Algorithms for Weak ε-Nets

Bernard Chazelle* Herbert Edelsbrunner† David Eppstein‡
Michelangelo Grigni§ Leonidas Guibas¶ Micha Sharir‖
Emo Welzl**

MANUSCRIPT

Abstract

In the plane, we can find a weak ε-net for convex sets consisting of $O(\varepsilon^{-2})$ points, in time $O(n\varepsilon^{-2})$. We can determine the smallest ε for which a given planar set is an ε-net in time $O(n^3)$.

In $\mathbb{R}^d$, we can find weak ε-nets of size $O\left(\frac{1}{\varepsilon^d}\log^{O(1)}\frac{1}{\varepsilon}\right)$ in time $O(n\ (1/\varepsilon)^{O(1)})$ (both exponents depending on d).

1 Introduction

The existence and construction of ε-nets has recently generated great interest in computational geometry, and has found wide applications in the design of efficient geometric algorithms. Let us briefly review the basic concepts related to ε-nets in our particular geometric setting. A *range space* is defined to be a pair (S, F), where S is a set of points in in R^d, and F is a family of subsets of R^d. A set in F is known as a *range*. Given the set S, consisting of n input points in R^d, and some collection

*Department of Computer Science, Princeton University
†Computer Science Department, University of Illinois at Urbana-Champaign
‡Department of Information and Computer Science, University of California, Irvine
§Department of Computer Science and Eng., University of California, San Diego
¶Computer Science Department, Stanford University
‖School of Mathematical Sciences, Tel Aviv University and Courant Institute of Mathematical Sciences, New York University
**Institut für Informatik, Freie Universität Berlin

of ranges F, a *(strong) ε-net* for F in S is defined to be a subset $N \subset S$, having the property that if any range in F contains at least εn points of S, then it must contain at least one point of the net N. A fundamental result in this area, due to Haussler and Welzl [6], is that if (S, F) has finite *Vapnik-Chervonenkis dimension*, there always exists an ε-net consisting of only $O((1/\varepsilon)\log(1/\varepsilon))$ points in S. Note that the size of the ε-net depends only on ε and not on n. Many common range spaces, including the cases where ranges consist of halfspaces or simplices in R^d, have this nice property. In fact, for many "geometric" range spaces, ε-nets exist of size only $O(1/\varepsilon)$.

It turns out that ε-nets not only exist, but are very abundant [6]; a random subset of S of the appropriate size will be an ε-net with high probability. Efficient deterministic constructions for ε-nets have also been given for several geometric range spaces, primarily through the work of Matoušek [7, 12, 10, 8, 9, 15, 11, 13, 14]. The ε-nets for these spaces have found numerous uses in situations requiring efficient range query data structures and geometric divide-and-conquer techniques. In these techniques ε-nets are typically used to construct a "balanced" partition of the underlying space, in which each part of space gets its "fair" share of the complexity of the overall problem. A good survey of these ideas is given in the paper by Agarwal [1].

In this paper, we are concerned with range spaces of the form (S, C), where S is a set of points in the $\mathbb{R}^d$, and C is the collection of *all convex subsets of* $\mathbb{R}^d$. Unlike the range spaces defined using halfspaces or simplices, the range spaces defined using convex sets *do not* have finite Vapnik-Chervonenkis dimension. Indeed, as defined above any ε-net of a point set S in convex position must consist of at least $(1-\varepsilon)n$ of the points in S [6]. To avoid this difficulty, Haussler and Welzl introduced the concept of a *weak ε-net*. A weak ε-net is a set N as above, except that there is no longer any requirement that N be a subset of S. This important relaxation of the definition lets us find weak ε-nets for convex sets whose size is also a function of ε only, and not of n. Recently Alon, Bárány, Füredi, and Kleitman [2] showed that in fact $O(\varepsilon^{-2})$ points suffice in the plane, and they gave an $O(n^2)$ time algorithm for finding a weak net of that size. They also provided a faster algorithm, which finds a weak ε-net consisting of $O(\varepsilon^{-4.88})$ points in time $O(n\log(1/\varepsilon))$. Finally, they showed that weak ε-nets for convex sets exist in any higher dimension, of size $O_d(1/\varepsilon^{(d+1)(1-s_d)})$, where $s_d = 1/(4d+1)^{d+1}$. Although their construction in the plane is straightforward, the construction for higher dimensions uses a recent result proved by Živaljević and Vrećika [16] that utilizes advanced techniques from algebraic topology.

Even more recently, we [3] independently discovered the bound for two dimensions and improved the bounds in higher dimensions, using a substantially simpler argument. For any dimension d there exist weak ε-nets of size $O_d(\varepsilon^{-d}\log^{\beta_d}(1/\varepsilon))$ where $\beta_2 = 0$ and $\beta_{j+1} = 2j\beta_j + 1$ for $j \geq 2$, so that $\beta_3 = 1$ and $\beta_d \approx 0.149 \cdot 2^{d-1}(d-1)!$. A naive application of the construction yields an algorithm with time complexity $O(n^d \log 1/\varepsilon)$ for constructing such a weak ε-net.

2 New Results

In this paper we show the following three algorithmic results about weak ε-nets for convex sets: the first two results concern generating and testing weak ε-nets in the plane, and the last result concerns generating weak ε-nets in an arbitrary fixed dimension.

The first result combines benefits of both Alon *et al.* algorithms into a single result: linear time deterministic construction of ε-nets of the best known asymptotic complexity. Our second result lets us test whether a given input is an ε-net; this is important since certain randomized algorithms provide an ε-net with high probability but do not guarantee the correctness of their output. Our third result lets us compute weak ε-nets in arbitrary dimension in time linear in n (assuming d and ε are held constant). Specifically, we have the following three theorems.

- We can find a weak ε-net for convex sets in $S \subset \mathbb{R}^2$ consisting of $O(\varepsilon^{-2})$ points, in time $O(n\,\varepsilon^{-2})$.
- Given $S \subset \mathbb{R}^2$ of size n and another set $N \subset \mathbb{R}^2$, with $|N| < n$, we can find the minimum ε for which N is a weak ε-net for the range space (S, C), in time $O(n^3)$.
- We can find a weak ε-net for convex sets in $S \subset \mathbb{R}^d$ consisting of $O_d(\varepsilon^{-d} \log^{\beta_d}(1/\varepsilon))$ points, in time $O(n(1/\varepsilon)^{O_d(1)})$ (the exponent is $d^{O(d)}$).

Remark: The situation for testing weak ε-nets in $d \geq 3$ dimensions is extremely unclear. For variable d, testing weak ε-nets is NP-hard even for nets consisting of a single point: the problem reduces to the Open Hemisphere problem. But in fact, we do not know of a polynomial-time testing algorithm even in $d = 3$.

3 Linear Time Construction in the Plane

Our $O(\varepsilon^{-2})$-point net is very similar to Alon *et al.* [2], so we outline their construction before going on to ours. Let the point set S be bisected by a vertical line ℓ. Then there are (roughly) $n^2/4$ line segments, determined by pairs of points from S, that cross the line. Suppose some convex set C covers εn points, with at least $\varepsilon n/4$ points on each side of the vertical line ℓ. Then C contains at least $3n^2\varepsilon^2/16$ line segments crossing the portion of ℓ covered by C. If we include in our net $4\varepsilon^{-2}/3$ such crossing points, evenly spaced among the total $n^2/4$ such points, we can guarantee that at least one of them will then be included in C. So any convex set missing our net but covering εn points of S must cover at least $3\varepsilon n/4$ points out of the $n/2$ points on one or the other side of the line ℓ. We take care of such convex sets by finding nets for each side of the line recursively. This gives us a recurrence of the form

$$P(\varepsilon) \leq \frac{4}{3}\varepsilon^{-2} + 2P(\frac{3}{2}\varepsilon),$$

which solves to $O(\varepsilon^{-2})$ points in the entire net.

The time for this algorithm is dominated by choosing the $O(\varepsilon^{-2})$ evenly spaced points among the $O(n^2)$ points where line segments cross ℓ. A naive algorithm for this would take $O(n^2)$ time, which basically gives the bound claimed by Alon *et al.*. It is possible to use parametric search techniques to find these points more quickly; however we get the best time bounds by proceeding as follows. In our application it is not important that we find a set of $4\varepsilon^{-2}/3$ crossing points that are evenly spaced among all such points; instead we will be satisfied by a set of $O(\varepsilon^{-2})$ points, with no gap between points larger than $3n^2\varepsilon^2/16$. Let $\delta = 3\varepsilon^2/16$; then we restate our problem as follows:

(P1) Given a set S of n points, bisected by a vertical line ℓ, choose a set N of $O(1/\delta)$ points on ℓ so that no line segment s of ℓ contains more than δn^2 of the points where line segments in S cross ℓ, unless s contains at least one point in N.

In order to apply geometric duality, we make the problem slightly harder, by ignoring the constraint that ℓ bisect the set vertically, and by considering also crossing points determined by points both on the same side of ℓ:

(P2) Given a set S of n points, and a line ℓ, choose a set N of $O(1/\delta)$ points on ℓ so that no line segment s of ℓ is crossed by more than δn^2 lines determined by pairs of points in S, unless s contains at least one point in N.

We now apply projective duality, to get an equivalent problem stated in terms of line arrangements:

(P3) Given an arrangement of n lines, and a point p, choose a set N of $O(1/\delta)$ lines through p so that no double wedge w with vertex p contains more than δn^2 of the arrangement vertices, unless w also contains at least one line in N.

Since P3 is the projective dual of P2, if we express the input lines and points in projective coordinates, any algorithm for P3 will also solve problem P2 and vice versa, with only the names of the objects changing. Such an algorithm will *a fortiori* also solve P1. Note that each line in N can be determined by a point, together with the given point p. Since P3 is invariant under projective transformations we can assume without loss of generality that p is a point at vertical infinity, so that double wedges are transformed into vertical slabs:

(P4) Given an arrangement A of n lines, choose a set N of $O(1/\delta)$ points so that no vertical slab w contains more than δn^2 of the arrangement vertices, unless w contains at least one point in N.

The problem is now simple enough to solve directly. We use Matoušek's algorithm [9] to find a δ-cutting of A. This is a partition of the plane into $O(\delta^{-2})$ regions; each region is crossed by at most $O(\delta n)$ lines, and hence can contain at most $O(\delta^2 n^2)$ arrangement vertices. By a remark in an earlier paper of Matoušek [10], the cutting can be formed by $O(1/\delta)$ monotone polygonal paths, together with vertical line segments from each path vertex. We sort the vertices by their projection on the horizontal axis, and choose our set N to be a subset of these vertices, spaced every $\Theta(1/\delta)$ positions apart in this sorted order.

Any vertical slab that misses N can therefore cross at most $O(1/\delta)$ regions of the cutting: $O(1/\delta)$ of the regions are crossed by the left boundary of the slab, and a further $O(1/\delta)$ regions have the vertical line segments forming their left boundaries entirely contained in the slab. Therefore any such slab contains at most $O(\delta n^2)$ arrangement vertices. If we choose the constant factors appropriately, the vertices of the cutting thus form a set of $O(1/\delta)$ points meeting the requirements of problem P4. Matoušek's cutting algorithm takes time $O(n\delta^{-1}) = O(n\varepsilon^{-2})$ [9], as does his algorithm for transforming a cutting into the desired form [10].

We summarize our results so far:

Lemma 3.1 *We can find a set of $O(1/\delta)$ points satisfying problem P1, in time $O(n\ \delta^{-1})$.*

Proof: We relax P1 and form the projective dual to transform the problem into P3. We then perform a projective transformation taking the point p to infinity, resulting in a problem of form P4. We solve P4 as described above, and reverse the projective transformation and the projective duality, to transform the solution to P4 into a solution to P1. The transformations all take $O(n)$ time so the total time is bounded by that for solving P4.

Theorem 3.2 *We can find a weak ε-net for convex sets in the plane, consisting of $O(\varepsilon^{-2})$ points, in time $O(n\varepsilon^{-2})$.*

Proof: We carry out the recursion describe earlier in the summary of Alon's result, using Lemma 3.1 to solve the step in which we find $O(\varepsilon^{-2})$ evenly spaced points among the $O(n^2)$ points where line segments cross the bisecting vertical line ℓ, and then calling the algorithm recursively on the sets on each side of the line, using an appropriately larger value of ε in the recursive calls. The time for this can be expressed in the recurrence

$$T(n,\varepsilon) = O(n\varepsilon^{-2}) + 2T(\frac{1}{2}n, \frac{3}{2}\varepsilon),$$

which solves to $O(n\varepsilon^{-2})$ total time.

4 Testing Weak ε-Nets in the Plane

Throughout this section we assume that we are given an input set S, $|S| = n$, together with a purported weak ε-net for convex sets, N, for which $|N| = m < n$. We show here how to test whether N is in fact a weak ε-net if ε is given, or more generally how to compute as output the minimum ε for which N is a weak ε-net.

If N is not an ε-net, there must be some convex region containing $(\varepsilon \cdot n + 1)$ points of S, and no points of N. Our strategy is to find a convex region R such that $|R \cap N| = 0$ and $|R \cap S|$ is as large as possible. We do this by using a dynamic programming technique of Eppstein *et al.* [5].

First, note that we can take R to be the convex hull $CH(R \cap S)$; therefore R is a convex polygon having as its vertices points of S. We can triangulate this polygon by connecting each vertex by a diagonal to the bottommost vertex; our dynamic programming algorithm proceeds by building this triangulation from left to right. For each possible triangle T in such a polygon we find the polygon P_T for which T is rightmost in the triangulation, for which $|P_T \cap N| = 0$, and maximizing $|P \cap S|$ among all such polygons. Since some triangle must be rightmost in the true optimal region R, R can be found by maximizing $|P_T \cap S|$ over all possible triangles T.

More explicitly, assume the points p_1, p_2, $\ldots p_n$ of S are sorted in order from left to right. The outline of our algorithm is as follows. Let $\Delta(i, j, k)$ denote the triangle with vertices p_i, p_j, and p_k, and let $cw(i, j, k)$ be the predicate that $\Delta(i, j, k)$ has those points as vertices in clockwise order. Array $M[i, j, k]$ will be used to store the number $|P_{\Delta(i,j,k)} \cap S|$; the actual polygon can be reconstructed if desired by keeping track of the values of ℓ giving the maxima below.

for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do if** $y_j > y_i$ **then**
    for $k \leftarrow 1$ **to** n **do**
      if $|\mathit{\Delta}(i, j, k) \cap N| \neq 0$ **then** $M[i, j, k] \leftarrow 0$
      else if $y_k > y_i$ **and** $cw(i, j, k)$ **then begin**
        $M[i, j, k] \leftarrow 2$;
        for $\ell \leftarrow 1$ **to** n **do**
          if $y_\ell > y_i$ **and** $cw(i, \ell, j)$ **and** $cw(\ell, j, k)$ **then**
            $M[i, j, k] \leftarrow \max(M[i, j, k], M[i, \ell, j])$;
        $M[i, j, k] \leftarrow M[i, j, k] + |\mathit{\Delta}(i, j, k) \cap S| - 2$;
      end.

The correctness of the algorithm follows from the following facts. First, any polygon constructed as above contains no points of N, and has all points above y_i; this follows by an easy induction. Second, all such polygons are in fact convex: by induction it is formed by attaching triangle $\Delta(i, j, k)$ to a facet ij of another convex polygon, the new angle i is convex since all points on both line segments forming the angle are above y_i, and the new angle $p_\ell p_j p_k$ at j is convex since we tested it explicitly. Third, $M[i, j, k]$ correctly counts the number of points in such a polygon, and gives the maximum over all such polygons. Fourth, the true optimal polygon satisfies all the tests in the algorithm, so it will actually be included in this maximization.

A naive implementation of the algorithm would take time $O(n^4)$. We show below how to reduce this time bound to $O(n^3)$. First, for any triangle $\Delta(i, j, k)$, we need to be able to count the numbers of points in $\Delta(i, j, k) \cap S$ and $\Delta(i, j, k) \cap N$. Both can be accomplished in constant time per triangle if we count, for each line segment ij, the numbers of points in S and N inside the slab with vertical boundaries through i and j, and below ij; all such counts can easily be found in time $O(n^3)$.

The remaining $O(n^4)$ in the time bound comes from the quadruply nested loop, in which for each k we find the value of ℓ maximizing $M[i, j, \ell]$ over all ℓ satisfying $y_\ell > y_i$, $cw(i, \ell, j)$, and $cw(\ell, j, k)$. If not for the latter restriction, this computation would be identical for all different values of k, and the loops could be un-nested. But if we enumerate all values of both k and ℓ together, in clockwise order by the slopes of the undirected lines jk and $j\ell$, starting with the slopes least far clockwise of ij, it will always be the case that whenever we are examining some particular k, the values of ℓ already seen will be exactly those for which $cw(\ell, j, k)$. The modified algorithm is as follows.

for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do if** $y_j > y_i$ **then begin**
    $L \leftarrow 2$;
    for $k \in \{1, 2, \ldots n\}$ in clockwise order **do**
      if $|\Delta(i, j, k) \cap N| = 0$ **and** $y_k > y_i$ **do begin**
        if $cw(i, j, k)$ **then** $M[i, j, k] \leftarrow L + |\Delta(i, j, k) \cap S| - 2$
        else $L \leftarrow \max(L, M[i, \ell, j])$;
      end;
  end.

It is shown in [5] how to perform the clockwise enumerations in $O(n^3)$ total time (the information we need to do this is essentially captured by the dual line arrangement, which can be constructed in $O(n^2)$ time and space), and that $O(n^3)$ bound applies to our algorithm. The space used is $O(n^2)$

Theorem 4.1 *Given sets S and N, we can find the minimum ε for which N is a weak ε-net for convex sets in S, in time $O(n^3)$ and space $O(n^2)$.*

Remark: For this problem the situation in higher dimensions is extremely unclear. As we might expect, for variable d, testing weak ε-nets is NP-hard even for nets consisting of a single point: the problem reduces to the Open Hemisphere problem. But in fact, we do not know of a polynomial-time testing algorithm even in $d = 3$.

5 Constructing Weak ε-Nets in $\mathbb{R}^d$

We first review the construction from [3]. Let S be a given set of n points in $\mathbb{R}^d$; with no loss of generality we assume that the points of S lie in general position. We construct a $(d-1)$-level structure from S, where each level j consists of a collection of one-dimensional trees, each tree corresponding to a $(d-j+1)$-flat of the form $x_1 = a_1, \ldots, x_{j-1} = a_{j-1}$.

At the first level, we have only one tree, and its flat is all of $\mathbb{R}^d$. We project the points of S on the x_1-axis, and denote by S_1 the resulting set. We consider the set of all intervals on the x_1-axis connecting pairs of points in S_1, and construct an *interval tree* over these intervals. This is a binary tree whose root node r corresponds to a point $x_1 = a_r$, such that at most half of the intervals lie fully to the left of a_r and at most half lie fully to its right. The intervals containing a_r are stored at the root, and the left (resp. right) subtree of the interval tree is obtained recursively by applying the same bisection step to the intervals lying fully to the left (resp. to the right) of a_r.

This definition of an interval tree applies to arbitrary collections of intervals on a line. In this first-level structure, though, where the intervals form a complete graph, the same tree can also be constructed by building a balanced binary tree on S_1, and by storing each interval at the unique node of the tree whose left subtree stores one endpoint and its right subtree stores the other endpoint. In what follows we will make use of this alternative representation (this balanced binary tree underlies the earlier $d = 2$ construction).

Assuming $d > 2$, we describe the second level of our structure. Let v be a node of the first-level interval tree, corresponding to a hyperplane f_v of the form $x_1 = a_1$. Let N_v denote the set of segments stored at v: segments with endpoints in S that cross f_v but no ancestor hyperplane. For each segment pq in N_v, let the unordered pair (pq) label the point $pq \cap f_v$. Let K_v denote the resulting collection of points. Consider now the collection of all segments in f_v that connect pairs of points of K_v of the form (pq), (pr) (*i.e.* points whose labels share all but one point of S). We project these segments onto the x_2-axis within f_v, and construct an interval tree for these projected segments. A node of this second-level tree corresponds to a $(d-2)$-flat f of the form $x_1 = a_1, x_2 = a_2$, and the intervals $[(pq), (pr)]$ stored at that node correspond to triangles pqr that are spanned by points of S and intersect f. The second level of our structure consists of one such tree for each node v from the first level tree. It is easily verified that each triangle pqr spanned by S is stored in exactly one second-level node.

We continue in this manner, constructing another level of the structure for each dimension $j \le d-1$. Level j consists of a segment tree for each node v in a tree of level $j-1$. A node v of a $(j-1)$-level interval tree corresponds to a $(d-j+1)$-flat

f_v of the form $x_1 = a_1, \ldots, x_{j-1} = a_{j-1}$, and each segment stored at v corresponds to a $(j-1)$-simplex $p^1p^2\cdots p^j$ spanned by S and intersecting f_v at a point. Let N_v denote the set of all these simplices and K_v denote the set of intersection points of these simplices with f_v, each labeled by the unordered set of vertices from S. We form the collection of segments within f_v that connect pairs of points of K_v of the form $(p^1\cdots p^{j-1}q)$, $(p^1\cdots p^{j-1}r)$, project these segments onto the x_j-axis (within f_v), and construct an interval tree on this set of projected segments. Again, a node u of the resulting tree corresponds to a $(d-j)$-flat f_u within f_v, and each segment that u stores, having the form $[(p^1p^2\cdots p^{j-1}p^j), (p^1p^2\cdots p^{j-1}p^{j+1})]$, corresponds to the j-simplex $p^1p^2\cdots p^{j+1}$, which must intersect f_u by convexity.

Let v be a node in some interval tree of the last level $d-1$. The node v stores a list N_v of $(d-1)$-simplices that cross the line f_v associated with v, and a corresponding list K_v of the intersection points of these simplices with f_v. We sort the list K_v in increasing order of the x_d-coordinates of its points. Let Q denote the union of all lists K_v, over all such nodes v of the last-level interval trees; note Q does not depend on ε. Given ε, we now describe a sampled subset N of Q which will be a weak ε-net.

Let v_1 denote the node of the first-level interval tree in whose substructure v lies, and suppose that v_1 lies at depth $\ell \geq 0$ in the first-level tree. We define a sequence $\{\beta_j\}_{j\geq 2}$ of integers by $\beta_2 = 0$ and $\beta_{j+1} = 2j\beta_j + 1$, for $j \geq 2$. We now put $\varepsilon_v = \frac{1}{2}\left(\frac{3}{4}\right)^{\ell}\varepsilon$, $M_v = \frac{c}{\varepsilon_v^d}\log^{\beta_d}\frac{1}{\varepsilon_v}$, for an appropriate constant c depending on d, and we sample every (n^d/M_v)-th point of K_v. If K_v has fewer than M_v points, we do not sample any point of K_v. Let N be the union of all such samples from Q, over all K_v.

Theorem 5.1 ([3]) *Set N is a weak ε-net for S containing $O\left(\frac{1}{\varepsilon^d}\log^{\beta_d}\frac{1}{\varepsilon}\right)$ points.*

A given simplex occurs at most a constant number of times as a label in the structure, thus the size of the above multilevel structure is $O(n^{d-1})$, dominated by the last-level points in Q. Thus a straightforward implementation of the construction requires $O(n^{d-1})$ time to construct the structure, and $O(n^d\log(1/\varepsilon))$ additional time to sample the list N from Q.

6 Computing Weak ε-Nets in $\mathbb{R}^d$

In this section we develop algorithms for computing a weak ε-net N of the above size, given as input a set S of n points in $\mathbb{R}^d$. Our goal is a deterministic algorithm with running time $O((1/\varepsilon)^{O(1)}\, n)$, where the $O(1)$ depends only on d.

To avoid enumerating the simplices of S, we will first compute a δ-approximation R of S over a family of ranges to be specified, with VC-dimension $d^{O(d)}$, and $\delta \approx \varepsilon^{2d+1}/d^2$. Note that R will have size $r = (1/\varepsilon)^{d^{O(d)}}$, and it may be computed either by random sampling or deterministically in time $O((1/\varepsilon)^{d^{O(d)}}n)$ [4, 9]. `[The deterministic construction depends on having ``shatter'' and ``extension''` `oracles, is it clear that we have those?]` By the methods of the last section, we compute the full multilevel structure for R in time $O((1/\varepsilon)^{d^{O(d)}})$, and sample its level $d-1$ lists as before, but sampling more densely by a constant factor to be determined; thus the size of the resulting sample N is a constant factor larger than the size guaranteed by Theorem 5.1. We claim the resulting sample N is a weak ε-net for the original set S.

In order to specify the ranges that R must approximate, we inductively define two predicates $P_j(a,b,c;p^1,\ldots,p^j)$ and $S_j(a,b,c;p^1,\ldots,p^{j+1})$, that make assertions about which simplices will appear as points and segments stored in interval tree nodes at level j. The vector parameters $a,b,c\in\mathbb{R}^j$ describe the multi-level interval history of some node in level j, and the p^i are points in $\mathbb{R}^d$.

Our intention is that if v is a node in level j of a multilevel structure built for S, then there are parameters a_v,b_v,c_v such that for all $p^1,\ldots,p^j\in S$, the j-simplex $p^1\cdots p^{j+1}$ is as a segment stored at v iff $S_j(a_v,b_v,c_v;p^1,\ldots,p^{j+1})$ is true. Similarly $P_j(a_v,b_v,c_v;p^1,\ldots p^j)$ characterizes the $(j-1)$-simplices that may appear as endpoints of such segments, *i.e.* $(p^1\cdots p^j)$ appears as a point on the x_j-axis lying between upper and lower bounds established by the segment-tree ancestors of v. More precisely, a_j is the x_j-coordinate bisected by v, $[b_j,c_j]$ is the upper and lower bounds on x_j established by ancestors of v in this segment tree, and the values a_k,b_k,c_k for $k<j$ are inherited from the earlier levels of the structure leading to the tree containing v. Note also that P_j ignores a_j.

We define these predicates as follows (in the following, for $a\in\mathbb{R}^j$, let a' denote the vector of its first $j-1$ components, and a hat denotes an omitted element):

$$\begin{aligned}
P_1(a,b,c;p^1) &\equiv b_1<p^1_1 \wedge p^1_1<c_1\\
S_1(a,b,c;p^1,p^2) &\equiv P_1(a,b,c;p^1)\wedge P_1(a,b,c;p^2)\wedge \mathrm{cross}_1(a;p^1,p^2)\\
P_j(a,b,c;p^1,\ldots,p^j) &\equiv S_{j-1}(a',b',c';p^1,\ldots,p^j)\wedge \mathrm{between}_j(a',b_j,c_j;p^1,\ldots,p^j)\\
S_j(a,b,c;p^1,\ldots,p^{j+1}) &\equiv \bigvee_{1\le k<\ell\le j+1}\left(\begin{array}{ll} & P_j(a,b,c;p^1,\ldots,\widehat{p^k},\ldots p^{j+1})\\ \wedge & P_j(a,b,c;p^1,\ldots,\widehat{p^\ell},\ldots p^{j+1})\\ \wedge & \mathrm{cross}_j(a;p^1,\ldots,\widehat{p^k},\ldots,\\ & \qquad\widehat{p^\ell},\ldots,p^{j+1},p^k,p^\ell)\end{array}\right)
\end{aligned}$$

The expression for S_j states that the j-simplex may occur as a segment in $\binom{j+1}{2}$ different ways, depending on which pair of facets appear as its endpoints. The predicate $\mathrm{cross}_j(a;q^1,\ldots,q^{j+1})$ (independent of b and c) determines whether the two $(j-1)$-simplices $q^1\ldots q^j$ and $q^1\ldots q^{j-1}q^{j+1}$, as points in the flat $x_1=a_1,\ \ldots,\ x_{j-1}=a_{j-1}$, lie on opposite sides of $x_j=a_j$. This may be computed as the product of two determinants, and hence is the sign of a polynomial of degree $O(d)$. We also need a similar predicate $\mathrm{between}_j$, which determines whether the $(j-1)$-simplex, as a point in the flat specified by a, falls within the x_j-interval $[b_j,c_j]$.

The above definitions exactly mimic the construction of the multi-level structure; hence given a node v in level j, an appropriate choice of a_v,b_v,c_v will characterize those points and segments stored at node v.

Now we specify the ranges that R must approximate: by fixing a, b, c and all points $p^2, p^3, \ldots$ except p^1, the predicates P_j and S_j define ranges for the variable point p^1. These are the ranges (for all P_j, $j \leq d$, and S_j, $j < d$) which we want R to approximate within δ. The inductive definitions show that these ranges have shatter function exponent (and hence VC-dimension) bounded by $d^{O(d)}$. Note in particular that the P_d predicate characterizes those $(d-1)$-simplices captured by a bottom level line and falling within the x_d-interval $[b_d, c_d]$.

For a set of points X in $\mathbb{R}^d$, let $Pr_X[P_j(a,b,c)]$ denote the probability that a random j-tuple $(q^1, \ldots, q^j)$ of points chosen uniformly from X will satisfy $P_j(a,b,c;q^1,\ldots,q^j)$. In the following, we suppress the a, b, c parameters.

Lemma 6.1 *Given that R is a δ-approximation of S for the ranges given above, and given $j \leq d$ and parameters $a, b, c \in \mathbb{R}^j$, then $|Pr_S[P_j] - Pr_R[P_j]| \leq d \cdot \delta$.*

Proof: Consider the intermediate probabilities $p_\ell = Pr_{S^\ell, R^{j-\ell}}[P_j]$, $0 \leq \ell \leq j$; that is, the probability of $P_j(a,b,c)$ over random j-tuples with the first ℓ elements chosen from S and the remaining $j-\ell$ elements chosen from R. Then $p_0 = Pr_R[P_j]$, $p_j = Pr_S[P_j]$, and it will suffice to show that $|p_\ell - p_{\ell+1}| \leq \delta$.

Fix a choice of $q^1, q^2, \ldots q^\ell \in S$ and $q^{\ell+2}, \ldots q^j \in R$ Consider the remaining variable point $q = q^{\ell+1}$. By the symmetry of P_j, R is a δ-approximation of S on the range of all points q that would satisfy P_j. Thus for this choice of the other parameters, the probability error is at most δ. Summing over all choices, the expected error is still at most δ. □

An entirely analogous statement holds for the S_j predicates.

Given a set T of εn points from S, we want to show some point of N in the convex hull of T. The principle difficulty is that N and the multilevel structure were built from R, which could easily be disjoint from T, so perhaps none of the simplices from T are actually stored in the structure. We say that a j-simplex defined over S is *captured* at a level j segment tree node v if a, b, c defines the history of v, and $S_j(a,b,c)$ is true of the simplex. For a simplex that happens to be defined over points in R, it is captured at v iff it is stored at v.

As a first step, find a subset T_0 of $\varepsilon_0 n$ points evenly split by some node v_1 at depth ℓ in the first level binary tree, where $\varepsilon_0 = \frac{1}{2}(3/4)^\ell \varepsilon$. The argument is almost identical; we traverse down the tree towards the child with more points of T until we find a good split. Only the stopping condition is changed: originally we could be sure to stop because $\frac{1}{4}(3/4)^\ell \varepsilon \leq 2^{-\ell}$, now we only have $2^{-\ell} + \delta$, since R approximates S over P_1 ranges. But δ is so small compared to ε (and hence $2^{-\ell}$) that ℓ could only be larger by one. We will need the coarse bound $\varepsilon_0 > \varepsilon^2$. Note that v_1 captures $\Omega(\varepsilon_0^2 n^2)$ 1-simplices spanned by T_0.

We have just shown the $j = 1$ case of the following inductive claim for all $j < d$: there is a node v_j in the jth level of the multilevel structure for R that captures $\Omega(\varepsilon_0^{j+1} n^{j+1} / \log^{\beta_{j+1}} \frac{1}{\varepsilon_0})$ of the j-simplices spanned by points of T_0.

For the inductive step, assume the claim for $j-1$. The $(j-1)$-simplices captured by v_{j-1} become points when intersected with the flat of v_{j-1}, and we order these points by their x_j coordinate. The points are labeled by unordered j-tuples from T_0. These points induce a graph of segments, where two points are connected iff their labels differ by one component (and hence the segments correspond to j-simplices over T_0). By the inductive hypothesis and the Selection Lemma argument in [3], there exists a value $x_j = a^*$ that stabs a subset $\mathcal{M}$ of these segments, of size $|\mathcal{M}| = \Omega(\varepsilon_0^{j+1} n^{j+1} / \log^{2\beta_j} \frac{1}{\varepsilon_0})$. Let $\mu = |\mathcal{M}| / \binom{n}{j+1}$, the fraction of all j-simplices over S that appear in $\mathcal{M}$.

The value a^* traces a path π down through the jth level segment tree, finally exiting to the left or right of a leaf node. Each segments in $\mathcal{M}$ is either captured by a node on the path π, or it falls out below the leaf.

Consider the j-simplices defined over R and stored as segments in this segment tree, certainly there are no more than $\binom{r}{j+1}$. Since the segment tree halves the number of segments at each step, at most $2^{-\ell}$ fraction of all segments can fall in the left or right subtree below a node of depth ℓ. Apply this observation to the node v' in π at depth $\log(1/\mu) + 2 = \Theta(j \log \frac{1}{\varepsilon_0})$, so that at most $\mu/3$ fraction of all j-simplices over R could possibly fall into the subtree. By the approximation Lemma 6.1 (applied to the S_j range defined by the subtree bounds and $a_j = a^*$), at most $\mu/3 + d\delta$ fraction of all j-simplices over S captured by a^* can fall into the subtree below v'. By our choice of δ and the estimate $\varepsilon_0 > \varepsilon^2$, this is at most $\mu/2$. `[This may require rechecking the leading constant in the definition of` δ`.]` Thus at most half of the segments in $\mathcal{M}$ can fail to be captured by one of these first $O(j \log \frac{1}{\varepsilon_0})$ nodes along π. By the pigeon-hole principle, one of these nodes v_j captures at least $\Omega(|\mathcal{M}| / \log \frac{1}{\varepsilon_0})$ of the segments in $\mathcal{M}$. By the definition of $\beta_j = 2\beta_{j-1} + 1$, we are done with the induction.

At the bottom level the flat corresponding to v_{d-1} is a line parallel to the x_d-axis, and we know that at least $\Omega(\varepsilon_0^d \log^{\beta_d} \frac{1}{\varepsilon_0})$ fraction of all $(d-1)$-simplices defined over S are captured by v_{d-1}, and hence intersect the line. Let $[b_d, c_d]$ be the upper and lower bounds on the intersections. By Lemma 6.1 applied to $P_d(a, b, c)$ (with a', b', c' defined by v_{d-1}) and by the choice of δ, a similar fraction of $(d-1)$-simplices defined over R are stored at v_{d-1} and intersect the flat inside the interval $[b_d, c_d]$. Thus the bottom-level sampling that created N, with a suitably chosen leading constant, is sure to choose at least one of these points inside the interval, and therefore a point inside the convex hull of T.

Acknowledgments

The authors wish to express their gratitude for the generous support and hospitality of the DEC Palo Alto Systems Research Center.

References

[1] P. K. Agarwal. Geometric partitioning and its applications. In J. E. Goodman, R. Pollack, and W. Steiger, editors, *Computational Geometry: Papers from the DIMACS special year*. Amer. Math. Soc., 1991.

[2] N. Alon, I. Bárány, Z. Füredi, and D. Kleitman. Point selections and weak ε-nets for convex hulls. *Combin., Probab. Comput.*, 1(3):189–200, 1992.

[3] B. Chazelle, H. Edelsbrunner, M. Grigni, L. Guibas, and M. Sharir. Improved bounds on weak ϵ-nets for convex sets. In *Proc. 25th Annu. ACM Sympos. Theory Comput. (STOC 93)*, pages 495–504, 1993.

[4] B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. In *Proc. 4th ACM-SIAM Sympos. Discrete Algorithms*, pages 281–290, 1993.

[5] D. Eppstein, M. Overmars, G. Rote, and G. Woeginger. Finding minimum area k-gons. *Discrete Comput. Geom.*, 7:45–58, 1992.

[6] D. Haussler and E. Welzl. Epsilon-nets and simplex range queries. *Discrete Comput. Geom.*, 2:127–151, 1987.

[7] J. Matoušek. Construction of ϵ-nets. *Discrete Comput. Geom.*, 5:427–448, 1990.

[8] J. Matoušek. More on cutting arrangements and spanning trees with low crossing number. Technical Report B-90-2, Fachbereich Mathematik, Freie Universität Berlin, Berlin, 1990.

[9] J. Matoušek. Approximations and optimal geometric divide-and-conquer. In *Proc. 23rd Annu. ACM Sympos. Theory Comput.*, pages 505–511, 1991. Also to appear in *J. Comput. Syst. Sci.*

[10] J. Matoušek. Cutting hyperplane arrangements. *Discrete Comput. Geom.*, 6:385–406, 1991.

[11] J. Matoušek. Reporting points in halfspaces. In *Proc. 32nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 207–215, 1991.

[12] J. Matoušek. Spanning trees with low crossing number. *Informatique Theor. Appl.*, 25:103–123, 1991.

[13] J. Matoušek and O. Schwarzkopf. Linear optimization queries. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 16–25, 1992.

[14] J. Matoušek, R. Seidel, and E. Welzl. How to net a lot with little: small ϵ-nets for disks and halfspaces. In *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, pages 16–22, 1990.

[15] J. Matoušek. Efficient partition trees. *Discrete Comput. Geom.*, 8:315–334, 1992.

[16] R. Živaljević and S. Vrećika. The colored Tverberg problem and complexes of injective functions. *J. Combin. Theory Ser. A*, 61:309–318, 1992.