Linear Approximation of Simple Objects *

Kasturi R. Varadarajan† Pankaj K. Agarwal†

Abstract

Let $P = \{P_1, P_2, \ldots, P_m\}$ be a set of m convex polygons in the plane with a total number of n vertices, and for $1 \le i \le m$, let $w_i \in \mathbb{R}^+$ be a weight associated with P_i. The weighted distance between a line ℓ and a polygon P_i is given by $d(\ell, P_i) = \min_{p \in P_i, q \in \ell} d(p,q).w_i$, where $d(p,q)$ is the Euclidean distance between p and q. We want to compute a line ℓ that minimizes the maximium distance between ℓ and the polygons of P. We present an $O(n\alpha(n)\log^3 n)$ algorithm to compute such a line. We also give an $O(n^{2+\varepsilon})$ time algorithm, where ε is an arbitrarily small positive constant, to solve the three dimensional version of this problem; here, P is a set of convex polytopes in $\mathbb{R}^3$, and we want to compute a plane h that minimizes the maximum weighted distance between h and the polytopes.

1 Introduction

Problem Statement. Let $P = \{P_1, P_2, ..., P_m\}$ be a set of m convex polygons in the plane with a total number of n vertices, and for $1 \le i \le m$, let $w_i \in \mathbb{R}^+$ be a weight associated with P_i. The distance between a line ℓ and a convex polygon P_i, denoted by $d(P_i, \ell)$, is defined to be $\min_{p \in P_i, q \in \ell} d(p,q).w_i$, where $d(p,q)$ is the Euclidean distance between p and q. We want to compute a line ℓ that minimizes the maximium distance between ℓ and the polygons of P. We refer to this problem as the *weighted linear approximation* of convex polygons (or linear approximation for brevity). If all weights are the same, say, 1, we refer to the problem as the unweighted linear approximation problem. We also study the same problem in 3-space too. Given a set P of m convex polytopes with a total of n vertices and a set of weights, we want to compute a plane h such that the maximum weighted distance between h and P is minimized.

Previous Results. The problem of approximating sets of points in the plane is encountered in fields such as statistical analysis, computer vision, pattern recognition, and computer graphics, and is usually referred to as the *linear approximation* or the *linear regression* problem. The problem consists of finding the "best" line approximating a set of points. There are many possibilities for the optimality criterion used. A good survey of algorithms that use a variety of criteria can be found in [17, 18]. For example, we may want to find a line minimizing the maximum orthogonal distance to the points or minimizing the sum these distances. Algorithms solving problems with these or other optimality criteria are discussed in [7, 13, 15, 16, 27]. In some cases, the data points to be approximated are not defined precisely, but are themselves approximated by simple objects such as polygons or circles. Naturally, in such cases, we may still want to find the "best" approximating line. In [24], O'Rourke examined the problem of finding a line consistent with a set of data ranges, i.e., a line intersecting a set of vertical line segments. Morris and Norback [22] gave some characterizations of both the unweighted and weighted versions of the linear approximation of points. Following their basic ideas, Lee and Wu [19] gave an optimal $\Theta(n \log n)$ time algorithm for the unweighted version, and an $O(n^2 \log n)$ algorithm for the weighted version, which was improved by Houle et al. [15] to $O(n \log n)$ (in fact, their algorithm works in higher dimensions as well).

In this paper we consider a generalization of the above problem where we want to approximate a set of convex polygons by a line. Robert and Toussaint [23] gave an $O(n \log n)$ time algorithm for the unweighted version, and an $O(n^2 \log n)$ time algorithm for the weighted version. In this paper, we present an $O(n \log^3 n)$ time algorithm for the weighted version of this problem, thereby improving the previous

Work on this paper has been supported by NSF Grant CCR-93-01259, an NYI award, and matching funds from Xerox Corporation.

Department of Computer Science, Box 90129, Duke University, Durham, NC 27708-0129

known bounds by roughly an order of magnitude. We also give an $O(n^{2+\varepsilon})$ time algorithm for the three dimensional version of the problem, which, as far as we know, is the first nontrivial algorithm.

A problem closely related to the linear approximation of convex polygons is to compute a line transversal of a set of objects in the plane if there exists one; given a set of objects S, a line transversal of S is a line that intersects all objects of S. Atallah and Bajaj [4] gave an $O(n \log n)$ algorithm for computing a transversal of a set of circles, and Edelsbrunner *et al.* [9] gave an $O(n \log n)$ time algorithm to compute a description of all line transversals of a set of line segments. Interested readers are referred to [10, 26] for a survey of known results about transversals.

Parametric Search. Our algorithm is based on Megiddo's parametric search technique [21], which has been successfully used to solve a wide variety of geometric problems [1, 2, 5]. The basic idea behind the parametric search technique is as follows: Suppose we have a problem $\mathcal{P}(r)$ that receives as input n data items and a real parameter r. We want to find a value r^* of r at which the output of $\mathcal{P}(r)$ satisfies certain properties. Suppose we have an efficient sequential algorithm A_s for solving $\mathcal{P}(r)$ at any given r, and that, as a by-product, the algorithm A_s can also determine whether the given r is equal to, less than, or greater than the desired value r^*. Assume, moreover, that the flow of execution of A_s depends on comparisons, each of which involves testing the sign of a low-degree polynomial in r and in the input items.

Megiddo's technique then runs the algorithm A_s "generically," without specifying the value of r, with the intention of simulating its execution at the unknown r^*. Each time a comparison is to be made, the few roots of the associated polynomial (treated as a polynomial in r only) are computed, and we run A_s "off-line" at each of them, thereby determining the location of r^* among these roots, and thus the sign of the polynomial at r^*, i.e., the outcome of the comparison at r^*. If one of the roots is r^*, we stop right away, because we have found the value of r^*, otherwise the execution of the generic A_s is resumed. As we proceed through the execution, each comparison we resolve further constrains the range in which r^* can lie, and we thus obtain a sequence of progressively smaller intervals, each known to contain r^*. It can be shown that the generic algorithm has to make a comparison whose polynomial vanishes at r^* (see [2] for a proof), which will cause the computation to stop at the desired value r^*.

The above procedure works even if, for a given r, A_s can only determine whether r is less than r^* or at least r^* (i.e., cannot distinguish between $r > r^*$ and $r = r^*$); this is the case in our application. In this case, the interval containing r^* is half-closed, of the form $(\alpha, \beta]$; since the algorithm performs a test at r^*, it follows that the upper endpoint of the final interval must be r^*.

The cost of this implicit search is usually dominated by C_sT_s, where C_s is the maximum number of comparisons executed by A_s. Since this bound is usually too high, Megiddo suggests replacing the generic A_s by a parallel algorithm, A_p. If A_p uses Π processors and runs in T_p parallel steps, then each such step involves at most Π *independent* comparisons, that is, each can be carried out without having to know the outcome of the others. We can then compute the roots of all Π polynomials associated with these comparisons, and run a binary search to find the location of r^* among them (using, say, the serial algorithm A_s at each binary step). This requires $O(\Pi + T_s \log \Pi)$ time per parallel step, for a total of $O(\Pi T_p + T_s T_p \log \Pi)$ time, which often results in a saving of nearly an order of magnitude in the running time. Since the parallel algorithm is simulated sequentially, we can use the parallel computation model of Valiant [28].

In our case, r^* is the smallest value of r for which there exists a line ℓ such that $d(P, \ell) \leq r$, and the *decision problem* is to determine, given a set P of m convex polygons in the plane with a total of n vertices, the associated set of weights W and a real number $r \geq 0$, if there is a line ℓ such that $d(P, \ell) \leq r$ (We let $d(P, \ell)$ denote $\max_i d(P_i, \ell)$).

In order to apply Megiddo's technique, we need efficient sequential and parallel algorithms for the decision problem.

2 Geometric Preliminaries

Lower envelopes. Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a collection of n d-variate, possibly partially defined, functions, all algebraic of some constant maximum degree b (and if they are partially defined, the domain of definition of each f_i is also described by a constant number of polynomial equalities and inequalities of maximum degree, say, b, too). The *lower envelope* of $\mathcal{F}$ is defined to be

$$\mathcal{L}_{\mathcal{F}}(\mathbf{x}) = \min f_i(\mathbf{x})$$

where the minimum is taken over all functions of $\mathcal{F}$ that are defined at $\mathbf{x}$. Similarly, one can define the *upper envelope* of $\mathcal{F}$ as

$$\mathcal{U}_{\mathcal{F}}(\mathbf{x}) = \max f_i(\mathbf{x}).$$

The minimization diagram $\mathcal{M}_{\mathcal{F}}$ of $\mathcal{F}$ is the decomposition of $\mathbb{R}^d$ into maximal connected relatively open cells, of any dimension, so that within each cell the same subset of functions appear on the envelope $\mathcal{L}_{\mathcal{F}}$. If the functions of $\mathcal{F}$ are partially defined, we also require that, over each cell c, each of the polynomials defining the domain of definition of any function that attains $\mathcal{L}_{\mathcal{F}}$ over c has a fixed sign. Informally, this means that if a function f attains $\mathcal{L}_{\mathcal{F}}$ over a cell c, then either c is fully contained in the boundary of the domain of f or is disjoint from that boundary. The ***combinatorial complexity*** of $M_{\mathcal{F}}$ and of $\mathcal{L}_{\mathcal{F}}$ is the number of cells of all dimensions in $M_{\mathcal{F}}$. The ***maximization diagram*** $\mathcal{M}'_{\mathcal{F}}$ and its combinatorial complexity are defined in an analogous manner.

For $d = 1$, by the theory of Davenport-Schinzel sequences, the complexity of $\mathcal{L}_{\mathcal{F}}$ (or $\mathcal{U}_{\mathcal{F}}$) is $\lambda_{b+2}(n)$, where $\lambda_t(n)$ is the maximum length of a Davenport-Schinzel sequence of order t, composed of n symbols; see [26] for details on Davenport-Schinzel sequences. It is well known that for a fixed t, $\lambda_t(n)$ is close to linear. The minimization diagarm $\mathcal{M}_{\mathcal{F}}$, and thus the lower envelope, can be represented as the sequence of its vertices sorted from left to right. For $d > 1$, the result of Sharir [25] implies that the complexity of $\mathcal{M}_{\mathcal{F}}$ is $O(n^{d+\varepsilon})$, for any $\varepsilon > 0$. For $d = 2$, $\mathcal{M}_{\mathcal{F}}$ can be computed in $O(n^{2+\varepsilon})$ time, for any $\varepsilon > 0$ [3].

Transversals. Let $\mathcal{S}= \{S_1, S_2, \ldots, S_n\}$ be a collection of n compact convex sets in the plane. A line that intersects all sets of $\mathcal{S}$ is called a ***transversal*** (or a ***stabber***) of $\mathcal{S}$. It is convenient to study transversals by applying a standard form of geometric duality, in which the ***dual*** of a point (a, b) is the line $y = ax + b$ and the dual of a line $y = \alpha x + \beta$ is the point $(-\alpha, \beta)$ (see [8]). We will denote the dual of an object γ by γ^*. Note that this duality is undefined on vertical lines; thus our analysis will handle only non-vertical transversals. For a compact convex set R, we define its ***stabbing region*** R^* to be the set of points dual to the lines that intersect R. The boundary ∂R^* of R^* is the set of points dual to the tangents to R. It is easily checked that R^* is bounded from above by a convex x-monotone curve and from below by a concave x-monotone curve; they will be referred to as the ***upper*** and ***lower arms*** of R^*, respectively. The upper and lower arms can be defined as graphs of totally defined univariate functions. The dual of a point l^* lying on the upper (resp. lower) boundary of R^* is a tangent l to R such that R lies in the closed half-plane lying below (resp. above) l. The ***stabbing region*** (or the ***space of all transversals***) of $\mathcal{S}$ is the intersection $\mathcal{S}^* = \bigcap_{i=1}^{n} S_i^*$. By definition, the lines dual to any point in $\mathcal{S}^*$ are precisely all the non-vertical transversals of $\mathcal{S}$. The definition of stabbing regions for a convex set in $\mathbb{R}^3$, and for a family of convex sets in $\mathbb{R}^3$, can be extended in an obvious manner. We refer the reader to [26] for more details about transversals.

3 Linear Approximation of Convex Polygons in 2-d

Let $P = \{P_1, P_2, ..., P_m\}$ be a set of m convex polygons in the plane with a total of n vertices, W the associated set of weights, and r a non-negative real number. The decision problem for the linear approximation of P is to determine whether there is a line ℓ such that $d(P, \ell) \leq r$.

Let $D(a, b)$ denote the disk of radius b centered at a. Let D_i denote the Minkowski sum $P_i + D(0, r/w_i)$. D_i is a convex region, whose boundary is the outer boundary of the region swept by a disk of radius r/w_i as its center moves around the boundary of the convex polygon P_i. Then the decision problem is equivalent to the following problem: Is there a line ℓ that stabs $\mathcal{D} = \{D_i \mid 1 \leq i \leq n\}$. Let T_i (respectively B_i) denote the upper (respectively lower) arm of D_i^*, the stabbing region of D_i. We let $\mathcal{T} = \{T_i \mid 1 \leq i \leq m\}$ and $\mathcal{B} = \{B_i \mid 1 \leq i \leq m\}$. T_i is a continuous piecewise hyperbolic function, and if n_i is the number of vertices in polygon P_i, T_i is constituted of at most n_i pieces. Let H_i denote the collection of hyperbolic arcs constituting T_i, and let $\mathcal{H} = \bigcup_i H_i$. Note that $\mathcal{H}$ has at most n elements, and the lower envelope $\mathcal{L}_{\mathcal{T}}$ is the same as the lower envelope $\mathcal{L}_{\mathcal{H}}$ of the hyperbolic arcs. The combinatorial complexity of the lower envelope $\mathcal{L}_{\mathcal{H}}$ depends on the maximum number of intersections between two arcs of $\mathcal{H}$. An intersection between two arcs H_i and H_j in the dual plane corresponds to an upper tangent common to two disks in the primal plane. Since two non-identical disks have at most two common upper tangents, we conclude that curves H_i and H_j intersect at most twice. Therefore the combinatorial complexity of $\mathcal{L}_{\mathcal{T}}$ (or $\mathcal{L}_{\mathcal{H}}$) is at most $\lambda_4(n) = O(n2^{\alpha(n)})$ [26]. Using a similar argument, we can conclude that the combinatorial complexity of $\mathcal{U}_{\mathcal{B}}$ is $O(n2^{\alpha(n)})$.

3.1 Sequential algorithm for the decision problem

We first check whether $\mathcal{D}$ has a vertical line transversal. Let I_j denote the projection of the convex region D_j onto the x-axis. $\mathcal{D}$ has a vertical line transversal if and only if $\bigcap_{j=1}^{m} I_j \neq \emptyset$. Since $\bigcap_{j=1}^{m} I_j$ can be computed in $O(n)$ time, we can determine in $O(n)$ time whether $\mathcal{D}$ has a vertical line transversal. If the answer is 'no', we next check whether $\mathcal{D}$ has a non-vertical line transversal, which, by the above discussion is equivalent to checking whether $\mathcal{D}^* \neq \emptyset$. We observe that the stabbing region $\mathcal{D}^* = \bigcap_i D_i^*$ of $\mathcal{D}$ is precisely the set of points that lie above the upper envelope $\mathcal{U}_\mathcal{B}$ and below the lower envelope $\mathcal{L}_\mathcal{T}$.

We construct $\mathcal{L}_\mathcal{T}$ (actually $\mathcal{L}_\mathcal{H}$) and $\mathcal{U}_\mathcal{B}$ using the efficient algorithm of Hershberger in $O(n\alpha(n)\log n)$ time. We determine whether there is any point above $\mathcal{U}_\mathcal{B}$ and below $\mathcal{L}_\mathcal{T}$ as follows: We merge the vertices of the minimization diagram $\mathcal{M}_\mathcal{T}$ and of the maximization diagram $\mathcal{M}'_\mathcal{B}$. Let $b_1, b_2, \ldots, b_k$ be the merged list of vertices. The bounds on the combinatorial complexity of $\mathcal{U}_\mathcal{B}$ and $\mathcal{L}_\mathcal{T}$ imply that $k = O(n2^{\alpha(n)})$. Let $f \in \mathcal{B}$ and $f' \in \mathcal{T}$ denote the functions that appear in $\mathcal{U}_\mathcal{B}$ and $\mathcal{L}_\mathcal{T}$ respectively, in the interval (b_i, b_{i+1}). Arguing as earlier, we can conclude that f and f' intersect in at most two points. Therefore, we can determine in constant time whether there is any point with abcissa in the interval (b_i, b_{i+1}) that lies above the graph of f and below the graph of f'. We walk through all the b_i's in $O(k) = O(n2^{\alpha(n)})$ time. This completes the discussion of our sequential algorithm for the decision problem.

Theorem 3.1 *Given a set P of m convex polygons in the plane with a total of n vertices, the associated set of positive real weights W and a real number $r \geq 0$, we can determine in $O(n\alpha(n)\log n)$ time whether there is a line ℓ such that $d(P,\ell) \leq r$. The algorithm can be modified to return one such line if it exists.*

3.2 A Parallel Algorithm for the Decision Problem

Our parallel algorithm for the decision problem runs along the same lines as its sequential counterpart. We first compute $\bigcap_{j=1}^m I_j$ in $O(\log n)$ time using $O(n)$ processors. If the intersection is empty, we construct $\mathcal{L}_\mathcal{T}$ and $\mathcal{U}_\mathcal{B}$, and determine if there is any point that lies above $\mathcal{U}_\mathcal{B}$ and below $\mathcal{L}_\mathcal{T}$. To construct the envelopes in parallel, we employ the algorithm by Goodrich [11].

Theorem 3.2 (Goodrich) *Given $F = \{f_1, f_2, ..., f_n\}$ be a collection of functions, where $f_i : \mathbb{R} \to \mathbb{R}, 1 \leq i \leq n$, such that every pair of functions has at most k intersections, one can construct the upper envelope of the functions in F in $O(\log n)$ time using $O(\lambda_k(n))$ processors in Valiant's parallel comparison model.*

The theorem implies an algorithm for constructing $\mathcal{U}_\mathcal{B}$ and $\mathcal{L}_\mathcal{T}$ in $O(\log n)$ time using $\lambda_4(n) = O(n2^{\alpha(n)})$ processors. To check whether there is a point that lies above $\mathcal{U}_\mathcal{B}$ and below $\mathcal{L}_\mathcal{T}$, we merge the lists of vertices of $\mathcal{M}_\mathcal{T}$ and $\mathcal{M}'_\mathcal{B}$ (each list has $O(n2^{\alpha(n)})$ elements) as follows. We compute the rank of an element a in each list by binary searching each list with a in $O(\log n)$ time. The ranks of all the elements are found in $O(\log n)$ time in parallel by $O(n2^{\alpha(n)})$ processors. Once the ranks are computed, the merged list can be computed easily. Let $b_1, b_2, ..., b_k$ be the merged list of breakpoints. Let $f \in \mathcal{B}$ and $f' \in \mathcal{T}$ denote the functions that appear in $\mathcal{U}_\mathcal{B}$ and $\mathcal{L}_\mathcal{T}$ respectively, in the interval (b_i, b_{i+1}). We can determine in constant time whether there is any point with abcissa in the interval (b_i, b_{i+1}) above the graph of f and below the graph of f'. We do this in parallel (using $O(n2^{\alpha(n)})$ processors) for all intervals determined by the breakpoints. We conclude the discussion of the parallel algorithm.

Theorem 3.3 *Given a set P of m convex polygons in the plane with a total of m vertices, the associated set of positive real weights W, and a real number $r \geq 0$, one can determine if there is a line ℓ such that $d(P,\ell) \leq r$ in $O(\log n)$ time using $O(n2^{\alpha(n)})$ processors.*

Applying Megiddo's parametric search technique as discussed in Section 1 with the algorithms of Theorems 3.1 and 3.3 immediately implies an $O(n\alpha(n)\log^3 n)$ time algorithm to compute r^*, the smallest value of r for which there is a line ℓ such that $d(P,\ell) \leq r$. We then compute a line ℓ that attains $d(P,\ell) = r^*$ by modifying the sequential algorithm of Theorem 3.1.

Theorem 3.4 *Given a set P of m convex polygons in the plane with a total of n vertices and the associated set of positive real weights, one can compute, in $O(n\alpha(n)\log^3 n)$ time, a line ℓ that minimizes the maximum distance to the polygons of P.*

4 Linear Approximation of Convex Polytopes in 3-d

In this section, we present an algorithm for the linear approximation of convex polytopes in $\mathbb{R}^3$. Given a set $P = \{P_1, \ldots, P_m\}$ of m convex polytopes in $\mathbb{R}^3$, with a total of n vertices, and an associated set $W = \{w_1, \ldots, w_m\}$ of positive real weights, we want to compute a plane h that minimizes $d(h,P)$, the maximum distance between h and the polytopes of P. As in the planar version of this problem, we employ the parametric search technique of Megiddo. The decision problem in this case is: Given a non-negative real number r, determine whether there is a plane h such that $d(h,P_i) \leq r$, for $1 \leq i \leq m$.

Let $B(a,d)$ denote the ball of radius d centred at a. Let D_i denote the Minkowski sum $P_i + B(0, r/w_i)$. Then the decision problem is equivalent to determining whether $\mathcal{D} = \{D_i \mid 1 \leq i \leq m\}$ has a plane transversal. Let T_i (respectively B_i) denote the set of points dual to upper (respectively lower) tangents of D_i. T_i and B_i are xy-monotone surfaces and can be regarded as the graphs of bivariate, piecewise algebraic functions composed of n_i pieces, where n_i is the number of vertices of P_i. Each piece is a partially defined algebraic function of some constant maximum degree b, and the domain of definition of each piece is also bounded by a constant number of algebraic arcs of maximum degree, say, b, too. D_i^*, the stabbing region of $\mathcal{D}_i$, is the region lying between T_i and B_i. Let H_i (respectively G_i) denote the collection of pieces constituting T_i (respectively B_i). Let $\mathcal{T} = \{T_i \mid 1 \leq i \leq m\}$, $\mathcal{B} = \{B_i \mid 1 \leq i \leq m\}$, $\mathcal{H} = \bigcup_i H_i$ and $\mathcal{G} = \bigcup_i G_i$. The stabbing region $\mathcal{D}^*$ of $\mathcal{D}$ is the set of points lying above the upper envelope $\mathcal{U}_\mathcal{B}$ and below the lower envelope $\mathcal{L}_\mathcal{T}$. By a result of Agarwal *et al* [3], the combinatorial complexity of $\mathcal{D}^*$ is $O(n^{2+\varepsilon})$. Note that the lower envelope $\mathcal{L}_\mathcal{T}$ (resp. $\mathcal{U}_\mathcal{B}$) is the same as the lower envelope $\mathcal{L}_\mathcal{H}$ (resp. $\mathcal{U}_\mathcal{G}$), and that $\mathcal{H}$ and $\mathcal{G}$ are collections of $O(n)$ (partially defined) surfaces.

Our sequential algorithm for the decision problem first checks whether $\mathcal{D}$ has a vertical plane (a plane parallel to the z-axis) transversal. Let I_j denote the projection of D_j on to the xy-plane, and let $I = \bigcup_j I_j$. Clearly, $\mathcal{D}$ has a vertical plane transversal if and only if I has a line transversal. Using the sequential algorithm in the last section, we can determine in $O(n\alpha(n)\log n)$ time whether $\mathcal{D}$ has a vertical plane transversal. If the answer is 'no', we proceed to determine whether $\mathcal{D}^*$ is empty.

If the total number of surfaces in $\mathcal{G}$ and $\mathcal{H}$ is less than a (suitably chosen) constant, we check in constant time if there is a point $\mathbf{x} \in \mathbb{R}^2$ such that $\mathcal{U}_\mathcal{G}(\mathbf{x}) \leq \mathcal{L}_\mathcal{H}(\mathbf{x})$ (using, say, the algorithm of [3]). Otherwise, we fix some sufficiently large constant parameter r, and choose a subset E (resp. F) of $ar\log r$ surfaces from $\mathcal{G}$ (resp. $\mathcal{H}$), where a is a constant independent of r. We compute the minimization diagram M_F of F and the maximization diagram M_E' of E. By a result of Sharir [25], the complexity of M_E' and M_F is $O(r^{2+\varepsilon})$. We overlay the two diagrams, and compute the vertical decomposition of the overlay. By the result of [3], the resulting planar subdivision Π also has $O(r^{2+\varepsilon})$ faces, each bounded by at most four edges. Let c be a cell of Π. Let f be the unique function that attains $\mathcal{U}_E$ in c; there may be no such f, in which case we assume that f is $-\infty$ everywhere. Let f' attain $\mathcal{L}_F$ in c; if no such f' exists, we set f' to $+\infty$ everywhere. If the sign of $(f - f')$ is not the same over all points in c, we refine c further, into $O(1)$ cells, so that the sign is invariant over each of the refined cells. Abusing the notation slightly, let c denote one of the refined cells. If $f'(\mathbf{x}) < f(\mathbf{x})$ for points in c, we can conclude that $\mathcal{L}_\mathcal{H}(\mathbf{x}) < \mathcal{U}_\mathcal{G}(\mathbf{x})$ for points in c, and we can discard c. Otherwise, let $\bar{c}$ be the vertical cell

$$\bar{c} = \{(\mathbf{x}, z) \mid \mathbf{x} \in c \text{ and } f'(\mathbf{x}) \leq z \leq f(\mathbf{x})\}$$

If the graph of a function of $\mathcal{H}$ (resp. $\mathcal{G}$) lies below (resp. above) $\bar{c}$, then also $\mathcal{L}_\mathcal{H}(\mathbf{x}) < \mathcal{U}_\mathcal{G}(\mathbf{x})$, and we can discard c, so we assume there is no such function. Let $\mathcal{H}_c \subseteq \mathcal{H}$ (respectively $\mathcal{G}_c \subseteq \mathcal{G}$) denote the functions whose graphs intersect the cylinder $\bar{c}$. $\mathcal{D}^*$ is nonempty over c if and only if there is a point $\mathbf{x} \in c$ such that $\mathcal{U}_{\mathcal{G}_c}(\mathbf{x}) \leq \mathcal{L}_{\mathcal{H}_c}(\mathbf{x})$, so we solve the problem recursively for $\mathcal{H}_c$ and $\mathcal{G}_c$ over c.

By the theory of random sampling [6], $|\mathcal{H}_c|, |\mathcal{G}_c| \leq n/r$. Moreover the sets E and F can be chosen deterministically in $O(n)$ time [20]. Hence, if $T(n)$ denotes the maximum running time, we obtain the following recurrence.

$$T(n) \leq O(r^{2+\varepsilon}).T(n/r) + O(n).$$

This solves to $T(n) = O(n^{2+\delta})$, for any $\delta > \varepsilon$. Hence, we can determine in $O(n^{2+\varepsilon})$ time, for any $\varepsilon > 0$, whether $\mathcal{D}$ has a plane transversal.

The above algorithm can be parallelized using standard techniques [12]. The parallel algorithm runs in time $O(\log^3 n)$ and uses $O(n^{2+\varepsilon})$ processors. We omit further details from this version. Plugging this into the parametric search paradigm, we obtain:

Theorem 4.1 *Given a set of polytopes P with a total of n vertices, and the associated set of weights W, we can compute, in $O(n^{2+\varepsilon})$ time for any $\varepsilon > 0$, a plane h that minimizes the maximum distance to the polytopes of P.*

References

[1] P. K. Agarwal and J. Matousek, Ray shooting and parametric search, *SIAM J. Computing* 22 (1993), 794–806.

[2] P. K. Agarwal and M. Sharir, Planar geometric location problems, *Algorithmica*, 11 (1994), 185–195.

[3] P. K. Agarwal, O. Schwarzkopf, and M. Sharir, The overlay of lower envelopes in three dimensions and its applications, *Proc. 11th Annual Symp. Computational geometry* (1995), to appear.

[4] M. Atallah and C. Bajaj, Efficient algorithms for common transversals, *Inf. Proc. Letters* 25 (1987), 87–91.

[5] B. Chazelle, H. Edelsbrunner, L. Guibas, and M. Sharir, Diameter, width, closest line pair, and parametric searching, *Discrete Comput. Geom.* 10 (1993), 183–196.

[6] K. Clarkson and P. Shor, Applications of random sampling in computational geometry II, *Discrete Comput. Geom.* 4 (1989), 11–30.

[7] N. N. Doroshko, M. M. Korneenko, and N. N. Metelskij, Optimal placement of a line relative to a planar point system, *Institute of Math., Byelorussian Academy of Sciences*, 208 (1984), 1–19.

[8] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer–Verlag, Berlin, 1987.

[9] H. Edelsbrunner, H. A. Maurer, F .P. Preparata, A. L. Rosenberg, E. Welzl, and D. Wood. Stabbing line segments, *BIT* 22 (1981), 274–281.

[10] J. E. Goodman, R. Pollack, and R. Wenger, Geometric transversal theory, in *New Trends in Discrete and Computational Geometry*, Janos Pach (Ed.), Springer–Verlag, Berlin, 1991.

[11] M. T. Goodrich, Using approximation algorithms to design parallel algorithms that may ignore processor allocation, *Proc. 32nd Annu. Symp. on Foundations of Computer Science* (1991), 711–722.

[12] M. T. Goodrich, Geometric partitioning made easier, even in parallel, *Proc. 9th Annu. Symp. on Computational Geometry* (1993), 73–82.

[13] L. J. Guibas, M. Overmars, and J. M. Robert, The exact fitting problem for points, *Proc. 3rd Canadian Conference on Computational Geometry* (1991), 171–174.

[14] J. Hershberger, Finding the upper envelope of n line segments in $O(n \log n)$ time, *Info. Proc. Letters* 33 (1989), 169–174.

[15] M. E. Houle, H. Imai, K. Imai, and J. -M. Robert, Weighted orthogonal linear L_∞-approximation and applications, *Lect. Notes Computer Sci.* 382 (1989), 183–191.

[16] M. E. Houle and G. T. Toussaint, Computing the width of a set, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10 (1988), 761–765.

[17] H. Imai and M. Iri, Polygonal approximations of a curve, in *Computaional Morphology*, D. T. Toussaint (Ed.), North Holland, Amsterdam, 1988.

[18] N. M. Korneenko and H. Martini, Hyperplane approximation and related topics, in *New Trends in Discrete and Computational Geometry*, Janos Pach (Ed.), Springer–Verlag, Berlin, 1991.

[19] D. T. Lee and Y. F. Wu, Geometric complexity of some location problems, *Algorithmica* 1 (1986), 193–212.

[20] J. Matousek, Approximations and optimal geometric divide and conquer, *Proc. 23rd ACM Symp. on Theory of Computing* (1991), 1–10.

[21] N. Megiddo, Applying parallel computation in the design of serial algorithms, *J. ACM* 30 (1983), 852–865.

[22] J. G. Morris and J. P. Norback, Linear facility location—solving extensions of the basic problem, *European J. Oper. Res.* 12 (1983), 90–94.

[23] J. Robert and G. Toussaint, Linear approximation of simple objects, *Computational geometry: Theory and Applications* 4 (1994), 27–52.

[24] J. O'Rourke. An on-line algorithm for fitting straight lines between data ranges, *Communications of the ACM* 24 (1981), 574–578.

[25] M. Sharir, Almost tight upper bounds for lower envelopes in higher dimensions, *Discrete Comput. Geom.* 12 (1994), 327–345.

[26] M. Sharir and P. K. Agarwal, *Davenport-Schnizel Sequences and their Geometric Applications*, Cambridge University Press, Cambridge-New York-Melbourne, 1995.

[27] G. T. Toussaint, On the complexity of approximating polygonal curves in the plane, *Proceedings, IASTED Symp. on Robotics and Automation* (1985), 59–92.

[28] L. Valiant, Parallelism in comparison problems, *SIAM J. Comput.* 4 (1975), 348–355.