On Approximate Halfspace Range Counting and Relative Epsilon-Approximations*

Boris Aronov
Dept. Computer Info Science
Polytechnic University
Brooklyn, NY 11201, USA
http://cis.poly.edu/~aronov

Sariel Har-Peled
Dept. Computer Science
University of Illinois
201 N. Goodwin Avenue
Urbana, IL 61801, USA
http://uiuc.edu/~sariel

Micha Sharir
School of Computer Science
Tel Aviv University
Tel Aviv 69978 Israel
and
Courant Institute of Math. Sci.
New York University
New York, NY 10012, USA
michas@post.tau.ac.il

ABSTRACT

The paper consists of two major parts. In the first part, we re-examine relative ε-approximations, previously studied in [12, 13, 18, 25], and their relation to certain geometric problems, most notably to approximate range counting. We give a simple constructive proof of their existence in general range spaces with finite VC dimension, and of a sharp bound on their size, close to the best known one. We then give a construction of smaller-size relative ε-approximations for range spaces that involve points and halfspaces in two and higher dimensions. The planar construction is based on a new structure—spanning trees with small *relative crossing number*, which we believe to be of independent interest.

In the second part, we consider the approximate halfspace range-counting problem in $\mathbb{R}^d$ with *relative* error ε, and show that relative ε-approximations, combined with the shallow partitioning data structures of Matoušek, yields efficient solutions to this problem. For example, one of our data structures requires linear storage and $O(n^{1+\delta})$ preprocessing time, for any $\delta > 0$, and answers a query in time $O(\varepsilon^{-\gamma} n^{1-1/\lfloor d/2 \rfloor} 2^{b \log^* n})$, for any $\gamma > 2/\lfloor d/2 \rfloor$; the choice of γ and δ affects b and the implied constants. Several variants and extensions are also discussed.

Categories and Subject Descriptors: F.2.2 [**Theory of Computation**]: Nonnumerical Algorithms and Problems—*Geometrical problems and computations, computations on discrete structures.*

General Terms: Algorithms, Theory.

Keywords: Epsilon-approximations, range spaces, range queries, approximate range queries, halfspaces, VC-dimension, discrepancy, partition trees.

*Work by BA and MS has been supported by a joint grant from the U.S.-Israel Binational Science Foundation. Work by BA has also been partially supported by NSF ITR Grant CCR-00-81964 and NSA MSP Grant H98230-06-1-0016. Work on this paper by SH-P was partially supported by an NSF CAREER award CCR-0132901. Work by MS was also supported by NSF Grant CCF-05-14079, by Grant 155/05 from the Israel Science Fund, and by the Hermann Minkowski–MINERVA Center for Geometry at Tel Aviv University. A full version of part of the paper is available online at [15].

SCG'07, June 6–8, 2007, Gyeongju, South Korea.

1. INTRODUCTION

The main problem that has motivated the study in this paper is *approximate range counting*. In abstract terms, we are given a *range space* $(X, \mathcal{R})$, where X is a set of n objects and $\mathcal{R}$ is a collection of subsets of X, called *ranges*. In a typical geometric setting, X is a subset of some infinite ground set U (e.g., $\mathbb{R}^d$), and $\mathcal{R} = \{R \cap X \mid R \in \mathcal{R}_U\}$, where $\mathcal{R}_U$ is a collection of subsets (*ranges*) of U of some simple shape (such as halfspaces). (To simplify the notation, we will use $\mathcal{R}$ and $\mathcal{R}_U$ interchangeably.) The goal is to preprocess X into a data structure that supports efficient queries of the form: Given $R \in \mathcal{R}_U$, compute a number t such that

$$(1-\varepsilon)|X \cap R| \le t \le (1+\varepsilon)|X \cap R|.$$

We refer to such an estimate t as an *ε-approximate count* of $X \cap R$. (Typically, but not exclusively, ε is a pre-specified parameter.)

The motivation for seeking approximate range counting techniques is that exact range counting is expensive. For instance, consider the classical *halfspace range counting* problem [20], which is the main specific problem studied in this paper. Here, for a point set of size n in $\mathbb{R}^d$, for $d \ge 2$, the best known algorithm for exact halfspace range counting with near-linear storage takes $O(n^{1-1/d})$ time [20]. As shown in several recent papers, as well as in this paper, faster solutions exist for the approximate case, in which the query time is close to $O(n^{1-1/\lfloor d/2 \rfloor})$ [1, 2, 3, 17, 16].

Notice that the problem becomes more challenging as the size of $X \cap R$ decreases. At the extreme, when $|X \cap R| < 1/\varepsilon$, we must produce the count *exactly*. In particular, we need to be able to detect (without any error) *empty ranges*, i.e., those satisfying $X \cap R = \emptyset$. Thus approximate range counting (in the sense defined above) is at least as hard as range emptiness detection.

We make the standard assumption that the range space $(X, \mathcal{R})$ (or, in fact, $(U, \mathcal{R}_U)$) has *finite* (i.e., independent

of n) VC *dimension* δ which is indeed the case in many geometric applications; see [8, 14, 21, 24] for definitions and more details.

Epsilon-approximations. A standard and general technique for tackling the approximate range counting problem is to use ε-approximations. An *(absolute-error) ε-approximation* for $(X, \mathcal{R})$ is a subset $A \subset X$ such that, for each $R \in \mathcal{R}$,

$$\left| \frac{|A \cap R|}{|A|} - \frac{|X \cap R|}{|X|} \right| < \varepsilon. \tag{1}$$

As shown by Vapnik and Chervonenkis [26] (see also [8, 21, 24]), there always exist absolute-error ε-approximations of size $\frac{c\delta}{\varepsilon^2} \log \frac{\delta}{\varepsilon}$, where c is an absolute constant. As a matter of fact, any random sample of these many elements of X is an ε-approximation with constant probability. Moreover, such a sample of size $\frac{c\delta}{\varepsilon^2} \log \frac{\delta}{\varepsilon} + \frac{c}{\varepsilon^2} \log \frac{1}{q}$ is an ε-approximation with probability at least $1-q$, for a sufficiently large constant c. Therefore, to guarantee success with *high probability*, i.e., with probability of failure at most $1/n^{O(1)}$, one needs to choose a sample of size

$$\frac{c\delta}{\varepsilon^2} \log \frac{\delta}{\varepsilon} + \frac{c'}{\varepsilon^2} \log n = O\left(\frac{1}{\varepsilon^2} \log n \right).$$

Approximations of size $O\left(\frac{\delta}{\varepsilon^2} \log \frac{\delta}{\varepsilon} \right)$ can be constructed in deterministic time $O(\delta)^{3\delta} (\varepsilon^{-2} \log \frac{\delta}{\varepsilon})^{\delta} n$ [9]. In fact, there always exist smaller (absolute-error) ε-approximations, of size

$$O\left(\frac{1}{\varepsilon^{2-2/(\delta'+1)}} \log^{c-1/(\delta'+1)} \frac{1}{\varepsilon} \right),$$

where δ' is the exponent of either the *primal* shatter function (and then $c = 2$) or the *dual* shatter function of the range space $(X, \mathcal{R})$ (and then $c = 1$); see [8, 9, 23].

In this paper, we consider a variant of this classical structure, which provides *relative-error approximations*. Ideally, we want a subset $A \subset X$ such that, for each $R \in \mathcal{R}$,

$$(1-\varepsilon) \frac{|X \cap R|}{|X|} \leq \frac{|A \cap R|}{|A|} \leq (1+\varepsilon) \frac{|X \cap R|}{|X|}. \tag{2}$$

This "definition" suffers however from the same syndrome as the definition of approximate range counting; that is, as $|X \cap R|$ shrinks, the absolute precision of the approximation has to increase. At the extreme, when $A \cap R = \emptyset$, $X \cap R$ must also be empty; in general, we cannot guarantee this property, unless we take $A = X$, which defeats the entire purpose of using small-size ε-approximations to speed up approximate counting.

For this reason, we refine the definition as follows: a *relative (p, ε)-approximation* is a subset $A \subset X$ that satisfies Eq. (2) for each $R \in \mathcal{R}$ with $|R| \geq pn$, where $0 < p < 1$ is another fixed parameter. It is known (see [18]) that there exist subsets with this property of size $\frac{c\delta}{\varepsilon^2 p} \log \frac{1}{p}$, where c is an absolute constant. As a matter of fact, any random sample of these many elements of X is a relative (p, ε)-approximation with constant probability. To guarantee success with probability at least $1 - q$, one needs to sample $\frac{c}{\varepsilon^2 p} \left(\delta \log \frac{1}{p} + \log \frac{1}{q} \right)$ elements of X, for a sufficiently large constant c [18].

To appreciate the above bound on the size of relative (p, ε)-approximations, it is instructive to observe that, for a given parameter p, any absolute error (εp)-approximation A will approximate "large" ranges (of size at least pn) to within *relative* error ε, as is easily checked, so it is a relative (p, ε)-approximation. However, the Vapnik-Chervonenkis bound on the size of A, namely, $\frac{c\delta}{\varepsilon^2 p^2} \log \frac{\delta}{\varepsilon p}$, is larger by roughly a factor of $1/p$ than the bound of [18] stated above.

The existence of a relative (p, ε)-approximation A provides a simple mechanism for approximate range counting: For a range R, count $A \cap R$ exactly, say, by brute force in $O(|A|)$ time, and output $|A \cap R| \cdot |X|/|A|$ as an ε-approximate count of $X \cap R$. However, this will work only for ranges of size at least pn. As we will show in the second part of this paper, an appropriate incorporation of relative (p, ε)-approximations into standard range searching data structures yields a procedure for approximate range counting that works, quite efficiently, for ranges of *any* size.

Our results. This paper has two main parts. In the first part, we present several constructions and bounds involving relative (p, ε)-approximations. We first give an alternative construction of general relative (p, ε)-approximations, which follows the standard discrepancy-based construction of absolute-error (p, ε)-approximations [8], but uses a more careful analysis that shows that the resulting set is indeed a relative (p, ε)-approximation. That is, for a given threshold $0 < p < 1$, the resulting set A is of size $O\left(\frac{\delta}{\varepsilon^2 p} \log \frac{\delta}{\varepsilon p} \right)$, and, for any range R of size at least pn, gives an absolute approximation error of $\varepsilon |R \cap X|/|X|$. That is, Eq. (2) holds. Thus, for such ranges R, $\frac{|R \cap A|}{|A|} \cdot |X|$ is an ε-approximate count of $R \cap X$. The construction is randomized, but can easily be derandomized using standard techniques, similar to those used to obtain the deterministic constructions cited above.

Note that the size of our approximation is slightly worse than the bound of [18], when $p \gg \varepsilon$. However, our construction is useful because it can be enhanced, in certain geometric situations, to yield relative (p, ε)-approximations of smaller size. We study two cases in detail, one involving points in $\mathbb{R}^2$ and halfplane ranges, and the other involving points in $\mathbb{R}^d$, $d \geq 3$, and halfspace ranges. Given a threshold parameter $0 < p < 1$, the size of the approximation set is $O\left(\frac{1}{\varepsilon^{4/3} p} \log \frac{1}{\varepsilon p} \right)$ in the plane, and $O\left(\frac{1}{\varepsilon^{3/2} p} \log \frac{1}{\varepsilon p} \right)$ in 3-space; the bounds for higher dimensions are spelled out in Section 4.2.

In the planar case, the construction is based on an interesting generalization of spanning trees with small crossing number, a result that we believe to be of independent interest. Specifically, we show that any finite point set P in the plane has a spanning tree with the following property: For any $k \leq |P|$, any *k-shallow* line (a line that has at most k points of P in one of the halfplanes that it bounds) crosses at most $O(\sqrt{k} \log(n/k))$ edges of the tree. (The classical construction of Welzl [27] guarantees this property only for $k = n$; i.e., it guarantees the uniform crossing number $O(\sqrt{n})$.) We refer to such a tree as a *spanning tree with low relative crossing number*, and show how to use it in the construction of small-size relative (p, ε)-approximations.

Things are more complicated in three (and higher) dimensions. We were unable to extend the planar construction of

spanning trees with low relative crossing number to higher dimensions, and this remains an interesting open problem. (We give a counterexample that shows why the planar construction cannot be extended "as is" to 3-space.) Instead, we base our construction on the shallow partition theorem of Matoušek [19], and construct a set A of size $O\left(\frac{1}{\varepsilon^{3/2}p}\log\frac{1}{\varepsilon p}\right)$, which yields an absolute approximation error of at most εp for halfspaces that contain *at most* pn points. Note that this is the "wrong" inequality—to guarantee small relative error we need this to hold for all ranges with *at least* pn points. To overcome this difficulty, we construct a *sequence* of approximation sets, each capable of producing a relative ε-approximate count for ranges that have roughly the same size, where these sizes grow geometrically, starting at pn and ending at roughly n. The sizes of our approximation sets decrease geometrically, so that the size of the first set (that caters to ranges with about pn points), which is $O\left(\frac{1}{\varepsilon^{3/2}p}\log\frac{1}{\varepsilon p}\right)$, dominates asymptotically the overall size of all of them. We output this sequence of sets, and show how to use them to obtain an ε-approximate count of any range with at least pn points.

The situation is somewhat more involved in higher dimensions. The basic approach used in the three-dimensional case can be extended to higher dimensions, using the appropriate version of the shallow partition theorem. However, the bounds get somewhat more complicated (see Section 4.2), and apply only under certain restrictions on the relationships between ε, p, and n.

Approximate range counting: Alternative recent solutions. Consider again the problem of approximate range counting and focus on the case of halfspace ranges in $\mathbb{R}^d$. Three recent papers address this problem, and achieve improvements similar to ours. The first result is due to Aronov and Har-Peled [2, 3], who reduce this problem to range emptiness, by performing binary search on the size $|X \cap R|$ for the given range R, until the desired relative error is attained. Each decision step in the search is made by accessing $O\left(\frac{1}{\varepsilon^2}\log n\right)$ different range emptiness structures on certain random samples of X. This technique is a general reduction from approximate range searching to range emptiness testing. In the revised version [3], the algorithm answers a query in time $O\left(\frac{1}{\varepsilon^2}\log n\right) Q_{\text{empty}}(n)$, where $Q_{\text{empty}}(n)$ is the time to answer a range emptiness query. The storage and preprocessing costs are, respectively, $O\left(\frac{1}{\varepsilon^2}\log n\right) S_{\text{empty}}(n)$, and $O\left(\frac{1}{\varepsilon^2}\log n\right) T_{\text{empty}}(n)$, where $S_{\text{empty}}(n)$, $T_{\text{empty}}(n)$ are the storage and preprocessing costs for the range emptiness data structure.

A second approach is presented by Kaplan and Sharir [17], who exploit a general technique of Cohen [10] for estimating the number of data objects in a given subset R of a larger set X (see also [11]). In this approach, one assigns to each data object of X, independently, a random *weight*, drawn from an exponential distribution with density e^{-x}, sorts the objects by their weights into a random permutation, and then finds the *minimum rank* in that permutation of the objects in the query range R. As in the technique of Aronov and Har-Peled, one then repeats this experiment $O(\frac{1}{\varepsilon^2}\log n)$ times,[1] computes the average μ of the *weights* of the minimum elements, and approximates $|R|$ by $1/\mu$.

To apply this machinery to approximate halfspace range counting, one needs a data structure that preprocesses the given set X of points, and a given random permutation thereof, into a data structure that can answer *halfspace-minimum range queries* efficiently: Given a query halfspace h, find the point of X of minimum rank among those contained in h. Kaplan and Sharir present such structures for halfspaces in $\mathbb{R}^3$ (a revised version [16] extends it to any dimension). The performance of their algorithm is comparable to that of Aronov and Har-Peled's algorithm.

A third paper, by Afshani and Chan [1] has just appeared. It caters to the 3D version of the problem, and improves the performance of the preceding algorithms. In particular, it achieves optimal (expected) query time in $\mathbb{R}^3$, with a data structure of expected linear size.

Our results (continued). In the second part of the paper we present an efficient solution for the approximate halfspace range counting problem, in which relative ε-approximations are heavily used. We focus mainly on halfspace ranges in $\mathbb{R}^d$, $d \geq 4$. The performance of our algorithms is comparable with (and somewhat improves) the previous work of [2, 16].

Whereas the algorithm of Aronov and Har-Peled uses a range emptiness procedure as a black box, we examine the inner workings of such a procedure (or of a *shallow range reporting procedure*, which has comparable performance), and turn it into an approximate counting procedure. Informally, the range emptiness or reporting data structures of Matoušek [19] consist of a *partition tree*, whose nodes store certain canonical subsets of X, and which has the property that a query with a range that is *shallow* at a node v (i.e., one that contains only a few points of the subset stored at v; see below for a more precise definition) visits only a small number of children of v. When the procedure realizes that the query visits too many children, it stops and reports that the range cannot be shallow. For emptiness queries, this immediately implies that the range is not empty. For reporting queries, one can then afford to perform the reporting by brute force, knowing that the output size is large enough and thus commensurable with the size of the entire set stored at v.

In contrast, our solution exploits the fact that the range is *deep* (that is, not shallow), to invoke an auxiliary mechanism that approximates its size. Our main auxiliary mechanism is to use relative approximations, as discussed above. In this manner, we derive two variants of our general approach. The first algorithm uses $O(n)$ storage, $O(n^{1+\delta})$ preprocessing time, for any $\delta > 0$ (which reduces to $O(n \log n)$ for certain choices of parameters), and answers approximate halfspace range counting queries in $\mathbb{R}^d$ in $O\left(\varepsilon^{-\gamma} n^{1-1/\lfloor d/2 \rfloor} \operatorname{polylog} n\right)$ time, where γ can be chosen arbitrarily from the interval $(2/\lfloor d/2 \rfloor, 2)$; the choice of γ and δ affects the implied constants and the power of the logarithm in the query time. Note that the storage and preprocessing costs are *independent of* ε, and the dependence of the query time on ε is considerably lower than in the previous approaches.

A slight weakness of this solution is that the query time bound, ignoring its dependence on ε, is comparable with the overhead term in the bound for halfspace range *reporting* [19], whereas the query time in the solution of Aronov and Har-Peled [2, 3] is expressed in terms of the cost of halfspace

[1] In both techniques, this is a consequence of using Chernoff bounds to guarantee high probability of success.

range *emptiness* queries, which is $O\left(n^{1-1/\lfloor d/2\rfloor}\cdot 2^{O(\log^* n)}\right)$ [19]. On one hand, this replaces the polylogarithmic factor in our time bound by a smaller factor, but, on the other hand, one has to multiply this bound by $O\left(\frac{1}{\varepsilon^2}\log n\right)$ in the algorithm of [3], making the dependence on ε slightly worse.

Our second implementation demonstrates that the fine-tuning done in [19] to achieve the improved bound for emptiness queries can also be carried out in our context, leading to an algorithm that uses linear storage and $O(n^{1+\delta})$ preprocessing time, for any $\delta>0$, and answers a query in time $O\left(\varepsilon^{-\gamma}n^{1-1/\lfloor d/2\rfloor}\cdot 2^{O(\log^* n)}\right)$, where γ can be chosen anywhere in the same interval as above. This bound compares favorably with the one in [2], both in terms of the dependence on ε and the factors that depend on $\log n$. Moreover, the storage used by both solutions is $O(n)$, *independent* of ε, which is a significant improvement over the previous results.

The general technique that we propose is sufficiently modular, so as to support various extensions and variants. One interesting variant is a data structure that answers efficiently halfspace range minimum queries, with respect to a random permutation of the input set, of the sort that is needed for the technique of [10, 16, 17] described above. Another variant is a data structure where ε need not be pre-specified, and can be part of the query; in contrast, the "competing" structures described above have to be built with the knowledge of the value of ε.

In closing, we note that in some recent work by Arya and others [5, 6], approximate range counting is interpreted differently, in that one seeks an exact count in a range that closely approximates the input range, according to some geometric error measure.

Due to lack of space, many details are omitted in this version. They can be found in the two full versions [3, 4].

2. RELATIVE APPROXIMATIONS IN GENERAL RANGE SPACES

Our construction is based on the following well known result, which works by pairing up the points into a perfect matching, and randomly coloring the endpoints of each matching edge.

Theorem 2.1 ([8]) *Let $(X,\mathcal{R})$ be a set system defined over $n=|X|$ elements, where $\mathcal{R}=\{R_1,\ldots,R_m\}$. One can construct, in $O(nm)$ deterministic time, a coloring $\chi: X\to\{-1,1\}$, where each color class has exactly $n/2$ elements, such that, for any $j=1,\ldots,m$, the discrepancy of R_j is $|\chi(R_j)|\le\sqrt{2\,|R_j|\ln(2m)}$, where $\chi(R_j)=\sum_{x\in R_j}\chi(x)$.*

Remarks: (1) A more precise statement is that $\chi(R_j)\le\sqrt{2\xi(R_j)\ln(2m)}$, for each j, where $\xi(R)$ is the number of pairs of the matching that the range R separates (or "crosses"). Later, we use special matchings with small values of the quantities $\xi(\cdot)$, and consequently obtain improved discrepancy bounds, which in turn leads to improved bounds on the size of the relative approximation sets.
(2) We refer to the process of extracting a subset of X of half the size, by using low-discrepancy coloring, as *halving*. Note that if the range space $(X,\mathcal{R})$ has VC dimension δ, then, by Theorem 2.1, we can compute such a halving in $O(|X|^{\delta+1})$ deterministic time, since $|\mathcal{R}|=O(|X|^{\delta})$ by Sauer's lemma [8].

Setting $W=\chi^{-1}(1)$ and $B=\chi^{-1}(-1)$, Theorem 2.1 states that, for each R_j, $\big||W\cap R_j|-|B\cap R_j|\big|\le\sqrt{2\,|R_j|\ln(2m)}$. Since $|W\cap R_j|=|X\cap R_j|-|B\cap R_j|$, we have, for each R_j,

$$\big||X\cap R_j|-2\,|B\cap R_j|\big|\le\sqrt{2\,|R_j|\ln(2m)}. \qquad (3)$$

Assume now that the set system $(X,\mathcal{R})$ has finite VC dimension δ. This implies (by Sauer's lemma [24]) that $|\mathcal{R}|\le(ne/\delta)^{\delta}$. To simplify the exposition, we will assume that $\delta>2$, so that $m=|\mathcal{R}|\le n^{\delta}$. Moreover, for any subset $X'\subseteq X$ of size n', the number of distinct ranges in $\mathcal{R}'=\{R_j\cap X'\mid j=1,\ldots,m\}$ is $m'\le(n')^{\delta}$.

We construct a sequence of subsets $P_k\subseteq P_{k-1}\subseteq\ldots\subseteq P_0=X$, with k to be determined shortly, so that, for each $i>0$, P_i is obtained by halving from P_{i-1} via Theorem 2.1. Put $n_i=|P_i|=n/2^i$, for $i=1,\ldots,k$.

Lemma 2.2 *Let $0<p<1$ be a given parameter, and assume that k satisfies $n_k\ge\frac{4\delta}{p}\ln\frac{4\delta}{p}$. Then, for any range R in $\mathcal{R}$ that contains at least pn points of X, and for each $i=0,\ldots,k$, we have, for some absolute constant c,*

$$|P_i\cap R|\le\mathsf{c}\,|P_0\cap R|\,/2^i.$$

Remarks: (1) The proof of Lemma 2.2, given in the full version [15], is somewhat similar to the standard argument showing the existence of ε-approximations using discrepancy [8].
(2) In what follows, we need several variants of Lemma 2.2, in which the starting points are refined versions of Eq. (3). The proofs are easy modifications of the initial proof, and some of them are given in the full version [15].

Lemma 2.3 *There exists an index k such that $|P_k|=n_k=\Theta\left(\frac{\delta}{\varepsilon^2 p}\ln\frac{\delta}{\varepsilon p}\right)$, and such that, for any range $R\in\mathcal{R}$ that contains at least pn points, we have $\big||R\cap P|-2^k|R\cap P_k|\big|\le\varepsilon|R\cap P|$.*

Proof. Fix a range R, and use the following notation: Denote by $\widehat{\lambda}$ the size of $P_0\cap R$, and put $\lambda_i=|P_i\cap R|$, for $i=0,\ldots,k$ (so $\lambda_0=\widehat{\lambda}$). Lemma 2.2 implies that

$$\left|\widehat{\lambda}-2^k\lambda_k\right| \;\le\; \sum_{i=1}^{k}2^{i-1}\sqrt{2\delta\lambda_{i-1}\ln(2n_i)}$$

$$\le\sum_{i=1}^{k}2^{i-1}\sqrt{2\delta\left(c\frac{\widehat{\lambda}}{2^{i-1}}\right)\ln(2n_i)}\le\mathsf{c}_1 2^{k/2}\sqrt{\widehat{\lambda}\ln n_k},$$

which follows, for some constant c_1 proportional to $\sqrt{\delta}$, since this summation behaves like an increasing geometric series and is therefore dominated by the last term. We want the right-hand side to be smaller than $\varepsilon\widehat{\lambda}$, which is equivalent to $\mathsf{c}_1^2 2^k\ln n_k\le\varepsilon^2\widehat{\lambda}$. Since we assume $\widehat{\lambda}\ge pn$, this will hold if we require that $\mathsf{c}_1^2 2^k\ln n_k\le\varepsilon^2 pn$. Since $n_k=n/2^k$, this amounts to requiring $\frac{\mathsf{c}_1^2}{\varepsilon^2 p}\le\frac{n_k}{\ln n_k}$, which holds for $n_k\ge 2\frac{\mathsf{c}_1^2}{\varepsilon^2 p}\ln\frac{\mathsf{c}_1^2}{\varepsilon^2 p}$. This bound meets the lower bound requirement on n_k, given in Lemma 2.2, provided that c_1 is a sufficiently large multiple of $\sqrt{\delta}$. This completes the proof of the lemma. □

Theorem 2.4 *Let $(X, \mathcal{R})$ be a range space with finite* VC *dimension δ, where $|X| = n$, and let $0 < \varepsilon < 1$ and $0 < p < 1$ be given parameters. One can construct a relative (p, ε)-approximation for $(X, \mathcal{R})$, of size $O\left(\frac{\delta}{\varepsilon^2 p} \log \frac{\delta}{\varepsilon p}\right)$, in $\min\left\{O(\delta)^{3\delta}\left(\frac{1}{p^2\varepsilon^2}\log\frac{\delta}{\varepsilon}\right)^{\delta} n,\ O(n^{\delta+1})\right\}$ deterministic time.*

The desired approximation set is simply the set P_k from Lemma 2.3. See [15] for details.

3. RELATIVE APPROXIMATIONS IN THE PLANE

In this section we present a construction of smaller-size relative (p, ε)-approximations for the range space involving a set of points in the plane and the set of halfplanes as ranges. The key ingredient of the construction is the result of the following subsection.

3.1 Spanning trees with small relative crossing number

We derive a refined "weight-sensitive" version of the classical construct of *spanning trees with small crossing number*, as obtained by Chazelle and Welzl [7, 27]. We believe that this refined version is of independent interest, and expect it to have additional applications in the future.

In accordance with standard notation used in the literature, we denote from now on the underlying point set by P. We first recall the standard result:

Theorem 3.1 ([27]) *Let P be a set of n points in $\mathbb{R}^d$. One can compute a spanning tree $\mathcal{T}$ of P such that each hyperplane in $\mathbb{R}^d$ crosses at most $O(n^{1-1/d})$ edges of $\mathcal{T}$.*

Definition 3.2 Let P be a set of n points in the plane. For a non-vertical line ℓ, let w_ℓ^+ (resp., w_ℓ^-) be the number of points of P lying above (resp., below or on) ℓ, and define the *weight* of ℓ, denoted by w_ℓ, to be $\min(w_\ell^+, w_\ell^-)$.

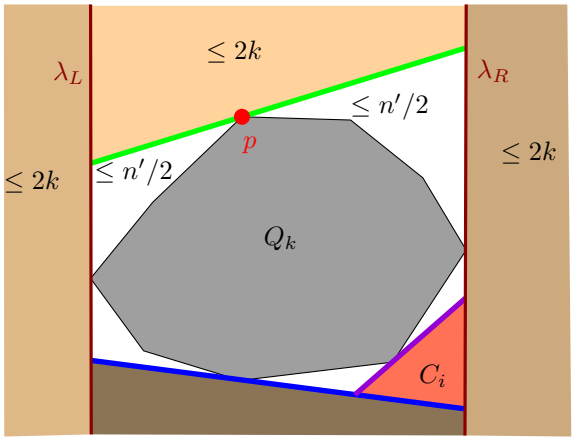
Let $Q_k = Q_k(P)$ be the intersection of all closed halfspaces that contain at least $n - k$ points of P. By the centerpoint theorem (see [22]), the set Q_k is a nonempty convex polygon, for $k < n/3$.

Lemma 3.3 *Let P be a set of n points in the plane. (i) Any line ℓ that avoids the interior of Q_k has weight $w_\ell \le 2k$. (ii) Any line ℓ that intersects the interior of Q_k has weight $w_\ell > k$.*

PROOF. Translate ℓ parallel to itself until it supports Q_k. The new line ℓ' must pass through a vertex v of Q_k which is defined by the intersection of two closed halfplanes, each having k points in its complement. Thus, the union of the complements of these two halfplanes contains at most $2k$ points, and it contains ℓ' and ℓ. Thus, ℓ has at most $2k$ points on one of its sides.

The second claim is easy: If the weight of ℓ were at most k then, by definition, the interior of Q_k would be completely contained on one side of ℓ. □

Lemma 3.4 *The set $P \setminus Q_k$ can be covered by pairwise openly disjoint (possibly unbounded) triangles $C_1, \ldots, C_u$, each containing at most $2k$ points of P, such that* any *line intersects at most $O(\log(n/k))$ of these triangles, Furthermore, $C_i \cap \partial Q_k \neq \emptyset$, for $i = 1, \ldots, u$.*



PROOF. We construct triangles $\widehat{C}_i$ iteratively, as follows. Let λ_L and λ_R be the two vertical lines supporting Q_k on its left and on its right, respectively. $\widehat{C}_1$ (resp., $\widehat{C}_2$) is the halfplane to the left (resp., right) of λ_L (resp., λ_R). The construction maintains the invariant that the complement of the union of the triangles $\widehat{C}_1, \ldots, \widehat{C}_i$ constructed so far is a convex polygon K_i that contains Q_k and the boundary of K_i passes through some of the vertices of Q_k, so that $K_i \setminus Q_k$ consists of pairwise disjoint connected "pockets". (Initially, after constructing $\widehat{C}_1$ and $\widehat{C}_2$, we have two pockets—the regions lying respectively above and below Q_k, between λ_L and λ_R.)

Each step of the construction picks a pocket that contains more than $2k$ points of P, and finds a line ℓ that supports Q_k at a vertex of the pocket, and subdivides the pocket into two sub-pockets and a third piece that lies on the other side of ℓ. The line ℓ is chosen so that the two resulting sub-pockets contain an equal number of points of P. The third piece, which contains at most $2k$ points of P by Lemma 3.3(i), is taken to be the next triangle $\widehat{C}_{i+1}$, and the construction continues in this manner until each pocket has at most $2k$ points. We then terminate the construction, adding the (triangular) convex hulls of all the remaining pockets to the output collection.

One can verify that each line ℓ intersects at most $O(\log(n/k))$ triangles, and that all other properties also hold. See [15] for details. □

Lemma 3.5 *Let $1 \le k \le n$ be a pre-specified parameter. One can construct a spanning tree $\mathcal{T}$ for $P' := P \setminus Q_k$, such that each line intersects at most $O(\sqrt{k}\log(n/k))$ edges of $\mathcal{T}$.*

PROOF. Construct the decomposition of $P \setminus Q_k$ into u covering polygons $C_1, \ldots, C_u$, using Lemma 3.4.

For each $i = 1, \ldots, u$, construct a spanning tree $\mathcal{T}_i$ of $P \cap C_i$ with crossing number $O(k^{1/2})$, using Theorem 3.1. In addition, connect one point of $P \cap C_i$ to an arbitrary vertex of $\partial C_i \cap \partial Q_k$. It is easily checked that the trees $\mathcal{T}_1, \ldots, \mathcal{T}_u$ are vertex-disjoint, and collectively use all the points of $P \setminus Q_k$.

Let G be the planar straight-line graph formed by the union of $\partial Q_k, \mathcal{T}_1, \ldots, \mathcal{T}_u$, plus the connecting segments just introduced, and let $\mathcal{T}^*$ be a spanning tree of G that contains all the points of $P \setminus Q_k$.

Let ℓ be any line in the plane. The proof of the preceding lemma implies that ℓ intersects at most $O(\log(n/k))$ of the polygons C_i. Hence, ℓ crosses at most two edges of ∂Q_k,

at most $O(\log(n/k))$ of the connecting segments, and it can cross edges of at most $O(\log(n/k))$ trees $\mathcal{T}_i$, for $i = 1, \ldots, u$. Since ℓ crosses at most $O(k^{1/2})$ edges of each such tree, we conclude that ℓ crosses at most $O(\sqrt{k}\log(n/k))$ edges of $\mathcal{T}^*$.

Finally, we get rid of the extra "Steiner vertices" of $\mathcal{T}^*$ (those not belonging to $P \setminus Q_k$) in a straightforward manner, by rooting $\mathcal{T}^*$ at some point of $P \setminus Q_k$, and by replacing each path connecting a point $u \in P \setminus Q_k$ to an ancestor $v \in P \setminus Q_k$, where all inner vertices of the path are Steiner points, by the straight segment uv. This produces a spanning tree $\mathcal{T}$ of $P \setminus Q_k$, whose crossing number is at most that of $\mathcal{T}^*$. □

Theorem 3.6 *Given a point set P of n points in the plane, one can construct a spanning tree $\mathcal{T}$ for P such that any line ℓ crosses at most $O(\sqrt{w_\ell}\log(n/w_\ell))$ edges of $\mathcal{T}$. The tree $\mathcal{T}$ can be constructed in $O(n^{1+\varepsilon})$ deterministic time, for any fixed $\varepsilon > 0$.*

PROOF. We construct a sequence of subsets of P, as follows. Put $P_0 = P$. At the ith step, $i \geq 1$, consider the polygon $K_i = Q_{2^i}(P_{i-1})$, and let $P_i = P_{i-1} \cap K_i$. We stop when P_i becomes empty.

For each $i \geq 1$, construct a spanning tree $\mathcal{T}_i$ for $P_{i-1} \setminus K_i$, using Lemma 3.5 (with $k = 2^i$). Connect the resulting trees by straight segments into a single spanning tree $\mathcal{T}$ of P.

We claim that $\mathcal{T}$ is the desired spanning tree. Indeed, consider an arbitrary line ℓ of weight k. Observe that, by Lemma 3.3(ii), ℓ cannot cross any of the polygons K_i, for $i > U = \lceil \log_2 k \rceil$. Thus ℓ crosses only the first $O(\log k)$ layers of our construction and at most

$$\sum_{i=1}^{U} O\Big(\sqrt{2^i}\log(n/2^i)\Big) = O(\sqrt{k}\log(n/k))$$

edges of $\mathcal{T}$, as asserted. □

Remark: For any $n \geq 1$, there exists a set S_n of n points in convex position in 3-space, such that any partition of S_n into sets of size (roughly) k will have a plane that crosses at least $\Omega(\sqrt{n/k})$ sets of the partition. Without the convex position assumption, there exist sets for which this crossing number is at least $\Omega((n/k)^{2/3})$. See [15] for details.

3.2 Relative approximations for halfplanes

We can turn the above construction of a spanning tree with small relative crossing number into a construction of a relative (p, ε)-approximation for a set of points in the plane and halfplane ranges, as follows.

Let P be a set of n points in the plane, and let $\mathcal{T}$ be a spanning tree of P as in Theorem 3.6. By converting $\mathcal{T}$ to a spanning path, and then by picking every other edge of the path, we obtain a perfect matching M of P, with the same relative crossing number, i.e., the number of pairs of M that are separated by a halfplane of weight k is at most $O(\sqrt{k}\log(n/k))$.

We now construct a coloring of P with low discrepancy, by randomly coloring the points in each pair of M, as in Theorem 2.1. The analysis in the standard proof of that theorem (see the remark following the theorem) yields the following variant, which may be a result of independent interest in Discrepancy Theory.

Lemma 3.7 *Given a set P of n points in the plane, one can construct a coloring $\chi : P \mapsto \{-1, 1\}$, such that, for any halfplane h that contains k points of P, we have*

$$\chi(h \cap P) = O(k^{1/4}\log n).$$

We now continue with the analysis of Section 2, using the improved discrepancy bound of the preceding lemma. This can be shown to lead to the following improved bound.

Theorem 3.8 *Given a set P of points in the plane, and parameters $0 < \varepsilon < 1$ and $0 < p < 1$, one can construct a relative (p, ε)-approximation subset of size $O\left(\frac{1}{\varepsilon^{4/3}p}\log^{4/3}\frac{1}{\varepsilon p}\right)$.*

4. RELATIVE APPROXIMATIONS IN HIGHER DIMENSIONS

4.1 Relative Approximations in 3-Space

The construction in higher dimensions is different from the planar one, due to our present inability to extend the construction of spanning trees with low relative crossing number to three or higher dimension.

A hyperplane h *separates* a set $Q \subseteq \mathbb{R}^d$ if h intersects the interior of $\mathcal{CH}(Q)$; namely, each open halfspace bounded by h contains a nonempty subset of Q.

Theorem 4.1 *Let P be a set of n points in $\mathbb{R}^3$, and let $0 < \varepsilon < 1$, $0 < p < 1$ be given parameters. Then there exists a set $A \subset P$, of size $O\left(\frac{1}{\varepsilon^{3/2}p}\log\frac{1}{\varepsilon p}\right)$, such that, for any halfspace h of weight* at most *pn, we have*

$$\left|\frac{|h \cap A|}{|A|} - \frac{|h \cap P|}{|P|}\right| \leq \varepsilon p. \tag{4}$$

Remark: Notice the difference between Eq. (4) and the situation in the preceding sections: Up to now we have handled ranges of size *at least pn*, whereas Eq. (4) applies to ranges of size *at most pn*. This issue requires a somewhat less standard construction, that will culminate in a *sequence* of approximation sets, each catering to a different range of halfspace weights. Nevertheless, the overall size of these sets will satisfy the above bound, and the cost of accessing them will be small. See below for details.

PROOF. Put $k = pn$. We apply the shallow partition theorem of Matoušek [19], to obtain a partition of P into $s \leq n/k$ subsets $P_1, \ldots, P_s$, each of size between $k+1$ and $2k$, such that any *k-shallow* halfspace h (namely, a halfspace that contains at most k points of P) separates at most $c\log s = c\log(1/p)$ subsets, for some absolute constant c. (Note that if h meets any P_i, it has to separate it, because h is too shallow to fully contain P_i.) Without loss of generality, we can carry out the construction so that the size of each P_i is even.

We then construct, for each subset P_i, a spanning tree of P_i with crossing number $O(k^{2/3})$ [7, 27], and convert it to a perfect matching of P_i, with the same asymptotic bound on its crossing number, i.e., the maximum number of pairs in the matching that a halfspace separates.

We then color the endpoints of each matched pair independently with opposite colors, with equal probability. Let R_1 be the set of points colored -1; we have $|R_1| = n/2$. With high probability, the discrepancy of any halfspace h is at most $\sqrt{6\xi(h)\ln(2n)}$, where $\xi(h)$ is the crossing number of h.

Since h is assumed to be k-shallow, it follows by construction that $\xi(h) = O\left(\sum_i u_i^{2/3}\right)$, where $u_i = |h \cap P_i|$, and where the sum extends over those $O(\log r)$ subsets for which $u_i > 0$. Using Hölder's inequality, this yields $\xi(h) = O(k^{2/3}\log^{1/3} r)$, so the discrepancy of h is $O(k^{1/3}\log^{2/3} n)$.

We continue recursively in this manner for j steps, producing a sequence of subsets $R_0 = P, R_1, \ldots, R_j$, where R_i is obtained from R_{i-1} using the above coloring procedure, in which we use, instead of k, the parameter $k_{i-1} = pn \cdot \min\{\mathsf{c}/2^{i-1}, 1\}$, where c is the constant derived in the following lemma, which is a variant of Lemma 2.2 (see [15]).

Lemma 4.2 *For any halfspace h with at most $k = pn$ points of P, we have, for any $i \le j$,*

$$|h \cap R_i| \le \frac{\mathsf{c}k}{2^i} = \frac{\mathsf{c}pn}{2^i},$$

for an appropriate absolute constant c, provided that $n_j \ge \frac{2}{p}\ln\frac{1}{p}$, where $n_j = |P_j| = n/2^j$.

This implies, similar to the planar case,

$$\left|\frac{|h\cap P|}{|P|} - \frac{|h\cap R_j|}{|R_j|}\right| = O\left(\frac{2^{2j/3}k^{1/3}\log^{2/3}(n/2^{j-1})}{n}\right)$$

(see [15] for details). We choose j so that this bound is at most $\varepsilon p = \varepsilon k/n$. That is, $2^j = O\left(\frac{\varepsilon^{3/2}pn}{\log(n/2^j)}\right)$. Hence, the size of R_j is

$$|R_j| = \frac{n}{2^j} = O\left(\frac{\log(n/2^j)}{\varepsilon^{3/2}p}\right) = O\left(\frac{1}{\varepsilon^{3/2}p}\log\frac{1}{\varepsilon p}\right).$$

Taking $A = R_j$ completes the proof of the theorem. □

Theorem 4.1 implies the following [15].

Theorem 4.3 *Given a set P of n points in $\mathbb{R}^3$, and two parameters $0 < \varepsilon < 1$, $0 < p < 1$, we can construct $O\left(\log\frac{1}{p}\right)$ subsets of P, $A_1, \ldots, A_k$, of total size $O\left(\frac{1}{\varepsilon^{3/2}p}\log\frac{1}{\varepsilon p}\right)$, so that, given any halfspace h containing qn points of P, for $q \ge p$, we can find a set A_t that satisfies*

$$\left|\frac{|h\cap A_t|}{|A_t|} - \frac{|h\cap P|}{|P|}\right| \le \varepsilon\frac{|h\cap P|}{|P|}.$$

The time it takes to search for A_t and obtain the count $|h \cap A_t|$ by brute force is $O\left(\frac{1}{\varepsilon^{3/2}q}\log\frac{1}{\varepsilon q}\right)$.

4.2 Higher dimensions

The preceding construction can be generalized to higher dimensions, with some complications. We first introduce the following parameters:

$$\gamma = 1 + \frac{1-\frac{1}{d^*}}{d+1}, \quad \text{where} \quad d^* = \lfloor d/2\rfloor, \quad \text{and} \quad \mu = \frac{2d}{d+1}.$$

Note that, for $d \ge 4$, $\gamma > 1$ and tends to 1 as d increases, and $\mu < 2$ and tends to 2 as d increases.

The analogue of Theorem 4.1 is the following theorem [15].

Theorem 4.4 *Let P be a set of n points in $\mathbb{R}^d$, $d \ge 4$, and let $0 < \varepsilon < 1$, $0 < p < 1$, be as above. Then there exists a set $A \subset P$, of size $O\left(\frac{d^{\mu/2}}{\varepsilon^\mu p^\gamma}\log\frac{d}{\varepsilon p}\right)$, such that, for any halfspace h of weight* at most *pn, we have $\left|\frac{|h\cap A|}{|A|} - \frac{|h\cap P|}{|P|}\right| \le \varepsilon p$, provided that $n = \Omega\left(\frac{d^{\mu/2}}{p^\gamma}\log^{\mu/2}\frac{d}{p}\right)$.*

As in three dimensions, this implies [15]:

Theorem 4.5 *Given a set P of n points in $\mathbb{R}^d$, and two parameters $0 < \varepsilon < 1$, $0 < p < 1$, we can construct $O\left(\log\frac{1}{p}\right)$ subsets of P, $A_1, \ldots, A_k$, of total size $O\left(\frac{d^{\mu/2}}{\varepsilon^\gamma p^\mu}\log^{\mu/2}\frac{1}{\varepsilon p}\right)$, so that, given any halfspace h containing qn points of P, for $q \ge p$, we can find a set A_t that satisfies*

$$\left|\frac{|h\cap A_t|}{|A_t|} - \frac{|h\cap P|}{|P|}\right| \le \varepsilon\frac{|h\cap P|}{|P|}.$$

The time it takes to search for A_t and obtain the count $|h \cap A_t|$ is $O\left(\frac{d^{\mu/2}}{\varepsilon^\gamma q^\mu}\log^{\mu/2}\frac{1}{\varepsilon q}\right)$.

For further discussion of this theorem, see [15].

5. APPROXIMATE HALFSPACE RANGE COUNTING: GENERAL APPROACH

In this section we return to the problem of approximate halfspace range counting, and describe, in somewhat more detailed but still high-level terms, our general technique. Concrete implementations As already mentioned, hereafter we focus on the case of halfspace ranges in $\mathbb{R}^d$, for $d \ge 4$, rather than more general range spaces.

We use Matoušek's shallow partition theorem [19], which yields, for any positive integer parameter $r < n$, a partition of P into $r/2 \le s \le r$ subsets $P_1, P_2, \ldots, P_s$, where, for each i, $n/r \le |P_i| \le 2n/r$, and P_i is enclosed in a simplex Δ_i, such that any hyperplane that bounds an (n/r)-shallow halfspace crosses at most $\mu(r) = O(r^{1-1/\lfloor d/2\rfloor})$ simplices Δ_i. Such a partition can be constructed in time $O(n^{1+\delta})$, for any $\delta > 0$. For $r \le n^\xi$, for a suitable constant $\xi = \xi(d) > 0$, it can be computed in $O(n\log r)$ time.

For a fixed set P and a choice of parameter r at every interior node, the shallow partition theorem induces, in a natural way, a tree $T = T(P)$, called a *shallow partition tree* of P, whose root stores the entire P, and some bounding simplex Δ of P. The root has $s \le r$ children, each storing one of the sets P_i and its bounding simplex Δ_i. The tree is expanded further in the same manner, possibly with different values of r at different nodes, stopping when we reach nodes whose associated sets have size smaller than some specific threshold.

Denote by P_v the subset of P stored at a node v of T, and by r_v the parameter r used when constructing the partition of P_v. Our proposed approximate range counting data structure is, effectively, an augmented shallow partition tree, where we store some additional information at each node v; in the main implementations that we present, this is a relative $(1/r_v, \varepsilon/2)$-approximation A_v of P_v. Querying with a halfspace h proceeds as follows: When visiting a node v, if the boundary of h meets many (more than $\mu(r_v)$) simplices of the set $S_v := \{\Delta_i\}$ of the partition at v, or if h fully contains one of these simplices, it cannot be $(|P_v|/r_v)$-shallow

with respect to P_v, so $|h \cap P_v| > |P_v|/r_v$. Then we answer the approximate range counting query for P_v by counting $h \cap A_v$. Otherwise, we recursively obtain an ε-approximate count at all the children of v whose simplex is crossed by ∂h, and return the sum of the answers, which is easily seen to be an ε-approximate count of $|h \cap P_v|$. See Algorithm 1 for the pseudocode.

Algorithm 1 Pseudocode of our main algorithm

```
1: function ApproxCount(halfspace h, node v of an augmented shallow partition tree, ε)
2:     if v is a leaf node then return LeafNodeApxCount(h,v,ε).
       ▷ S = S_v is the set of simplices associated with children of v.
       ▷ r = r_v is the partition parameter at v.
3:     if ∂h crosses at most μ(r) simplices of S and no simplex is fully contained in h
4:     then                                   ▷ Shallow halfspace, recurse.
5:         answer ← 0.
6:         for all children ξ of v whose bounding simplex is crossed by ∂h do
7:             answer ← answer + ApproxCount(h,ξ,ε).
8:         return answer.
9:     else                                   ▷ Deep halfspace, answer locally.
10:        return DeepApproxCount(h,v,ε).
```

It remains to specify, for each node v, the parameter r_v used at v, the threshold $n_0(\varepsilon)$ for the size of P_v, below which v becomes a leaf, and three subroutines:
The implicit subroutine (that we call SearchSim(h, v)) used in lines 3 and 6 of the algorithm to determine how many, and which, of the simplices of S_v are met by the hyperplane ∂h and whether any of them is contained in h.
A procedure LeafNodeApxCount(h, v, ε) that directly estimates the count for h at a leaf v of the tree.
A procedure DeepApproxCount(h, v, ε) that estimates the count of a deep range h at a node v, using the relative approximation set (or any other appropriate auxiliary structure).

Let $Q_{\text{sim}}(s)$, $Q_{\text{leaf}}(n, \varepsilon)$, and $Q_{\text{deep}}(n, \varepsilon)$ be upper bounds on the running times of these three respective operations, where $r/2 \le s \le r$ is the number of simplices to test against, n is the size of the point set associated with the current node, and ε is the approximation parameter. We obtain the following recurrences for the preprocessing time $T(n, \varepsilon)$, storage $S(n, \varepsilon)$, and query time $Q(n, \varepsilon)$ of our data structure. The parameter $r = r_v$ is the one used at the current node of the tree; in our implementations it is a function of n (and possibly ε). For simplicity, we use $s = r$ in the recurrences, for the maximum possible number of children of a node. We use S_{sim}, T_{sim} (S_{leaf}, T_{leaf} and S_{deep}, T_{deep}) to denote the storage and preprocessing time required by SearchSim (LeafNodeApxCount and DeepApproxCount, respectively). T_{part} is the time needed to construct the partition at v.

$$Q(n,\varepsilon) \le \begin{cases} Q_{\text{sim}}(r) + \max\{Q_{\text{deep}}(n,\varepsilon),\ \mu(r)Q(n/r,\varepsilon)\} \\ Q_{\text{leaf}}(n,\varepsilon), \end{cases}$$

$$S(n,\varepsilon) \le \begin{cases} S_{\text{sim}}(r) + S_{\text{deep}}(n,\varepsilon) + \sum_{i=1}^{r} S(n_i,\varepsilon) \\ S_{\text{leaf}}(n,\varepsilon), \end{cases}$$

$$T(n,\varepsilon) \le \begin{cases} T_{\text{part}}(n,r) + T_{\text{sim}}(r) + T_{\text{deep}}(n,\varepsilon) + \sum_{i=1}^{r} T(n_i,\varepsilon), \\ T_{\text{leaf}}(n,\varepsilon), \end{cases}$$

where each $n_i \le 2n/r$ and $\sum_{i=1}^{r} n_i = n$, and where the first alternative is taken whenever $n > n_0(\varepsilon)$ and the second one otherwise.

6. CONCRETE IMPLEMENTATIONS

There are many ways to choose the parameters r_v, $n_0(\varepsilon)$, and to implement the above three procedures. We present two variants; the first is simpler and more naive (and has slightly poorer performance), and the second is more sophisticated with a slightly better performance. (Roughly speaking, the first implementation has performance comparable with that of the halfspace range *reporting* procedure of [19], whereas the second implementation has performance comparable with that of the halfspace range *emptiness* procedure of [19].)

First implementation. For each node v, put $n_v := |P_v|$. Here we choose $r_v := n_v^{\alpha'}$, for some $0 < \alpha' < \alpha := 1 - 1/\lfloor d/2 \rfloor$, whose concrete choice will be discussed below. We store at v a relative $(1/r_v, \varepsilon/2)$-approximation A_v for P_v, of size $\frac{cr_v}{\varepsilon^2} \log \frac{r_v}{\varepsilon}$ (for some absolute constant $c > 0$), which we obtain by taking a random sample of these many points from P_v. As shown above (see also [12]), such a sample is a relative $(1/r_v, \varepsilon/2)$-approximation of the desired kind with probability at least $1 - 1/r_v^b$, where $b = b(c)$ is linear in c. Later, we boost up the overall success probability, making the failure probability polynomially small in n itself. For now, we simply assume that A_v is indeed an approximation of the required type and size.

In this implementation, we use brute force for two of the three subroutines. We implement SearchSim by simply iterating over all simplices, and selecting those that ∂h crosses, stopping after collecting more than $\mu(r_v)$ of them, or after encountering a simplex that is fully contained in h. The cost is $O(r_v) = O(n_v^{\alpha'})$. We implement LeafNodeApxCount by iterating over P_v and counting $h \cap P_v$ explicitly at the cost of $O(n_v)$.

We implement DeepApproxCount recursively, by calling ApproxCount itself, on an auxiliary data structure constructed for A_v as the input set, with error parameter $\varepsilon/3$, which makes the combined relative error at most ε (for $\varepsilon < 1$).

In order for this implementation to work efficiently, we need to impose some restrictions on the choice of parameters. We require that $|A_v| = \frac{cr_v}{\varepsilon^2} \log \frac{r_v}{\varepsilon} \le \frac{n_v}{k \log^3 \log n_v}$, for some constant k. Intuitively, this requires that the size of A_v be small enough compared to that of P_v. By the choice of r_v, one can verify that this holds when

$$n > n_0(\varepsilon) := \left(\frac{c'}{\varepsilon^2} \log \frac{1}{\varepsilon} \log^3 \log \frac{1}{\varepsilon} \right)^{1/(1-\alpha')}, \qquad (5)$$

for an appropriate multiple c' of c. Our goal is to make the query time satisfy

$$Q(n,\varepsilon) \le F(\varepsilon) n^{\alpha} \log^{\beta} n, \tag{6}$$

for some parameter β and function $F(\varepsilon)$ whose specific choices will be discussed shortly. In particular, we want $Q_{\mathrm{deep}}(n,\varepsilon)$ and $Q_{\mathrm{leaf}}(n,\varepsilon)$ to satisfy this bound. For Q_{leaf} this requires

$$n_0(\varepsilon) \le c'' F(\varepsilon) n_0(\varepsilon)^{\alpha} \log^{\beta} n_0(\varepsilon),$$

where c'' is some constant, which we strengthen slightly by ignoring c'' and the polylogarithmic factor, and by replacing the inequality by an equality. We thus put

$$F(\varepsilon) := n_0(\varepsilon)^{1-\alpha} = \left(\frac{c'}{\varepsilon^2} \log \frac{1}{\varepsilon} \log^3 \log \frac{1}{\varepsilon} \right)^{\frac{1-\alpha}{1-\alpha'}}. \tag{7}$$

Hence $F(\varepsilon)$ is approximately of the form $1/\varepsilon^{\gamma}$, where $\gamma := 2(1-\alpha)/(1-\alpha')$ satisfies $2/\lfloor d/2 \rfloor < \gamma < 2$. Note that γ approaches its upper (resp., lower) bound as α' approaches α (resp., 0).

We can plug the various concrete bounds on Q_{sim}, Q_{deep}, Q_{leaf}, etc., into the recurrences at the end of Section 5, and then show [4] that $Q(n,\varepsilon)$ satisfies Eq. (6). Similarly, the storage bound $S(n,\varepsilon)$ is shown to be $O(n)$, with a constant that is *independent* of ε. Finally, the preprocessing time $T(n,\varepsilon)$ is shown to be $O(n^{1+\delta})$, for any $\delta > 0$, which improves to $O(n \log n)$ if α' is chosen sufficiently small.

We thus have our first main result, with all the ingredients in place, except for the high probability assertion, for which see below.

Theorem 6.1 *We can preprocess a set P of n points in $\mathbb{R}^d$, with a pre-specified error parameter $0 < \varepsilon < 1$, into a data structure of size $O(n)$ (independent of ε), so that, with high probability, for any query halfspace h, we can obtain a relative ε-approximate count of $h \cap P$, in time $O\left(\varepsilon^{-\gamma} n^{\alpha} \log^{\beta} n\right)$, where $\alpha = 1 - 1/\lfloor d/2 \rfloor$, γ can be chosen anywhere in $(2/\lfloor d/2 \rfloor, 2)$, and β depends on γ.*

The data structure can be constructed deterministically, except for the random samplings that produce the various relative approximations. The (worst-case) preprocessing cost is $O(n^{1+\delta})$, for any $\delta > 0$. It reduces to $O(n \log n)$ when γ is chosen sufficiently small.

Second implementation.

A somewhat more careful (and involved) implementation yields the following variant (see [4]).

Theorem 6.2 *We can preprocess a set P of n points in $\mathbb{R}^d$, with a pre-specified error parameter $0 < \varepsilon < 1$, into a data structure of size $O(n)$, independent of ε, so that, with high probability, for any query halfspace h, we can obtain an ε-approximate count of $h \cap P$, in time $O(\varepsilon^{-\gamma} n^{\alpha} \cdot 2^{b \log^* n})$, where $\alpha = 1 - 1/\lfloor d/2 \rfloor$, γ can be chosen anywhere in $(2/\lfloor d/2 \rfloor, 2)$, and b depends on the choice of γ. The preprocessing cost of the algorithm is $O(n^{1+\delta})$, for any $\delta > 0$.*

Ensuring high probability.

So far we have presented the data structures under the assumption that at each node v we have, or can efficiently construct, a relative $(1/r_v, \varepsilon/2)$-approximation of the required size. We can achieve this either by an expensive preprocessing that constructs these sets deterministically (see Theorem 2.4), or draws them at random and *verifies* that they are indeed relative approximations with the appropriate parameters. (Just drawing these sets at random, with the sizes specified above, will not work when analyzed naively, since it only guarantees *constant* failure probability at each node, and there are too many nodes to make sure that all draws are relative ε-approximations with high probability.) Alternatively, increasing the sample size by a factor of $\log n$, would guarantee low failure probability, but this might (slightly) affect the algorithm performance.

We argue that, nevertheless, using such a random sampling approach, with some additional mechanisms, does guarantee high success probability. The intuition is that we can think of the elements of all the relative approximation sets that a query halfspace reaches, as a sequence of independent Bernoulli trials, so that an appropriate weighted sum of their corresponding indicator variables is the approximate count that the algorithm produces. This implies that the errors that the individual relative approximation sets incur tend to cancel each other out, leading to an overall error that is much smaller than the sum of the individual errors. The precise and detailed analysis is given in the full version [4].

With this analysis, the proofs of Theorems 6.1 and 6.2 are now complete.

Discussion.

We conclude the presentation of the basic technique with a few comments.

(1) Notice that there is a sharp discontinuity in the performance of a query in the first implementation, as we reach the leaves of the partition tree. At internal nodes, we effectively ensure that the cost of the approximate counting via the $(1/r_v, \varepsilon/2)$-approximation stored at a node v is roughly n_v^{α}. In contrast, when we reach a leaf, the cost goes up to $\Theta(n_v)$. Quite likely, smoothly interpolating between these two scenarios should refine the dependence of the performance bounds on ε. We leave this as an open problem for further research.

(2) Our technique can be modified to produce a data structure where ε is not known in advance. The idea is to maintain, at each node v of the tree, many relative approximations, and use the one that fits the query; see [4].

Range-minimum queries.

We can apply our technique to design an efficient algorithm for answering range-minimum queries for halfspaces, with respect to a given random permutation of the input points, of the type needed in the approach of Cohen [10] and Kaplan et al. [16, 17], as described above. The only difference is that at each node v of the partition tree, we store the prefix of the first $cr_v \log n$ elements of the random permutation, restricted to P_v. Omitting all further details (see [4]), we obtain (the theorem parallels our first implementation; extending the second implementation can also be done):

Theorem 6.3 *One can preprocess a set P of n points in $\mathbb{R}^d$, and a random permutation π of P, into a linear-size data structure, such that the element of P with minimum rank in π in a query halfspace can be computed in time $O(n^{1-1/\lfloor d/2 \rfloor} \log^{\beta} n)$, for an appropriate constant $\beta = \beta(d)$. The preprocessing cost is $O(n^{1+\delta})$, for any $\delta > 0$, and it improves to $O(n \log n)$ if β is chosen sufficiently large.*

7. REFERENCES

[1] P. Afshani and T. M. Chan, On approximate range counting and depth, *These Proceedings*.

[2] B. Aronov and S. Har-Peled, On approximating the depth and related problems, *Proc. 16th Annu. ACM-SIAM Sympos. Discrete Algo.*, 2005, 886–894.

[3] B. Aronov and S. Har-Peled, On approximating the depth and related problems, manuscript, 2006. Full version of [2], available from `http://www.uiuc.edu/~sariel/papers/04/depth`.

[4] B. Aronov and M. Sharir, Approximate halfspace range counting, in preparation.

[5] S. Arya, T. Malamatos, and D. Mount, Space-time tradeoffs for approximate spherical range counting, *Proc. 16th Annu. ACM-SIAM Sympos. Discrete Algo.*, 2005, 535–544.

[6] S. Arya, T. Malamatos, and D. Mount, The effect of corners on the complexity of approximate range searching, *Proc. 22th Annu. ACM Sympos. Comput. Geom.*, 2006, 11–20.

[7] B. Chazelle and E. Welzl, Quasi-optimal range searching in spaces with finite VC dimension, *Discrete Comput. Geom.* 4 (1989), 467–490.

[8] B. Chazelle, *The Discrepancy Method: Randomness and Complexity*, Cambridge University Press, New York, 2001.

[9] B. Chazelle, The discrepancy method in computational geometry, chapter 44, in *Handbook of Discrete and Computational Geometry*, 2nd Edition, J.E. Goodman and J. O'Rourke, Eds., CRC Press, Boca Raton, 2004, 983–996.

[10] E. Cohen, Size-estimation framework with applications to transitive closure and reachability, *J. Comput. Syst. Sci.* 55 (1997), 441–453.

[11] E. Cohen and H. Kaplan, Spatially-decaying aggregation over a network: model and algorithms, *SIGMOD '04: Proc. 2004 ACM SIGMOD Internat. Conf. on Management of Data*, 2004, 707–718.

[12] E. Cohen, H. Kaplan, Y. Mansour and M. Sharir, Approximations with relative errors in range spaces of finite VC dimension, manuscript, 2006.

[13] D. Haussler, Decision theoretic generalizations of the PAC model for neural nets and other learning applications, *Inf. Comput.* 100 (1992), 78–150.

[14] D. Haussler and E. Welzl, Epsilon nets and simplex range queries, *Discrete Comput. Geom.* 2 (1987), 127–151.

[15] S. Har-Peled and M. Sharir, Relative ε-approximations in geometry, Manuscript, 2006. Available from `http://www.uiuc.edu/~sariel/papers/06/relative`.

[16] H. Kaplan, E. Ramos and M. Sharir, Randomized incremental construction of convex hulls and Voronoi diagrams, and approximate range counting, in preparation.

[17] H. Kaplan and M. Sharir, Randomized incremental construction of three-dimensional convex hulls and planar Voronoi diagrams, and approximate range counting, *Proc. 17th ACM-SIAM Sympos. Discrete Algorithms* (2006), 484–493.

[18] Y. Li, P.M. Long, and A. Srinivasan, Improved bounds on the sample complexity of learning, *J. Comput. Syst. Sci.* 62 (2001), 516–527.

[19] J. Matoušek, Reporting points in halfspaces, *Comput. Geom. Theory Appl.* 2 (1991), 169–186.

[20] J. Matoušek, Efficient partition trees, *Discrete Comput. Geom.* 8 (1992), 315–334.

[21] J. Matoušek, *Geometric Discrepancy*, Algorithms and Combinatorics, Vol. 18, Springer Verlag, Heidelberg, 1999.

[22] J. Matoušek, *Using the Borsuk-Ulam Theorem*, Universitext, Springer-Verlag, Berlin, 2003, Lectures on topological methods in combinatorics and geometry, Written in cooperation with Anders Björner and Günter M. Ziegler.

[23] J. Matoušek, E. Welzl and L. Wernisch, Discrepancy and approximations for bounded VC-dimension, *Combinatorica* 13 (1993), 455–466.

[24] J. Pach and P.K. Agarwal, *Combinatorial Geometry*, Wiley Interscience, New York, 1995.

[25] D. Pollard, Rates of uniform almost-sure convergence for empirical processes indexed by unbounded classes of functions, Manuscript, 1986.

[26] V.N. Vapnik and A. Ya. Chervonenkis, On the uniform convergence of relative frequencies of events to their probabilities, *Theory of Probability and its Applications* 16 (1971), 264–280.

[27] E. Welzl, On spanning trees with low crossing numbers, In *Data Structures and Efficient Algorithms, Final Report on the DFG Special Joint Initiative*, volume 594 of *Lect. Notes in Comp. Sci.*, pages 233–249, Springer-Verlag, 1992.