in V((R \cup S)^\cap)} n_v^c\,,$$

which equals $N\mathcal{E}_3$. Recall that $N = Cr^{\lfloor d/2 \rfloor}(n/r)^c$.

To prove the upper bound of $O(1/C)$, we consider another sample $\bar{S}$ drawn from $H \setminus R$ with probability $\bar{p} = p/2$. Let Q denote the quantity

$$\sum_{v \in V(H) \cap R^\cap} \bar{p}^{d_v}(1-\bar{p})^{n_v}\,.$$

This sum is nothing else than the expected number of vertices of $(R \cup \bar{S})^\cap$, which is at most $\mathbf{E}\,|R \cup \bar{S}|^{\lfloor d/2 \rfloor} = O(r^{\lfloor d/2 \rfloor})$, using the Upper Bound Theorem and Lemma 3.2. We estimate

$$\frac{1-\bar{p}}{1-p} \geq 1 + p/2 \geq e^{p/4} \geq e^{r/8n}\,.$$

Then we rewrite

$$Q = \sum_v p^{d_v} 2^{-d_v} (1-p)^{n_v} \left(\frac{1-\bar{p}}{1-p}\right)^{n_v} \geq 2^{-d} \sum_v p^{d_v}(1-p)^{n_v} e^{n_v r/8n}\,. \tag{3.4}$$

We have

$$n_v^c = \left(\frac{8cn}{r}\right)^c \left(\frac{n_v r}{8cn}\right)^c \leq \left(\frac{8cn}{r}\right)^c e^{n_v r/8n}.$$

Substituting this estimate into the middle sum in (3.3) and comparing with the lower bound for Q in (3.4), we obtain

$$\sum_v p^{d_v}(1-p)^{n_v} n_v^c \leq 2^d \left(\frac{8cn}{r}\right)^c Q = O((n/r)^c r^{\lfloor d/2 \rfloor}) = O(N/C)\,. \qquad \square$$

Putting these three lemmas together shows that $\mathcal{E} < 1/2$ for an suitable choice of C. Using Markov's inequality, this shows that $F_1(S) + F_2(S) + F_3(S) < 1$ or also that conditions C1–C3 are satisfied, with probability at least 1/2.

We can now finish the running time analysis of the randomized algorithm, under the condition that at each round, the sample S picked by the algorithm satisfies conditions C1–C3. Indeed, the work in the jth round is given in (3.2). Due to condition C2, the first term of the inner sum adds up to $O_C(r^{\lfloor d/2 \rfloor})$ over all the simplices of the geode, while condition C3 implies that the second term adds up to $O_C(r^{\lfloor d/2 \rfloor})\frac{n}{r} = O_C(nr^{\lfloor d/2 \rfloor - 1})$. Suppose that after the $(j-1)$st round, r hyperplanes have been inserted. The work in the jth round is then $O_C(r^{\lfloor d/2 \rfloor} + nr^{\lfloor d/2 \rfloor - 1})$. Finally, condition C1 implies that the size of the sample grows geometrically between rounds. Thus, if conditions C1–C3 are satisfied at every round, the total running time sums up as $O_C(n \log n + n^{\lfloor d/2 \rfloor})$.

In the derandomized algorithm, conditions C1–C3 will be fulfilled at all rounds. The total running time of the deterministic algorithm will thus be $O_C(n \log n + n^{\lfloor d/2 \rfloor})$

plus whatever time is needed to perform the deterministic computation of a suitable sample at each round.

The randomized algorithm does not check if the conditions are satisfied, however. Thus a bad sample at a given round could severely slow down the algorithm, however unprobable this may be. Nevertheless, the expected time of the algorithm is, by linearity of expectations, the sum of the expected times of all the rounds, and the expected time of a round is bounded by $O_C(r^{\lfloor d/2\rfloor} + nr^{\lfloor d/2\rfloor -1})$ as shown by Lemmas 3.3 to 3.5. Therefore, the expected time of the randomized algorithm is $O_C(n\log n + n^{\lfloor d/2\rfloor})$.

3.3. Derandomization — a first attempt. Let us first consider a straightforward derandomization of the above described algorithm by the Raghavan-Spencer method, recalling the basic strategy of that method and introducing some more notation. We are at the beginning of a round, with the geode of R as a semicutting, and we want to find $S \subseteq H \setminus R$ such that $F_1(S) + F_2(S) + F_3(S) \leq 1$ (thus satisfying conditions C1–C3). We order the hyperplanes of $H \setminus R$ into a sequence $h_1, h_2, \ldots, h_{n-r}$ (arbitrarily), and we process them one by one, deciding for each h_i whether to *accept* it (that is, to include it in S) or to *reject* it (not to include it in S).

Having processed $h_1, \ldots, h_k$, let $S^{(k)}$ denote the set of accepted hyperplanes among them. We define the energies $\mathcal{E}_j^{(k)}$ as the conditional expectations

$$\mathcal{E}_j^{(k)} = \mathbf{E}\left(F_j(S) \mid S \cap \{h_1, \ldots, h_k\} = S^{(k)}\right)$$

and $\mathcal{E}^{(k)}$ is again their sum. Further we let

$$\mathcal{E}_j^{(k|A)} = \mathbf{E}\left(F_j(S) \mid S \cap \{h_1, \ldots, h_k\} = S^{(k)},\, h_{k+1} \in S\right)$$

measure what the energy would be after accepting h_{k+1}, and similarly

$$\mathcal{E}_j^{(k|R)} = \mathbf{E}\left(F_j(S) \mid S \cap \{h_1, \ldots, h_k\} = S^{(k)},\, h_{k+1} \notin S\right)$$

for what the energy would be after rejecting h_{k+1}.

The strategy dictated by the Raghavan-Spencer method is as follows: the hyperplane h_{k+1} is accepted or rejected, whichever decision gives a lower total energy $\mathcal{E}^{(k+1)}$. The key property of the energy is

$$\mathcal{E}^{(k)} = p\,\mathcal{E}^{(k|A)} + (1-p)\,\mathcal{E}^{(k|R)}\,. \tag{3.5}$$

This together with the decision rule implies that $\mathcal{E}^{(k+1)} \leq \mathcal{E}^{(k)}$ for $k = 0, 1, \ldots, n - r - 1$, therefore the final energy is at most 1, and since it equals $\sum_j F_j(S^{(n-r)})$ for the (already fixed) sample $S^{(n-r)}$, this sample will satisfy the required conditions.

The evaluation of $\mathcal{E}_1^{(k)}$ and $\mathcal{E}_2^{(k)}$ is easy (at least with a limited but sufficient accuracy) and requires no more time than the other operations of the randomized algorithm itself. On the other hand, $\mathcal{E}_3^{(k)}$ appears much more demanding, and we cannot evaluate it exactly, so we use a suitable approximation instead, denoted by $\mathcal{AE}_3^{(k)}$. The sum $\mathcal{AE}^{(k)} = \mathcal{E}_1^{(k)} + \mathcal{E}_2^{(k)} + \mathcal{AE}_3^{(k)}$ will be called the *approximate energy.*

Here is a rough outline of our strategy. We shall be careful to define $\mathcal{AE}_3^{(k)}$ in such a way that it obeys an equation analogous to (3.5). Then we apply Raghavan-Spencer method with the approximate energy instead of the actual energy, producing a sample $S^{(n-r)}$ for which the approximate energy does not exceed the initial approximate energy $\mathcal{AE}^{(0)}$. To make everything work, we show the following:

LEMMA 3.6. *For every $k = 0, 1, \ldots, n-r$, $|\mathcal{E}_3^{(k)} - \mathcal{A}\mathcal{E}_3^{(k)}| < 1/3$.* The lemma is proved in the next section, where we explain how to compute the approximate energy. Using this for $k = 0$ together with $\mathcal{E}^{(0)} < 1/3$ (which follows from the results of section 3.2), we see that the initial approximate energy is smaller than $2/3$, and hence so is the final approximate energy. This in turn implies that the final energy $\mathcal{E}^{(n-r)}$ is less than 1, and hence that the sample $S^{(n-r)}$ satisfies conditions C1–C3.

3.4. Approximating the energy. In this section we define the approximate energy $\mathcal{A}\mathcal{E}_3^{(k)}$ and establish Lemma 3.6, assuming the existence of a certain oracle. The implementation of the oracle is discussed in the next section.

We begin by setting the initial value $\mathcal{A}\mathcal{E}_3^{(0)}$. Consider the expression for $\mathcal{E}_3 = \mathcal{E}_3^{(0)}$ in (3.3). We split the sum according to the simplices of $\mathcal{G}(R)$ containing the respective vertices, and we get

$$\mathcal{E}_3^{(0)} = \frac{1}{N} \sum_{s \in \mathcal{G}(R)} \sum_{v \in V(H) \cap s} p^{d_v} (1-p)^{n_v} n_v^c \,.$$

By a suitable general position assumption, we may suppose that a j-dimensional simplex $s \in \mathcal{G}(R)$ contains no vertices of $V(H)$ unless it is a part of a j-face of the polytope $R^\cap$, and thus a vertex of $V(H)$ in such a j-simplex is contained in $d-j$ hyperplanes of R and j hyperplanes of $H \setminus R$, or in other words, $d_v = j = \dim\, s$. In this sense, all the vertices within s are of the same type and we have $V(H) \cap s = V(H, s)$. This implies that

$$\mathcal{E}_3^{(0)} = \frac{1}{N} \sum_{s \in \mathcal{G}(R)} \sum_{v \in V(H,s)} p^{d_v} (1-p)^{n_v} n_v^c \,. \tag{3.6}$$

We describe an oracle, to be constructed later, that performs an approximate evaluation of the sums over a given simplex s. Let $\mathcal{O}$ be an oracle whose input is a j-simplex $s \in \mathcal{G}(R)$, and whose output is a number $\mathcal{O}(s)$, satisfying

$$\Big| \mathcal{O}(s) - \sum_{v \in V(H,s)} p^j (1-p)^{n_v} n_v^c \Big| \le E_s \stackrel{\text{def}}{=} \frac{1}{C^2 q_s^{\sqrt{c}}} n_s^c \,. \tag{3.7}$$

(Here is an attempt to give the reader some intuition about the choice of the error term E_s: the simplex s contains at most n_s^j vertices of $V(H)$, and for each vertex the summand is at most $p^j n_s^c$, thus the exact sum does not exceed $n_s^c q_s^d$. The approximation's relative accuracy is thus a suitable quasi-constant factor.)

We then define the initial approximate energy by

$$\mathcal{A}\mathcal{E}_3^{(0)} \stackrel{\text{def}}{=} \frac{1}{N} \sum_{s \in \mathcal{G}(R)} \mathcal{O}(s) \,.$$

Assuming (3.6) and (3.7), we have

$$\Big| \mathcal{A}\mathcal{E}_3^{(0)} - \mathcal{E}_3^{(0)} \Big| \le \frac{1}{N} \sum_{s \in \mathcal{G}(R)} E_s \le \frac{1}{C^3 r^{\lfloor d/2 \rfloor}} \sum_{s \in \mathcal{G}(R)} q_s^{c-\sqrt{c}} = O(1/C^2) \tag{3.8}$$

by the semicutting property of R, which establishes Lemma 3.6 for $k = 0$, provided C is big enough.

We proceed to the definition of $\mathcal{AE}_3^{(k)}$. For a vertex $v \in V(H) \cap (R \cup S^{(k)})^\cap$, let m_v denote the number of hyperplanes *among* $\{h_{k+1}, \dots, h_{n-r}\}$ in its conflict list[2] (that is, not counting the rejected hyperplanes). We also let d_v be the number of hyperplanes among $\{h_{k+1}, \dots, h_{n-r}\}$ passing through v. In a manner analogous to the above expression for $\mathcal{E}_3$, we can write

$$\mathcal{E}_3^{(k)} = \frac{1}{N} \sum_v p^{d_v} (1-p)^{m_v} n_v^c\,, \tag{3.9}$$

where the summation is taken over all vertices v of the arrangement of $R \cup S^{(k)} \cup \{h_{k+1}, \dots, h_{n-r}\}$ lying in the (closed) polytope $(R \cup S^{(k)})^\cap$.

We describe an oracle $\mathcal{O}^{(k)}$, which can approximately evaluate a part of this sum over a suitable cell (the oracle $\mathcal{O}$ above can be seen as a weaker version of $\mathcal{O}^{(0)}$). The input of $\mathcal{O}^{(k)}$ is a j-dimensional cell σ. We assume that the affine span of σ is either $\mathbb{R}^d$ or contained in the intersection of hyperplanes of H (under a suitable general position assumption, other cells do not contain any relevant vertices) and that σ is completely contained in a single simplex $s \in \mathcal{G}(R)$ (this latter requirement is not so important for the current section, but it is needed in the construction of the oracle). The oracle returns a number $\mathcal{O}^{(k)}(\sigma)$ with

$$\left| \mathcal{O}^{(k)}(\sigma) - \sum_{v \in V(\{h_{k+1}, \dots, h_{n-r}\}, \sigma)} p^j (1-p)^{m_v} n_v^c \right| \le E_s = \frac{1}{C^2 q_s^{\sqrt{c}}} n_s^c, \tag{3.10}$$

where E_s is defined as in (3.7).

It might now seem natural to evaluate the approximate energy $\mathcal{AE}_3^{(k)}$ as follows: keep the portions of the arrangement of $S^{(k)}$ within each simplex $s \in \mathcal{G}(R)$, and call the oracle $\mathcal{O}^{(k)}$ on each cell from the resulting arrangements. It turns out that the error introduced in this way would be too large. Instead we compute the approximate energy incrementally, using the oracle to approximate the *difference* in energy caused by adding or rejecting a hyperplane.

Let us look at what happens with the contribution of various vertices to the total energy $\mathcal{E}_3$ when a hyperplane h_{k+1} is accepted or rejected; we begin with the accepting case. The contribution of vertices strictly above h_{k+1} remains unchanged (we say 'above' meaning 'on the same side of h_{k+1} as the origin'). The contribution of all vertices strictly below h_{k+1} becomes zero, and finally for each vertex on h_{k+1}, d_v decreases by one so that its contribution to the energy is multiplied by $1/p$. Denoting by $\mathcal{E}_{on}^{(k)}$ the contribution of the vertices on h_{k+1} to the sum (3.9) and $\mathcal{E}_{below}^{(k)}$ the contribution of the vertices below, we have

$$\mathcal{E}_3^{(k|A)} = \mathcal{E}_3^{(k)} - \mathcal{E}_{below}^{(k)} + \left(\frac{1}{p} - 1\right) \mathcal{E}_{on}^{(k)}\,.$$

Thus, an appropriate action after accepting h_{k+1} is the following: we let Σ_{on} be the set of all faces σ of the polytope $(R \cup S^{(k)})^\cap \cap h_{k+1}$ within s for all $s \in \mathcal{G}(R)$. We set

$$\mathcal{AE}_{on}^{(k)} \stackrel{\text{def}}{=} \frac{p}{N} \sum_{\sigma \in \Sigma_{on}} \mathcal{O}^{(k)}(\sigma)\,.$$

[2]To be formally consistent, we should also superscript m_v by (k), but this would overburden the notation.

(Note that the oracle includes the $p^{\dim \sigma}$ multiplicative factor, while an appropriate factor for a vertex on h_{k+1} in $\mathcal{E}_{on}^{(k)}$ is $p^{\dim \sigma+1}$; this is why the factor p appears in the definition.)

Then we gather the portion of $(R \cup S^{(k)})^{\cap}$ (strictly) below h_{k+1} inside each s, obtaining a set Σ_{below} of cells, and set

$$\mathcal{AE}_{below}^{(k)} \stackrel{\text{def}}{=} \frac{1}{N} \sum_{\sigma \in \Sigma_{below}} \mathcal{O}^{(k)}(\sigma)\,.$$

We define

$$\mathcal{AE}_3^{(k+1)} \stackrel{\text{def}}{=} \mathcal{AE}_3^{(k|A)} = \mathcal{AE}_3^{(k)} - \mathcal{AE}_{below}^{(k)} + \left(\frac{1}{p} - 1\right) \mathcal{AE}_{on}^{(k)}\,.$$

The discussion of the case when h_{k+1} is rejected is similar. The contribution of all vertices lying on h_{k+1} to the energy vanishes, and the number m_v for all vertices below h_{k+1} decreases by one, thus their contribution to the energy is multiplied by $1/(1-p)$. Hence an appropriate incremental definition is

$$\mathcal{AE}_3^{(k+1)} \stackrel{\text{def}}{=} \mathcal{AE}_3^{(k|R)} = \mathcal{AE}_3^{(k)} + \left(\frac{1}{1-p} - 1\right) \mathcal{AE}_{below}^{(k)} - \mathcal{AE}_{on}^{(k)}\,.$$

From these definitions, the promised analogy to (3.5), namely

$$\mathcal{AE}_3^{(k)} = p\,\mathcal{AE}_3^{(k|A)} + (1-p)\,\mathcal{AE}_3^{(k|R)}\,. \tag{3.11}$$

follows immediately.

As usual, we accept h_{k+1} if $\mathcal{AE}^{(k|A)} < \mathcal{AE}^{(k|R)}$ (otherwise, we reject it). Let us remark that since we have already established $\mathcal{AE}_3^{(0)} \le 2/3$, we know that the final approximate energy $\mathcal{AE}^{(n-r)} < 2/3$, and in particular that conditions C1 and C2 hold for the final sample $S^{(n-r)}$. Thus, any intermediate sample $S^{(k)}$ satisfies $|S^{(k)}| \le r$ as well as condition C2. We are thus free to use these conditions further on. From now on, the quantity r_s will be defined with respect to the final sample $S^{(n-r)}$ computed by the algorithm, that is, $r_s = |S_{|Os}^{(n-r)}| + 1$.

Proof. Proof of Lemma 3.6:

Let us analyze the approximation error. We have

$$\begin{aligned}\left|\mathcal{E}_3^{(k)} - \mathcal{AE}_3^{(k)}\right| \le \left|\mathcal{E}_3^{(0)} - \mathcal{AE}_3^{(0)}\right| &+ \\ \sum_{i;\ h_i \in S^{(k)}} &\left(\left|\mathcal{E}_{below}^{(i-1)} - \mathcal{AE}_{below}^{(i-1)}\right| + \frac{1-p}{p}\left|\mathcal{E}_{on}^{(i-1)} - \mathcal{AE}_{on}^{(i-1)}\right|\right) + \\ \sum_{i \in \{1,\ldots,k\};\ h_i \notin S^{(k)}} &\left(\frac{p}{1-p}\left|\mathcal{E}_{below}^{(i-1)} - \mathcal{AE}_{below}^{(i-1)}\right| + \left|\mathcal{E}_{on}^{(i-1)} - \mathcal{AE}_{on}^{(i-1)}\right|\right)\,.\end{aligned}$$

From (3.8), we know that the first term is $O(1/C)$. Let us consider the contribution of a single simplex $s \in \mathcal{G}(R)$ to the second and third terms; we consider the accepting and rejecting cases separately. The sets Σ_{on} and Σ_{below} have no more than $O(r_s^d)$ cells, and there are fewer than r_s accepted hyperplanes cutting s or separating it from O. For an accepted hyperplane h_i, the error of the oracle given by (3.10) is multiplied by $(1-p)/N \le 1/N$ for the cells in Σ_{on}, and by $1/N$ for the cells in Σ_{below}. The total error for the accepted hyperplanes is thus $O(r_s^{d+1} E_s/N)$.

For a rejected hyperplane h_i, the error for cells in Σ_{on} gets multiplied by p/N, and for cells in Σ_{below} by $p/(1-p)N \le 2p/N$. Thus, the contribution to the error is at most $O(n_s r_s^d p E_s/N) = O(q_s r_s^d E_s/N)$. Substituting for E_s and N, we get that the total contribution to the error for s does not exceed $O(q_s^{c-\sqrt{c}+1} r_s^{d+1}/C^3 r^{\lfloor d/2 \rfloor})$.

We have bounds for the sums of cth moments of the r_s and of the q_s. In order to deal with the product of their powers, we use the inequality $xy \le x^u + y^v$, where the exponents satisfy $1/u + 1/v = 1$. In our case we have $x = r_s^{d+1}$, $y = q_s^{c-\sqrt{c}+1} \le q_s^{c-\sqrt{c}/2}$, $u = 2\sqrt{c}$, $v = 1/(1-1/u) = c/(c-\sqrt{c}/2)$. Then $r_s^{d+1} q_s^{c-\sqrt{c}+1} \le r_s^{2(d+1)\sqrt{c}} + q_s^c$. The total error over all simplices thus becomes

$$\frac{O(1)}{C^3 r^{\lfloor d/2 \rfloor}} \left(\sum_{s \in \mathcal{G}(R)} r_s^{2(d+1)\sqrt{c}} + \sum_{s \in \mathcal{G}(R)} q_s^c \right).$$

The first sum is less than $C^2 r^{\lfloor d/2 \rfloor}$ by condition C2, and the second sum is $O(C r^{\lfloor d/2 \rfloor})$ by the semicutting property of the geode of R, hence the whole expression is $O(1/C)$. This proves Lemma 3.6. $\square$

It now remains for us to implement the oracle.

3.5. Implementing the oracle. LEMMA 3.7. *It is possible to maintain a data structure for each simplex $s \in \mathcal{G}(R)$ such that a call to the oracle $\mathcal{O}^{(k)}$ with a cell σ as described in the previous sections can be answered in $O_C(\mathrm{compl}(\sigma) q_s^{b\sqrt{c}})$ time for an absolute constant b, where $\mathrm{compl}(\sigma)$ denotes the combinatorial complexity of σ. The total time needed for updating the data structure for s during the round is bounded by $O_C(n_s r_s^d q_s^{b\sqrt{c}})$.*

Proof. The proof follows a similar construction in [7]. We define another quasi-constant quantity

$$\rho_s \stackrel{\text{def}}{=} C^{3d+6} q_s^{3d\sqrt{c}}.$$

Let $H_s^{(k)}$ denote the set of yet unprocessed hyperplanes in the conflict list of s, that is, $H_s^{(k)} = \{h_{k+1}, \ldots, h_{n-r}\}_{|Os}$. Whenever we want to call the oracle $\mathcal{O}^{(k)}$, we make sure we have an ε-approximation $A_s^{(k)}$ for the set $H_s^{(k)}$ (with ranges defined by segments) available, where ε is such that the absolute error of the approximation does not exceed n_s/ρ_s, and $|A_s^{(k)}| = O(\rho_s^2 \log \rho_s)$. A simple way to maintain such an ε-approximation under the deletion of hyperplanes is to start with, say, a $(1/2\rho_s)$-approximation $A_s^{(0)}$, keep it unchanged for a while and recompute a fresh $(1/2\rho_s)$-approximation after every $n_s/2\rho_s$ deletions. By the results of [20] (or by Theorem 2.1), each $(1/2\rho_s)$-approximation is computed in time conservatively estimated as $O(\rho_s^{2d+1} n_s)$, so the total time for the maintenance of the ε-approximations is $O_C(n_s q_s^{b\sqrt{c}})$ as claimed.

Suppose that we want to answer a call to the oracle $\mathcal{O}^{(k)}$ with a j-cell σ, that is, approximate the sum

$$\sum_{v \in V(H_s^{(k)}, \sigma)} p^j (1-p)^{m_v} n_v^c.$$

Let us set

$$\alpha_s \stackrel{\text{def}}{=} \frac{|H_s^{(k)}|}{|A_s^{(k)}|}.$$

We are ready to define the oracle value

$$\mathcal{O}^{(k)}(\sigma) \stackrel{\text{def}}{=} \sum_{v \in V(A_s^{(k)}, \sigma)} \alpha_s^j p^j (1-p)^{m_v} n_v^c \,. \tag{3.12}$$

Note that the quantities m_v and n_v can be computed exactly for all the vertices $v \in V(A_s^{(k)}, \sigma)$, in time conservatively estimated as $O_C(\rho_s^{3d} n_s)$ (we remark that we use them as well in (3.7), (3.10) for a vertex $v \in V(\{h_{k+1}, \ldots, h_{n-r}\}, \sigma)$, but these quantities are never actually computed). Up to easy details, we have thus completely described the algorithm implementing the oracle, and the times it takes to answer a call or maintain the data structure can easily be shown to stay within the claimed bounds. It remains to establish the bound on the accuracy.

In order to have a notationally simpler proof, we deal with the case $k = 0$ only (then $n_v = m_v$, and we can omit all the (k) superscripts). The case of a general k is entirely similar and is treated in detail in [5].

Our procedure replaces a summation over a discrete but large set of vertices by summation over a smaller suitably chosen set, and this very much resembles a numerical integration procedure. The notation is chosen to stress this analogy, and also the reasoning in the proof resembles simple estimates for the error of numerical integration. From a result of [7], we know that the number of vertices of the ε-approximation within any simplex multiplied by a suitable scaling factor approximates the number of vertices of H within the simplex with relative accuracy ε. Theorem 2.8 shows that this is even valid for cells of arrangements (since they are convex). Our strategy is to subdivide σ into small enough cells, so that the variation of the summand within each cell is small, but at the same time the number of small cells is not too large. Within each small cell, we treat the summand as essentially constant and use the vertex number approximation bound.

Here is a more detailed treatment. Our specific function f to be integrated (over the discrete set of vertices) is

$$f(t) = p^j (1-p)^t t^c \,.$$

We let $M_{f,T}$ stand for its *modulus of continuity* $M_{f,T}$ over a set T (as is done in a somewhat different context in [22]):

$$M_{f,T}(h) = \sup_{\substack{t_1, t_2 \in T \\ |t_2 - t_1| \le h}} |f(t_2) - f(t_1)| \,.$$

For a j-dimensional cell $\xi \subseteq \sigma$, let us denote

$$\Sigma_\xi f = \sum_{v \in V(H, \xi)} f(n_v) \tag{3.13}$$

and

$$\tilde{\Sigma}_\xi f = \sum_{v \in V(A_s, \xi)} \alpha_s^j f(n_v) \,. \tag{3.14}$$

We want to obtain a bound for the difference $|\Sigma_\xi f - \tilde{\Sigma}_\xi f|$. Let $T_\xi = \{n_v : v \in V(H, \xi)\}$, $f_\xi^{min} = \min_{t \in T_\xi} f(t)$, $f_\xi^{max} = \max_{t \in T_\xi} f(t)$, and $\Delta_\xi(f) = f_\xi^{max} - f_\xi^{min}$. We have

$$|V(H, \xi)| f_\xi^{min} \le \Sigma_\xi f \le |V(H, \xi)| f_\xi^{max}$$

and similarly

$$\alpha_s^j |V(A_s,\xi)| f_\xi^{min} \le \tilde{\Sigma}_\xi f \le \alpha_s^j |V(A_s,\xi)| f_\xi^{max} .$$

From this we get

$$\left|\Sigma_\xi f - \tilde{\Sigma}_\xi f\right| \le \left|\alpha_s^j |V(A_s,\xi)| - |V(H,\xi)|\right| f_\xi^{max} + |V(H,\xi)| \Delta_\xi(f) . \tag{3.15}$$

Theorem 2.8 shows that, for any j-dimensional cell ξ within s, we have

$$\left|\alpha_s^j |V(A_s,\xi)| - |V(H,\xi)|\right| \le \frac{n_s^j}{\rho_s} \tag{3.16}$$

To estimate the total error within σ, we subdivide σ into smaller cells. Namely, we fix a parameter

$$\nu_s \stackrel{\text{def}}{=} C^3 q_s^{2\sqrt{c}} ,$$

and we choose a $(1/\nu_s)$-net $\mathcal{N} \subseteq H_{|O s}$ (with respect to ranges defined by segments) of size $O(\nu_s \log \nu_s)$. We let Ξ be the portion of the arrangement of $\mathcal{N}$ within σ (by a suitable general position assumption, we may consider only j-dimensional cells in Ξ). Ξ is thus a subdivision of σ into $|\Xi| = O((\nu_s \log \nu_s)^j) = O(\nu_s^{d+1})$ cells.

Lemma 3.8. *For any $T \subseteq [0,\tau]$, $M_{f,T}(h) \le h\, p^j \tau^{c-1} (c - \log(1-p)\tau)$.*

Proof. We have $f'(t) = p^j (1-p)^t (c t^{c-1} + \log(1-p) t^c)$. By the mean value theorem, for any $t_1, t_2 \in T$,

$$|f(t_2) - f(t_1)| \le |t_2 - t_1| \sup_{t \in [t_1,t_2]} |f'(t)|$$

from which the result follows. (The reader should keep in mind that $1-p<1$, hence the minus sign in front of the logarithm when taking the absolute values of f'.) ◻

It now suffices to note, by the $(1/\nu_s)$-net property, and since for any $\xi \in \Xi$ we have $\sup T_\xi \le n_s$, that

$$\Delta_\xi(f) \le M_{f,T_\xi}\left(\frac{n_s}{\nu_s}\right) = O\left(\frac{n_s}{\nu_s} p^j (1 + \frac{r}{n} n_s) n_s^{c-1}\right) = O\left(\frac{q_s p^j}{\nu_s} n_s^c\right) \tag{3.17}$$

for any $\xi \in \Xi$, where we have taken into account the fact that $-\log(1-p) = O(r/n)$.

To obtain the total approximation error made by the oracle, we substitute (3.16), (3.17), into (3.15) and this yields

$$\left|\Sigma_\sigma f - \tilde{\Sigma}_\sigma f\right| \le \sum_{\xi \in \Xi} \left|\Sigma_\xi f - \tilde{\Sigma}_\xi f\right| \le O(\nu_s^{d+1}) \frac{n_s^j}{\rho_s} f_\sigma^{max} + O(n_s^j) \frac{q_s p^j}{\nu_s} n_s^c =$$

$$O\left(\frac{\nu_s^{d+1} q_s^d}{\rho_s} + \frac{q_s^{d+1}}{\nu_s}\right) n_s^c = O(E_s/C) ,$$

which validates (3.7), (3.10). ◻

As mentioned in the lemma, the time needed to answer a call to the oracle within a cell σ also depends on the complexity compl(σ) of that cell. However, the total complexity of all the cells in the sets Σ_{below} and Σ_{on} introduced in processing a hyperplane h_i is easily seen to be in $O(r_s^d)$; hence, the total time spent by the oracle when processing a hyperplane is $O_C(r_s^d q_s^{b\sqrt{c}})$.

With this lemma we can finish the time analysis of the whole algorithm. We have already seen at the end of section 3.2 that the running time of the deterministic algorithm is no more than the expected running time of the randomized algorithm plus the cost of computing the good sample and that of the oracle calls. The total time spent for computing local arrangements and testing each hyperplane during the round is easily shown to be $O_C(nr^{\lfloor d/2 \rfloor -1})$ using the fact that the geode of R defines a semicutting. This does not account for the oracle costs. There are at most n_s hyperplanes to process within the simplex s during the computation, so the total time needed will be at most proportional to

$$\sum_{s \in \mathcal{G}(R)} n_s r_s^d q_s^{b\sqrt{c}} \leq \frac{n}{r} \left(\sum_{s \in \mathcal{G}(R)} r_s^{2d} + \sum_{s \in \mathcal{G}(R)} q_s^{2b\sqrt{c}+2} \right)$$

which is $O_C(nr^{\lfloor d/2 \rfloor -1})$, by a calculation similar to that given at the end of the proof of Lemma 3.6. This is also the total running time for one round, which is then bounded by the expected running time of one round for the randomized algorithm. To summarize:

THEOREM 3.9. *The algorithm presented above computes the convex hull of n points deterministically in time $O(n \log n + n^{\lfloor d/2 \rfloor})$ for any fixed dimension $d \geq 2$.*

One final note is needed about the model of computation. The algorithm given here works in the so-called real-RAM model [1], where elementary arithmetic operations take unit time regardless of the size of the numbers. This is the traditional model used in computational geometry, to be contrasted with the so-called bit-model [1] where the size of the numbers also contributes to the time complexity. Note that exponentially large quantities are needed in the course of the algorithm. Nevertheless, because the algorithm runs in polynomial time, it is possible, while computing over logarithmic-size words, to approximate any intermediate number in our algorithm with a relative error smaller than an arbitrarily small constant: it is easily seen that such errors are too small to be of consequence in the derandomization technique we use.

REFERENCES

[1] A. V. AHO AND J. E. HOPCROFT AND J. D. ULLMAN, *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1983.

[2] N. ALON AND J. SPENCER, *The Probabilistic Method*, John Wiley & Sons, New York, 1992.

[3] N. AMATO, M. GOODRICH AND E. RAMOS, *Parallel algorithms for higher-dimensional convex hulls*, in Proc. 35th Annu. IEEE Sympos. Found. Comput. Sci., 1994, pp 683–694.

[4] N. AMATO, M. GOODRICH AND E. RAMOS, Computing faces in segment and simplex arrangements. In *Proc. 27th ACM Symp. Theor. Comput.*, pages 672–682, 1995.

[5] H. BRÖNNIMANN, *Derandomization of geometric algorithms*, Ph.D. Thesis, Dep. of Comput. Sci., Princeton University, 1995.

[6] B. CHAZELLE, *Cutting hyperplanes for divide-and-conquer*, Disc. Comput. Geom. 9 (1993), 145–158.

[7] B. CHAZELLE, *An optimal convex hull algorithm in any fixed dimension*, Disc. Comput. Geom. 10 (1993), 377–409.

[8] B. Chazelle, H. Edelsbrunner, M. Grigni, L. J. Guibas and M. Sharir, *Improved bounds on weak ε-nets for convex sets*, Disc. Comput. Geom. 13 (1995), 1–15.
[9] B. Chazelle and J. Friedman, *A deterministic view of random sampling and its use in geometry*, Combinatorica 10 (1990), 229–249.
[10] B. Chazelle and J. Matoušek *On linear-time deterministic algorithms for optimization problems in fixed dimension*, J. of Algorithms 21 (1996), 579–597.
[11] K. L. Clarkson, *A Randomized Algorithm for Closest-Point Queries*, SIAM J. Comput. 17 (1988), 830–847.
[12] K. L. Clarkson, *Randomized geometric algorithms*, in Computing in Euclidean Geometry, D.-Z. Du, F.K. Kwang, eds., Lecture Notes Series on Comp. 1 (1992), World Scientific, 117–162.
[13] K. L. Clarkson, P. W. Shor, *Applications of random sampling in computational geometry, II*, Disc. Comput. Geom. 4 (1989), 387–421.
[14] R. M. Dudley and R. S. Wenocur, *Some special Vapnik-Chervonenkis classes*, Discrete Math. 33 (1981), 313–318.
[15] H. Edelsbrunner, *Algorithms in combinatorial geometry*, Springer, 1987.
[16] Edelsbrunner, H., Mücke, E.P. *Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms*, ACM Trans. Graph. 2 (1990), 66–104.
[17] D. Haussler and E. Welzl, *ε-nets and simplex range queries*, Disc. Comput. Geom. 2 (1987), 127–151.
[18] J. Matoušek, *Construction of ε-nets*, Disc. Comput. Geom. 5 (1990), 427–448.
[19] J. Matoušek, *Cutting hyperplane arrangements*, Disc. Comput. Geom. 6 (1991), 385–406.
[20] J. Matoušek, *Approximations and optimal geometric divide-and-conquer*, J. Comput. System Sci. 50:2 (1995), 203–208.
[21] J. Matoušek, *Efficient partition trees*, Disc. Comput. Geom. 8 (1992), 315–334.
[22] H. Niederreiter, *Random number generation and quasi Monte-Carlo methods*, CBMS-NSF 63, SIAM, 1992.
[23] P. Raghavan, *Probabilistic construction of deterministic algorithms: Approximating packing integer programs*, J. Comput. System Sci. 37 (1988), 130–143.
[24] J. Spencer, *Ten Lectures on the Probabilistic Method*, CBMS-NSF, SIAM, 1987.