Minimizing the error of linear separators on linearly inseparable data

Boris Aronov [a], Delia Garijo [b], Yurai Núñez-Rodríguez [c], David Rappaport [d], Carlos Seara [e,*], Jorge Urrutia [f]

[a] Department of Computer Science and Engineering, Polytechnic Institute of NYU, USA

[b] Dept. de Matemática Aplicada I, Universidad de Sevilla, Spain

[c] Lakes Environmental Software, 60 Bathurst Dr. Unit 6, Waterloo, ON, N2V2A9, Canada

[d] School of Computing, Queen's University, Kingston, Canada

[e] Dept. de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Spain

[f] Instituto de Matemáticas, Universidad Nacional Autónoma de México, Mexico

Keywords: Linearly inseparable
Classifiers
Error minimizers

ABSTRACT

Given linearly inseparable sets R of red points and B of blue points, we consider several measures of how far they are from being separable. Intuitively, given a potential separator ("classifier"), we measure its quality ("error") according to how much work it would take to move the misclassified points across the classifier to yield separated sets. We consider several measures of work and provide algorithms to find linear classifiers that minimize the error under these different measures.

1. Introduction

Current massive data collection methods have provided researchers with a wealth of data, together with the challenge of making sense of it. Partitioning or clustering data as a method of data analysis is an important tool in providing meaning to large amounts of data. Performing this type of analysis is multifaceted, and can range from applications in geography and land use, pattern recognition, medical health studies, economics, detecting similarity between genres of music, and data mining to assist in targeted marketing strategies, to name but a few.

Partitioning data using separators or classifiers to perform cluster analysis on training sets is a standard technique, for example it is used in pattern recognition applications [22]. Thus the problem of determining if two disjoint point sets are separable has been widely studied in the literature. See for instance Megiddo [35] for the seminal fixed dimension linear programming method for linear separability. Elizondo [24] surveys a variety of techniques and discusses the application of linear separability to machine learning. O'Rourke et al. [36] and Boissonnat et al. [7] consider algorithms for circular separability, and Hurtado et al. [30] and Arkin et al. [3] examine a variety of separability criteria.

In some applications the training data may contain some points that have been misclassified resulting in the situation where no natural[1] partition scheme classifies the data. In this case classification is attempted where some amount of error is tolerated. Within that context, Aronov and Har-Peled [4] studied the following problem: given a bicolored point set, find a ball that contains the maximum number of red points without containing any blue points. Cortés et al. [18] address the

* Corresponding author.
E-mail addresses: aronov@poly.edu (B. Aronov), dgarijo@us.es (D. Garijo), yurai@cs.queensu.ca (Y. Núñez-Rodríguez), daver@cs.queensu.ca (D. Rappaport), carlos.seara@upc.edu (C. Seara), urrutia@matem.unam.mx (J. Urrutia).

[1] Here by a *natural partition* we mean a partition given by a simple classifier like disk, square, or a halfplane.

problem of finding two boxes S_R and S_B such that the number of red and blue points in S_R and S_B respectively is maximized, while ignoring the points in $S_R \cap S_B$. Mathematical programming techniques have been used in the operations research community to solve similar problems [5,19,31,40].

In this paper we present algorithms that minimize the error when using a linear separator. Given two linearly inseparable point sets we attempt to find a hyperplane which splits the union of the sets into disjoint subsets in such a way that some *error functions* are minimized. We call such hyperplanes *optimal classifiers*. The notion of *optimality* is left intentionally informal as the precise properties that should be optimized are application dependent. We will examine several different criteria for choosing an optimal classifier. We will proceed on the assumption that the dimension d of the problem is a small constant and be mostly concerned about the asymptotic dependence of the speed of our algorithms on the size n of the point sets.

Let R be a set of r red points and B a set of b blue points in $\mathbb{R}^d$. Let $n := r + b$ be the total number of points and assume that the point sets are *disjoint* and *in general position*, that is, no $d + 1$ of the points lie in the same hyperplane in $\mathbb{R}^d$. We say that R and B are *(linearly) separable* if there exists a *(linear) separator*, which is an oriented hyperplane so that the red points lie to its left and the blue points lie to its right. (Formally, each side of the hyperplane is a closed halfspace delimited by it, so points on the hyperplane are considered to lie on both sides simultaneously.) If there is no separator for R and B, then we say that the sets are *inseparable*.

Let $P = \{p_1, \ldots, p_n\} := R \cup B$. Let h be a hyperplane $x_1a_1 + \cdots + x_da_d = a_0$, and let h^- be the halfplane containing the points $(x_1, \ldots, x_d)$ such that $x_1a_1 + \cdots + x_da_d \leq a_0$, and h^+ be the halfspace that contains the points satisfying $x_1a_1 + \cdots + x_da_d \geq a_0$. We will say that h^- lies to the left of h, while h^+ lies to the right of h. If h were a separator of P, we would have $R \subset h^-$ and $B \subset h^+$. As it is not, it *misclassifies* the red points $R(h) := R \setminus h^-$ and the blue points $B(h) := B \setminus h^+$. We use $\Xi = \Xi(h) := R(h) \cup B(h)$ to denote the set of points misclassified by h. We use $s(h)$ to represent the quality of h as a classifier; it depends on h and $\Xi = \Xi(h)$. Our goal is to find a hyperplane that minimizes the cost under one of the following four measures, where $d(\cdot, \cdot)$ denotes the Euclidean distance between points in $\mathbb{R}^d$ and $d(p, X)$ denotes the Euclidean distance from a point p to a set X:

MinMax: Maximum Euclidean distance from h to a point in Ξ, i.e.,

$$s_\infty(h) := \max_{p \in \Xi(h)} d(p, h) = \max\{\max_{p \in R} d(p, h^-), \max_{p \in B} d(p, h^+)\}.$$

MinSum: Sum of the Euclidean distances from h to points in Ξ, i.e.,

$$s_1(h) := \sum_{p \in \Xi(h)} d(p, h) = \sum_{p \in R} d(p, h^-) + \sum_{p \in B} d(p, h^+).$$

MinSum^2: Sum of squares of the Euclidean distances from h to points in Ξ, i.e.,

$$s_2(h) := \sum_{p \in \Xi(h)} d^2(p, h) = \sum_{p \in R} d^2(p, h^-) + \sum_{p \in B} d^2(p, h^+).$$

MinMis: Just the cardinality of Ξ, i.e., the number of misclassified points,

$$s_0(h) = |R \setminus h^-| + |B \setminus h^+|.$$

We are interested in finding an optimal classifier, which we define to be a halfspace h_{OPT} minimizing the quantity $s(h)$; it is not always unique. Notice that since $d(p, h^\pm)$ is a continuous function of h, so are $s_\infty(h)$, $s_1(h)$, and $s_2(h)$.

We will use a standard duality transform. It maps a point $p \in \mathbb{R}^d$ to a non-vertical hyperplane $p^* \subset \mathbb{R}^d$, and vice versa, that is, it maps a non-vertical hyperplane h to the point h^* such with $(h^*)^* = h$ and $(p^*)^* = p$; moreover p is above h if and only if h^* is above p^*.

Outline of the paper. We present algorithms to find optimal classifiers using the four measures described above. In Section 2 we present solutions for the one-dimensional problem, as this will provide some illuminating intuition for proceeding to problems of higher dimension. We devote Section 3 to describing some crucial observations that relate the separability problems in one dimension to those in higher dimensions. In Sections 4–7 we study each of the measures in arbitrary dimension. The algorithms are based on existing techniques from the computational geometry literature. We show that finding an optimal classifier using the MinMax measure is equivalent to determining the penetration depth between two convex polyhedra, and can therefore be solved using existing methods. For optimizing classifiers using the MinSum, MinSum^2, and MinMis measures we use duality and levels in arrangements to systematically enumerate candidate solutions. The computational complexities of the algorithms are summarized in the following Table 1.

2. One dimension

We first consider the one-dimensional case of our set of problems. The input sets R and B lie on the real line. Then a classifier is a point h. We will assume that h^+ is the half-line $[h, +\infty)$ and h^- is the half-line $(-\infty, h]$; the reverse case is handled by a symmetric argument. For simplicity, we will omit the symmetric cases in the statement of our lemmas. We seek the point (or points) h_{OPT} minimizing $s(h)$.

Notice that $d(p, h^+)$ is convex as a function of h, as is its square ($(p - h)^2$ for $h \geq p$, and 0 for $h < p$); the same holds for $d(p, h^-)$. Since the first three error measures are defined as the maximum, sum, and the sum of squares of these functions

Table 1
Summary of the time complexities.

Dimension	MinMax	MinSum	MinSum^2	MinMis
$d = 1$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n \log n)$
$d = 2$	$\Theta(n \log n)$	$O(n^{4/3} \log^{1+\epsilon} n)$ $O(n^{4/3})^*$	$O(n^2)$	$O(n^2)$
$d = 3$	$O(n^2)$	$O(n^{5/2} \log^6 n)^*$	$O(n^3)$	$O(n^3)$
$d > 4$	$O(n^{\lceil d/2 \rceil})$	$O(n^d)$	$O(n^d)$	$O(n^d)$

* Randomized expected time.

over all $p \in P$, in each case $s(h)$ is a convex function of h. Therefore it attains its minimum at a unique closed interval. In fact, only s_1 may attain its minimum on a non-zero-length interval.

2.1. MinMax

Recall that $s_\infty(h)$ is the pointwise maximum of piecewise-linear convex functions and thus piecewise-linear and convex. It is easily checked that it is nowhere constant, since R and B are inseparable, and hence has a unique minimum.

Observation 1. *The optimal* MinMax *classifier* h_{OPT} *is the mean of the leftmost blue point and the rightmost red point and can be computed in* $\Theta(n)$ *time.*

Indeed, by definition, the cost $s_\infty(h)$ is realized by the misclassified points furthest from h and thus can be reduced by a small change of h in the appropriate direction, unless it is midway between extreme misclassified points, as claimed.

2.2. MinSum

Recall that $s(h) := s_1(h)$ is a sum of n piecewise-linear convex functions and thus piecewise-linear and convex. Therefore it achieves its minimum at a unique point or a closed interval, where it is constant. Specifically, between consecutive points of P, $s(h) = \sum_{p \in R(h)} d(p, h^-) + \sum_{p \in B(h)} d(p, h^+) = \sum_{p \in R(h)} (p - h) + \sum_{p \in B(h)} (h - p)$ is a linear function with slope $-|R(h)| + |B(h)|$. It has breakpoints at points of P. Therefore, we have

Theorem 2. *In one dimension, optimal* MinSum *is achieved at any point with the property that the number of red and blue misclassified points is equal. More precisely,* h_{OPT} *lies in the closed interval between the* r*th and* $(r + 1)$ *st point of* P*, counting from the left, or between the* b*th and* $(b + 1)$ *st point, counting from the right. This interval can be computed in optimal linear time.*[2]

Proof. The first statement follows from previous discussion, while the second can be deduced by counting the number of points to the left of h_{OPT}, when it does not coincide with a point of P: since the number of red points to the right of h_{OPT} is equal to the number of blue points to the left of it, the total number of points to its left is precisely r, as claimed. Since the minimum must be achieved on a *closed* interval, this must be the interval delimited by the rth and $(r + 1)$st points from the left; such an interval always exists, as $-|R(h)| + |B(h)|$ starts at $-r$, ends at b and shrinks by exactly one every time h crosses a point of P. Since the total number of points is $n = r + b$, the claim follows. The desired interval can be computed by using a linear-time select algorithm [6]. □

2.3. MinSum^2

Uniqueness of the optimum follows from our previous observations. Indeed, $d^2(p, h^+)$ and $d^2(p, h^-)$ are both convex, continuous, everywhere differentiable functions; each is composed of a quadratic and strictly convex portion and an identically zero portion. The sum of n such functions is convex. Moreover, its minimum can be attained along a non-zero-length interval only if all the constituent functions are zero within it, which is not possible for inseparable point sets.

To find the unique minimum, consider a candidate separator h. Then MinSum^2 error is given by

$$\begin{aligned} s_2(h) &= \sum_{p \in R} d^2(p, h^-) + \sum_{p \in B} d^2(p, h^+) = \sum_{p \in R(h)} d^2(p, h) + \sum_{p \in B(h)} d^2(p, h) \\ &= \sum_{p \in \Xi(h)} (p - h)^2 = h^2 \cdot |\Xi(h)| - 2h \cdot \sum_{p \in \Xi(h)} p + \sum_{p \in \Xi(h)} p^2, \end{aligned}$$

which is a piecewise quadratic function. Put $\Sigma(h) := \sum_{p \in \Xi(h)} p$. By previous discussion, $s_2(h)$ is strictly convex and differentiable everywhere, hence its minimum value must occur in that interval between consecutive points of P where

[2] A result similar to Theorem 2 is proved by Simon [42] but in the context of the support vector machines and the algorithmic learning theory.

$h_{\text{OPT}} := \Sigma(h)/|\Xi(h)|$ occurs within the interval; this value has a geometric interpretation: h_{OPT} is the arithmetic mean (i.e., the centroid) of the misclassified points.

Thus it remains to explain how to find this unique minimum. An $O(n \log n)$ algorithm is clear: after sorting P, we compute $|\Xi(-\infty)|$ and $\Sigma(-\infty)$, and then incrementally update them, maintaining $\Sigma(h)$ and $\Xi(h)$, and evaluating $\Sigma(h)/|\Xi(h)|$ for every interval between consecutive points of P, until we find the unique interval containing the local (and, therefore, global) minimum. In fact, the optimum can be identified in linear time by a prune-and-search procedure; refer to algorithm 1, where Σ represents the sum of the coordinates of the points we know will be misclassified in the optimal solutions and N represents the number of such points. Its correctness follows from the convexity of $s_2(h)$ and the above discussion. Its running time is linear, as it obeys a recurrence of the form $T(n) \leq cn + T(n/2)$. Thus we have shown

Theorem 3. *The one-dimensional* MinSum^2 *problem has a unique solution which can be found in optimal linear time.*

```
Input: Inseparable sets $R$ of red points and $B$ of blue points on a line, with a total of $n$ points.
Output: The value $h_{\text{OPT}}$ at which the MinSum$^2$ error measure $s_2(h)$ is minimized.
Initialization: $\Sigma \leftarrow 0$; $N \leftarrow 0$;
repeat
    Determine $p$ and $q$, the points that straddle the median rank in $P$;
    $\Sigma_B \leftarrow$ the sum of the coordinates of the blue points to the left of $p$ in $P$;
    $\Sigma_R \leftarrow$ the sum of the coordinates of the red points to the right of $q$ in $P$;
    $N_B \leftarrow$ the number of blue points to the left of $p$ in $P$;
    $N_R \leftarrow$ the number of red points to the right of $q$ in $P$;
    $h \leftarrow \frac{\Sigma_B+\Sigma_R+\Sigma}{N_B+N_R+N}$;
    switch $h$ do
        case $p < h < q$
            return $h$;
        end
        case $h < p$
            $P \leftarrow$ subset of $P$ to the left of $h$;
            $\Sigma \leftarrow \Sigma + \Sigma_R$;
            $N \leftarrow N + N_R$;
        end
        case $h > q$
            $P \leftarrow$ subset of $P$ to the right of $h$;
            $\Sigma \leftarrow \Sigma + \Sigma_B$;
            $N \leftarrow N + N_B$;
        end
    end
until ;
```

Algorithm 1: AVERAGEOFMISCLASSIFIED.

2.4. MinMis

Consider inseparable point sets R and B. A different way to achieve separability is by removing misclassified points. The MinMis problem for $R \cup B$ is equivalent to computing an optimal classifier for B and R that minimizes the number of misclassified points (see [25] for the problem in two dimensions).

In order to compute a classifier h_{OPT} yielding the minimum number of misclassified points, i.e., the classifier h_{OPT} that minimizes $s := s_0(h) = |\Xi(h)|$, we sort the points in $O(n \log n)$ time, obtaining a linear number of intervals delimited by consecutive points. Any point in an interval gives the same value of s. Scanning the points left to right, while maintaining the number of misclassified points of either color, one can determine the value of s in each interval, and therefore the minimum value, in linear time. We thus obtain an overall $O(n \log n)$ time algorithm for computing the optimal classifier (either a point or an interval).

Next we see that this algorithm is optimal. Consider the following ε-distance problem for points on a line [3]:

The ε-distance problem. Given a set of n points $x_1, \ldots, x_n$ on a line and a real value $\varepsilon > 0$, decide whether $|x_i - x_j| > \varepsilon$, for all $i, j \in \{1, \ldots, n\}$, $i \neq j$.

The ε-distance problem has an $\Omega(n \log n)$-time lower bound in the algebraic computation tree model [3]. Now we reduce the ε-distance problem to our MinMis problem as follows.

We are given $x_1, \ldots, x_n$ and $\varepsilon > 0$. For each x_i we add $r_i = x_i - \varepsilon/2$ to R and $b_i = x_i + \varepsilon/2$ to B. In addition, we create $10n$ red points to the left of all points considered so far and $10n$ blue points to the right of all of them. This can be easily done in linear time. This ensures that the left side of the classifier is considered red and the right is blue, for the optimal classifier. Refer to Fig. 1.

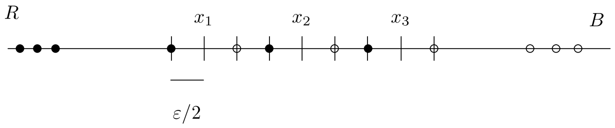

Fig. 1. Lower bound construction.

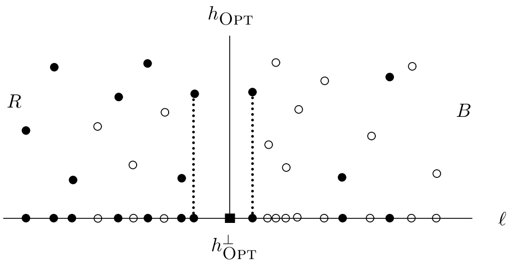

Fig. 2. Projecting an optimal solution.

Now, if $|x_i - x_j| > \varepsilon$, for all $i, j \in \{1, \ldots, n\}$, $i \neq j$, then the red and blue points coming from points x_i alternate along the line: red, followed by blue, followed by red, etc. Thus for a separator h lying between these points, the number of misclassified points oscillates between n and $n - 1$. In particular, it is easy to check that the number of misclassified points is at least $n - 1$ for any position of h and the minimum is $n - 1$.

On the other hand, if there exist i and $j \neq i$, such that $|x_i - x_j| \leq \varepsilon$, then there is at least one point common to the intervals $[r_i, b_i]$ and $[r_j, b_j]$, for $i \neq j$. Such a point misclassifies no more than $n - 2$ points. Thus we have proved

Theorem 4. *The one-dimensional* MinMis *problem can be solved in $O(n \log n)$ time and this is the best possible in the algebraic computation tree model.*

3. From one to higher dimensions

Before proceeding with the higher-dimensional versions of our problem, we make the following simple but crucial observation which follows from the fact that signed Euclidean distances to a hyperplane are preserved under an orthogonal projection to a line orthogonal to the hyperplane (refer to Fig. 2):

Observation 5. *Let h_{OPT} be an optimal classifier for inseparable sets $R, B \subset \mathbb{R}^d$, $d > 1$, for any of our error measures. Let ℓ be the line perpendicular to h_{OPT} and passing through the origin, and let $R^{\perp}$, $B^{\perp}$, and $h^{\perp}_{\mathrm{OPT}}$ be the respective orthogonal projections of R, B, and h_{OPT} to ℓ. Then $h^{\perp}_{\mathrm{OPT}}$ is an optimal classifier for the one-dimensional problem $R^{\perp}, B^{\perp}$ for the same error measure.*

We find the following general approach to obtain an optimal classifier useful for several different error measures. Given a non-vertical candidate classifier hyperplane h, we aim to place red points *to the left of* it and blue points *to the right of* it (vertical classifiers can often be handled by an extension of the following discussion, or directly, as a problem of finding an optimal classifier in one lower dimension). Classifiers with red points *to the right of* them and blue points *to the left* are handled by a symmetric argument.

Given a set $P = B \cup R$ of points, and a candidate classifier h, the exact analytical form of the error measure $s(h)$ depends on the set $\Xi(h)$ of misclassified points, which in turn is determined by the way in which h partitions P; in fact, for all measures but s_∞ the analytical form is completely determined by this bipartition. We now consider the situation in the dual: let $\mathcal{A} := \mathcal{A}(P^*)$ be the *arrangement* of the planes dual to points of P [23]. The various bipartitions of P by h correspond precisely to the various cells of $\mathcal{A}$ that may contain the point h^* dual to h. As we will see in following sections, the analytical form of $s(h)$ for MinSum and MinSum^2 is not only completely determined by the cell C containing h^*, but can also (1) be updated from cell to neighboring cell in constant time and (2) be used to compute $\arg\min_{h^* \in C} s(C)$ in constant time, under certain assumptions on our model of computation; see below for details. An analogous statement holds for MinMis, with "cells" replaced by "faces of any dimension", as this error measure is not continuous. This implies

Theorem 6. *Let R and B be inseparable points sets in $\mathbb{R}^d$, $d > 1$. An optimal classifier according to* MinSum, MinSum^2, *or* MinMis *error measure can be computed in $O(n^d)$ time.*

We devote the next section to MinMax, which requires separate treatment. We further discuss the remaining measures and related existing work in Sections 5–7.

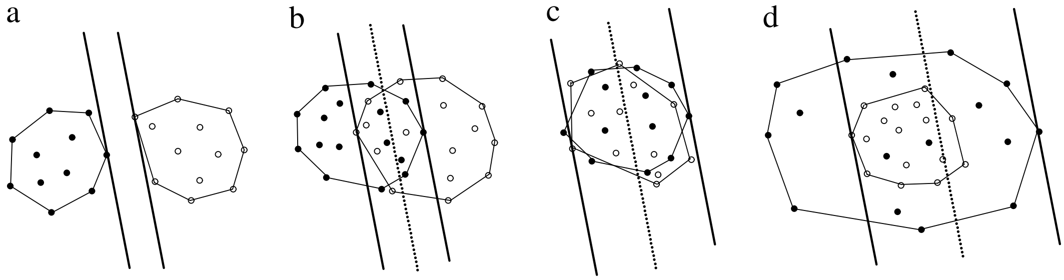

Fig. 3. Antipodal pairs from CH(B) and CH(R) are shown in four representative configurations.

4. Higher dimensions: MinMax

Recall that we have assumed that R and B are inseparable, so that the convex hulls CH(R) and CH(B) properly intersect. Combining Observations 1 and 5, we notice that an optimal classifier h in any fixed direction, for the MinMax measure, occurs half-way between the left supporting hyperplane of B and the right supporting hyperplane of R parallel to h. The error $s(h)$ is precisely half the distance between these hyperplanes. Hence minimizing $s(h)$ is equivalent to minimizing this distance, over all orientations of h. It is not difficult to see that the smallest such distance, over all possible orientations of h, is precisely the minimum distance by which one needs to translate CH(R) to separate it from CH(B). This quantity has been studied in the past, under the names *intersection depth* and *penetration depth* [2,8,15,21,32,33].

In the light of the previous discussion, the two-dimensional version of the problem can be solved in linear time by the *rotating-calipers method* [44], once the convex hulls of R and B have been computed. (Recall that the convex hull of a set of n points in two dimensions can be computed optimally in $O(n \log n)$ time, using $O(n)$ space [23].) See Fig. 3 for an illustration. Thus we have

Theorem 7. *The two-dimensional* MinMax *problem can be solved in* $O(n \log n)$ *time, using* $O(n)$ *space. This cannot be improved in the algebraic computation tree model.*

To show that the algorithm is worst-case optimal, we use a reduction from the Max–Gap problem for points on the first quadrant of the unit circle which is known to have an $\Omega(n \log n)$ lower bound in the algebraic computation tree model [3]. An instance of Max–Gap for points on the first quadrant of the unit circle is a set of points Z, together with the question: What is the maximum Euclidean distance between consecutive points?

The reduction is as follows. Let a set Z of n points on the open first quadrant of the unit circle be an instance of Max–Gap. Put $R_1 := Z = \{r_1, \dots, r_n\}$, where r_i are numbered in their x-order (which is not given). Reflect R_1 through the origin to obtain a set $B_1 = \{b_1, \dots, b_n\}$ of blue points in the third quadrant, as illustrated in Fig. 4. Construct three additional red points r'_i and symmetrically located blue points b'_i; refer to Fig. 4. We draw vertical and horizontal lines at distance 2 from the origin. The point r'_2 lies at the bottom left corner of the resulting 4×4 square; r'_1 is chosen on the left edge of the square so that $r_1 r'_1$ is tangent to the circle. The remaining additional points are constructed analogously. Put $R := R_1 \cup \{r'_1, r'_2, r'_3\}$ and $B := B_1 \cup \{b'_1, b'_2, b'_3\}$.

Now consider the resulting MinMax optimization problem for R and B. Observe that the smallest distance between parallel support lines of CH(R) and of CH(B) occurs when the lines pass through points that give the Max–Gap of Z, as indicated in the figure.

Thus the solution of the MinMax problem would yield a line h from which one can, in linear time, identify Max–Gap in Z, completing the proof.

In three dimensions, the MinMax problem (also known as penetration depth) can be solved by examining all pairs of potential contacts made by two supporting planes with opposite orientations, one for CH(R) and one for CH(B). (Recall that the convex hull of a set of n points in three dimensions can be computed optimally in $O(n \log n)$ time.) The problem with this approach is that the number m of such pairs of contacts is quadratic in the worst case, as in the width problem [29]. Any algorithm that evaluates all such pairs of contacts will run in worst-case time $\Omega(n^2)$. By using the techniques developed by Houle and Toussaint [29] we can obtain an optimal approximate MinMax separator in $O(m + n \log n)$ time, but this is not the best possible. There exists extensive literature on width computation and penetration depth, as mentioned above. In particular, in [2] it was shown how to compute the penetration depth in expected time $O(n^{3/2+\delta})$, for any $\delta > 0$. (In fact, the expected running time of the algorithm is actually $O(m^{1/2+\delta} n^{1/2} + n^{1+\delta})$, so it will run significantly faster when $m \ll n^2$.)

As for the width problem, for $d \geq 4$, an $O(n^{\lceil d/2 \rceil})$ time algorithm can be achieved by realizing the solution space as a convex polytope in $\mathbb{R}^{d+1}$, applying an optimal halfspace intersection algorithm, triangulating the resulting set, and explicitly optimizing $s_\infty(h)$ function over each simplex separately [10,16]. (Recall that the convex hull of a set of n points, or the intersection of n halfspaces, in any fixed dimension $d \geq 4$ can be computed optimally in $O(n^{\lceil d/2 \rceil})$ time [14].)

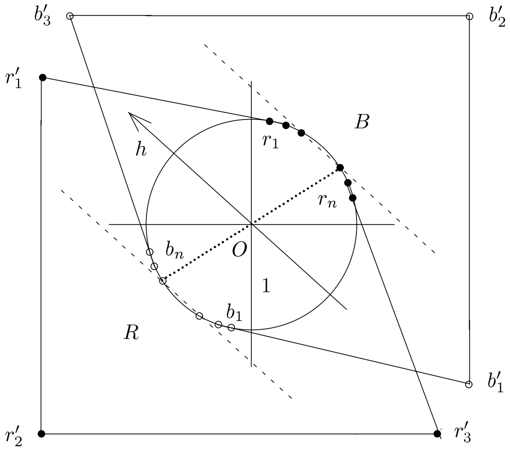

Fig. 4. The construction of the sets R and B from Z is shown illustrating the lower bound argument.

5. Higher dimensions: MinSum

As outlined at the end of Section 3, one can find the optimal classifier for the MinSum measure by effectively examining all candidate classifiers h, or equivalently, enumerating all possible placements of the point h^* dual to h in the arrangement $\mathcal{A}$. In this section we explain the details of this process and simplify it a great deal by proving the following theorem, which implies that only the vertices of the dual arrangement $\mathcal{A}$ need to be examined.

Theorem 8. *Let R and B be inseparable points sets in $\mathbb{R}^d$, $d > 1$. Then there is an optimal classifier h_{OPT} of R and B that contains d affinely independent points of $R \cup B$. Equivalently, h^*_{OPT} lies at a vertex of $\mathcal{A}$. The vertex must belong to a cell of the r-level of $\mathcal{A}$.*

Proof. We will make a slight notational adjustment, just for the duration of this section. We will be discussing ways in which a hyperplane h partitions a point set P. This is unambiguous as long as h does not pass through any of the points. In the dual, as long as h^* stays off the hyperplanes of P^*, there is a clear notion of which hyperplanes lie below it and which lie above. Starting with a point h^* in an open cell c of $\mathcal{A}$, consider the set $\Xi(c) = \Xi(h)$ of misclassified points. Now fix this set $\Xi(c)$ and let h^* vary over the *closed* cell $\bar{c}$: in the following discussion we treat just the points of $\Xi(c)$ as misclassified, and none other. This varies from our original definition in that some points contained in h will now be considered misclassified. This does not affect our measure of error, as the contribution of a point on h to $s(h)$ is zero, whether or not it is considered misclassified.

The effect of this adjustment in the dual is as follows: for a generic point $h^* \in c$, we determine which hyperplanes of P^* lie above and which below, and then extend this convention to points h^* lying on the boundary of c.

Recall that translating a candidate classifier h parallel to itself corresponds to moving h^* along a line parallel to the x_d-axis. Applying Theorem 2 and Observation 5 to the best classifier in this family of hyperplanes, we conclude that there must be precisely r hyperplanes above h^* and precisely b hyperplanes below it, i.e., h^* must lie in a (closed) cell on the r-level of $\mathcal{A}$. (A full-dimensional cell lies on the *r-level* of $\mathcal{A}$, or is an *r-level cell*, when precisely r hyperplanes pass below it.) Additionally, putting $\rho = \rho(h) := |R(h)|$ and $\beta = \beta(h) := |B(h)|$, we must have $\rho = \beta$ for an optimal classifier. We observe that

$$\begin{aligned} s(h) &= \sum_{p \in R(h)} d(p,h) + \sum_{p \in B(h)} d(p,h) = \rho d(h, C(R(h))) + \beta d(h, C(B(h))) \\ &= \rho \left(d(h, C(R(h))) + d(h, C(B(h))) \right), \end{aligned}$$

where we have used $C(\cdot)$ to denote the centroid of a set. Fix a closed r-level cell $\bar{c}$. Since (with our adjusted convention) ρ is a constant over $\bar{c}$, to minimize $s(h)$ over $\bar{c}$, it is sufficient to minimize the expression $H(h) := d(h, C(R(h))) + d(h, C(B(h)))$ over $\bar{c}$. We now argue that this latter expression can attain its minimum only at a vertex of $\bar{c}$.

In short, the function we are minimizing is the sum of the distances to centroids of the red and the blue misclassified points, which by definition lie on the opposite sides of h. If there were no restrictions on positioning the hyperplane h, the function would achieve its minimum value of zero when h passed through the two centroids. Minimizing $H(h)$, while constraining h^* to lie in $\bar{c}$, is equivalent to restricting our attention to those hyperplanes h that separate $\text{CH}(R(h))$ and $\text{CH}(B(h))$. We argue that among all such hyperplanes h, none can minimize $s(h)$ without passing through d (affinely independent) points of P. Indeed, $H(h) = |s| \sin \alpha$, where α is the angle between h and the segment $s := \overline{C(R(h))C(B(h))}$ and $|s|$ is the length of s; $H(h)$ cannot achieve its minimum of zero, since the endpoints of s lie on opposite sides of h and having s lie within h would imply that R and B are separable, contradicting our assumptions.

We first argue that h must be a separating tangent of $\text{CH}(R(h))$ and $\text{CH}(B(h))$. If h strictly separates the two sets, it can be rotated (say, around $s \cap h$) to decrease α and thus h. Hence h must touch at least one of the sets. If h misses the other set, it can be shifted parallel to itself without affecting $H(h)$ to strictly separate the two sets, yielding a contradiction, as above. Thus h is a tangent to the two sets and passes through at least two of their vertices.

As long as the affine dimension of $h \cap P$ is less than $d-1$, we can rotate h around $h \cap P$. It is easy to check that at least one "direction" of this rotation reduces α and thus $H(h)$. (For specificity, pick any $d-2$-flat π containing $h \cap P$ and rotate h around it: there is a one-dimensional family of hyperplanes containing π, so this rotation is well-defined. By assumption, if rotated by a sufficiently small amount in either direction, h continues to be an inner tangent of $\mathrm{CH}(R(h))$ and $\mathrm{CH}(B(h))$. In at least one direction, however, the angle α decreases. Hence the original h could not minimize $H(h)$, as claimed.)

Therefore, as long as h contains fewer than d points of P, there is at least one direction in which it can be infinitesimally rotated around the points $h \cap P$, while maintaining tangency to $\mathrm{CH}(R(h))$ and $\mathrm{CH}(B(h))$ and reducing α. Thus any such choice of a hyperplane cannot be a minimum, for a given bipartition. In other words, the affine hull of $h_{\mathrm{OPT}} \cap P$, for an optimal classifier h_{OPT}, must have dimension $d-1$, or h^*_{OPT} must be a vertex of $\mathcal{A}$, as claimed. □

5.1. MinSum *for* $d = 2$

Let h be a candidate classifier line for R and B according to the MinSum criterion, with h^+ (h^-) denoting the closed halfplane to the right (left) of h.

Let h: $ax + y + e = 0$. Denote by p_x, p_y the coordinates of a point p. The contribution of $p \in R(h)$ to $s(h) := s_1(h)$ is

$$\frac{ap_x + p_y + e}{\sqrt{a^2+1}},$$

while the contribution of $p \in B(h)$ is given by a similar expression, with a negative sign. Summing the contributions of all points and using the fact that $|B(h)| = |R(h)|$ (see proof of Theorem 8), we obtain

$$s(h) = s(a) = \frac{A_1 a + A_2}{\sqrt{a^2+1}},$$

where $A_1 = A_1(h) := \sum_{p\in R(h)} p_x - \sum_{p\in B(h)} p_x$ and $A_2 = A_2(h) := \sum_{p\in R(h)} p_y - \sum_{p\in B(h)} p_y$; A_1 and A_2 depend only on the bipartition of P by h and not on the precise placement of h; the function, for a fixed Ξ, depends only on a, the slope of h.

As noted above, the optimum must be achieved at a vertex of an r-level cell in $\mathcal{A}$ (see Fig. 5). Tight bounds on the maximum complexity (i.e., number of edges and vertices) of the r-level cells in an arrangement of n lines are not known—determining the order of magnitude of this quantity as a function of n is a long-standing open problem in discrete geometry. For $r = \Theta(n)$ it is known to be $ne^{\Omega(\sqrt{\log n})}$ [43] and $O(n^{4/3})$ [20]. There is extensive literature for constructing levels in line arrangements. The best known deterministic algorithm is due to Chan [9] and runs in $O(n^{4/3}\log^{1+\varepsilon} n)$ time and $O(n)$ space. Chan [9,13] presented a randomized algorithm that guarantees $O(n^{4/3})$ expected time for constructing the r-level in an arrangement of n lines in the plane; the bounds improve somewhat if $r \ll n$.

Once the r-level cells have been computed, the remaining computations can be carried out in time proportional to the size of the level. Hence we have a deterministic $O(n^{4/3}\log^{1+\varepsilon} n)$ time algorithm and a randomized $O(n^{4/3})$ expected running time algorithm for finding the set of all optimal classifiers minimizing the MinSum error measure in the plane.

Theorem 9. *The two-dimensional* MinSum *problem can be solved deterministically in time $O(n^{4/3}\log^{1+\varepsilon} n)$ for an arbitrarily small constant $\varepsilon > 0$ or in $O(n^{4/3})$ expected time.*

5.2. MinSum *for* $d \geq 3$

The foregoing discussion extends to three and higher dimensions. We illustrate the calculations in three dimensions. Let h: $ax + ey + z + f = 0$ be a plane with normal vector $(a, e, 1)$. Let $p = (p_x, p_y, p_z) \in P$, then $d(p, h)$ is given by

$$d(p,h) = \pm\frac{ap_x + ep_y + p_z + f}{\sqrt{a^2+e^2+1}},$$

with the sign chosen according to whether $p \in h^+$ or $p \in h^-$. Since, by Theorem 2 and Observation 5, $|B(h)| = |R(h)|$, we have

$$s(h) = \frac{aA_1 + eA_2 + A_3}{\sqrt{a^2+e^2+1}},$$

where $A_1 = A_1(h) := \sum_{p\in R(h)} p_x - \sum_{p\in B(h)} p_x$, $A_2 = A_2(h) := \sum_{p\in R(h)} p_y - \sum_{p\in B(h)} p_y$, and $A_3 = A_3(h) := \sum_{p\in R(h)} p_z - \sum_{p\in B(h)} p_z$. These values are constants for a fixed bipartition, i.e., for h^* in a fixed cell of $\mathcal{A}$. Hence, these quantities and the exact analytic expressions for $s(h)$ can be maintained in constant time, when moving from a level-r cell to an adjacent level-r cell.

The total complexity of the cells on the r-level is the number of their vertices, edges, and faces. Under general position assumptions, it is proportional to the number of vertices of the cells involved. The number of vertices of the r-level is at

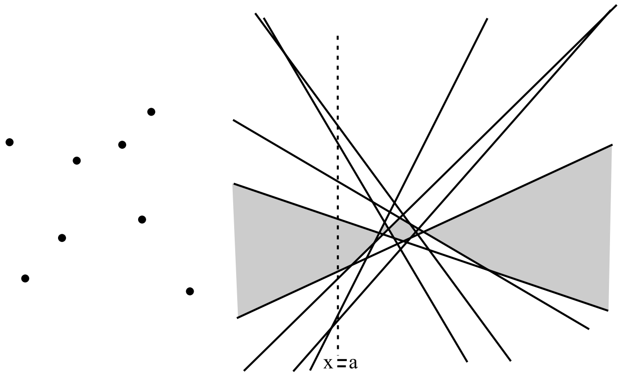

Fig. 5. A set of 8 points in the plane (left). The level-4 cells in the dual arrangement of the 8 lines (right).

most $O(nr^{3/2})$ [41]; the exact maximum complexity of the r-level is a long-standing open problem in discrete geometry. Chan [12] gives an $O(n\log n + nr^{3/2}\log^6 r)$ expected time algorithm for constructing the r-level in an arrangement of n planes. As before, given the set of r-level cells, we traverse these cells in a, say, depth-first-search order of the graph of their adjacencies, going from cell to neighboring cell c, updating the quantities $A_1(c)$, $A_2(c)$ and $A_3(c)$ in constant time and obtaining the exact closed-form equation for the function $s(c)$ for the current cell. By Theorem 8, evaluating $s(h)$ on all vertices of r-level cells is sufficient to locate the optimal classifier(s). Each such evaluation is done in constant time, apart from some initialization cost, so the running time is dominated by the complexity of computing the r-level.

Theorem 10. *The three-dimensional* MinSum *problem can be solved in* $O(n^{5/2}\log^6 n)$ *expected time.*

In higher dimensions, the same approach still applies. Namely the optimum is achieved by h dual to a vertex of an r-level cell of $\mathcal{A}$. Thus it is sufficient to evaluate the function at these vertices. This can be done in constant time per vertex, after some linear-time set-up. The bottleneck again is computing the said vertices.

Again, determining the order of magnitude of the maximum number of such vertices is a long-standing open problem in discrete geometry. It is asymptotically the same as the complexity of the r-level. For dimension $d \geq 4$, the best known upper bound for the size of the r-level is only slightly better than the $O\left(n^{\lfloor d/2\rfloor} r^{\lceil d/2\rceil}\right)$ [16]. More specifically, it is $O(n^{d-\alpha_d})$ for a very small constant $\alpha_d = 1/(4d-3)^d$. As Agarwal et al. [1] observed, the bound can be made sensitive to r, namely $O(n^{\lfloor d/2\rfloor} r^{\lceil d/2\rceil-\alpha_d})$. Matoušek et al. [34] give an $O(n^{4-2/45})$ upper bound for $d = 4$.

For an arbitrary fixed dimension d, the r-level in an arrangement of n hyperplanes in $\mathbb{R}^d$ can be constructed deterministically (see Chan [13]) in time

$$O\left(n^{\lfloor d/2\rfloor} r^{\lceil d/2\rceil}\left(\frac{\log n}{\log r}\right)^{O(1)}\right).$$

To summarize, we have proven

Theorem 11. *The d-dimensional* MinSum *problem can be solved deterministically in time*

$$O\left(n^{\lfloor d/2\rfloor} r^{\lceil d/2\rceil}\left(\frac{\log n}{\log r}\right)^{O(1)}\right) = O(n^d).$$

6. Higher dimensions: MinSum^2

We proceed to implement the plan outlined at the end of Section 3 for MinSum^2. Namely, we consider the dual arrangement $\mathcal{A}$ and evaluate $s_2(h)$ for h^* ranging over all cells c of $\mathcal{A}$. This corresponds to fixing the sets $R(h) = R(c)$ and $B(h) = B(c)$ and therefore the expression for $s_2(h)$ in terms of the coordinates of h^*. The function is a quadratic expression whose minimum can be computed in constant time, in constant dimension; this assumes the ability to compute roots of a system of $O(d)$ equations in d unknowns, which is not an uncommon assumption in computational geometry; in $d = 2$ the minima can be computed explicitly in radicals. We fill in some of the details below.

6.1. MinSum^2 *for* $d = 2$

Let h_{OPT}: $ax + y + e = 0$ be the optimal classifier line according to the MinSum^2 criterion. The squared distance between a point p and the line h is given by

$$d^2(p,h) = \frac{(ap_x + p_y + e)^2}{a^2+1}.$$

Thus

$$s(h) = s_2(h) = \sum_{p \in \Xi(h)} \frac{(ap_x + p_y + e)^2}{a^2+1}$$
$$= \frac{A_1 a^2 + A_2 + A_3 e^2 + 2A_4 a + 2A_5 ae + 2A_6 e}{a^2+1},$$

where $A_1 = A_1(h) := \sum p_x^2$, $A_2 = A_2(h) := \sum p_y^2$, $A_3 = A_3(h) := |\Xi(h)|$, $A_4 = A_4(h) := \sum p_x p_y$, $A_5 = A_5(h) := \sum p_x$, and $A_6 = A_6(h) := \sum p_y$, which are constants for a given partition; all the summations are over points $p \in \Xi(h)$. Thus, $s(h)$ only depends on a and e, so we write $s(h) = s(a, e)$. To identify the minimum of $s(a, e)$, we set its partial derivatives to zero.

$$\frac{\partial s(a,e)}{\partial a} = \frac{a^2(-2A_5 e - 2A_4) - 2a(A_3 e^2 + 2A_6 e + A_2 - A_1) + 2(A_5 e + A_4)}{(a^2+1)^2} = 0$$
$$\frac{\partial s(a,e)}{\partial e} = \frac{2A_3 e + 2A_5 a + 2A_6}{a^2+1} = 0.$$

These conditions can be rewritten as

$$e = A'_1 a + A'_2, \qquad A'_3 a^2 + A'_4 a + A'_5 = 0,$$

for coefficients A'_i that can be expressed explicitly in terms of A_j's. The system can be solved for (a, e), yielding at most two candidate points at which $s(a, e)$ may achieve its minimum in the interior of the current cell c; the points are discarded if they are found not to lie in c. To know which one is the minimum we evaluate the function at both points. Since the minimum may occur along an edge of $\mathcal{A}$, we must repeat this process for every such edge, expressing $s(\cdot, \cdot)$ as a function of position of h^* along the edge and computing its minimum. Finally, we also evaluate $s(\cdot, \cdot)$ at every vertex of $\mathcal{A}$.

To summarize, we compute the precise analytic form of the function $s_2(h)$ in each face (of every dimension) of the dual arrangement $\mathcal{A}$ and explicitly compute its minima. As the coefficients of the function can be updated in constant time from one face of the arrangement to its neighbor, a single traversal of the arrangement allows us to find the global minimum in $O(n^2)$ time.

Theorem 12. *The two-dimensional* MinSum^2 *problem can be solved in $O(n^2)$ time.*

Consider $s(\hat{a}, \cdot)$ as univariate function on a vertical line $a = \hat{a}$ in the dual plane. This corresponds to fixing the slope of a candidate classifier at $\hat{a}$ and varying its vertical position. From Observation 5 and the discussion in Section 2.3 we conclude that $s(\hat{a}, \cdot)$ has a unique minimum and its position varies continuously with $\hat{a}$. The minimum traces an a-monotone curve λ through $\mathcal{A}$, which here means a curve that meets every line $a =$ constant in precisely one point. Let X denote the number of intersections of λ with the lines of P^*. We can use the algorithm of Har-Peled [27] for "walking in arrangements" to compute these faces, provided we can follow the curve from face to face (which is possible by using explicit computation of the minimum's position as a function of $\hat{a}$, in a fixed cell, as outlined above) in expected time $O((X+n)\alpha(n)\log n)$, where $\alpha(\cdot)$ is the extremely slowly growing inverse Ackermann function. (Note that roughly comparable, though slightly larger deterministic running times can be obtained by using deterministic dynamic convex hull (or, equivalently, dynamic halfplane intersection) algorithms [37,38].)

Theorem 13. *The two-dimensional* MinSum^2 *problem can be solved in expected time $O((n+X)\alpha(n)\log n)$, where X is the number of candidate classifier lines h with the property that* (*a*) *h passes through one of the given points and* (*b*) *h is the optimal classifier among all the lines parallel to it.*

Does λ visit $\Theta(n^2)$ cells of $\mathcal{A}$, in the worst case? It would be interesting to determine the worst-case asymptotic behavior of the quantity X as a function of n. Can it really reach quadratic?

6.2. MinSum^2 *for $d \geq 3$*

The previous discussion generalizes to any dimension. We construct the dual arrangement $\mathcal{A}$ in $\Theta(n^d)$ time and compute an explicit description of the function $s_2(h)$, for all h^* in a given face Δ, by traversing the entire arrangement. Over a fixed face Δ, the function is always a ratio of two quadratic functions, with the same denominator. The function can be minimized over Δ in constant time, assuming we can solve systems of d equations in d unknowns, of degree at most three, in constant time. Repeating the calculation for every face Δ, we obtain the optimal classifier h_{OPT} in $O(n^d)$ time.

Theorem 14. *The d-dimensional* MinSum^2 *problem can be solved in $O(n^d)$ time, assuming systems of d polynomial equations in d unknowns and degree at most three can be solved in constant time.*

7. Higher dimensions: MinMis

We denote by k_{OPT} the smallest achievable number of misclassified points.

Again, we view the error measure $s_0(h)$ as a function, with h^* ranging over the dual arrangement $\mathcal{A}$. It is constant over any face of $\mathcal{A}$ and changes in easy to compute ways from face f to an adjacent face. Being the number of misclassified points, it is equal to the number of red hyperplanes strictly below f plus the number of blue hyperplanes strictly above f. This quantity can clearly be maintained in constant time per face, by traversing the entire arrangement, say in a depth-first manner, yielding an $O(n^d)$ time algorithm.

Theorem 15. *The d-dimensional* MinMis *problem can be solved in $O(n^d)$ time.*

We now discuss alternative approaches, reformulations, previous related work, and hardness arguments. Houle [28] gave an $O(n^2)$ time algorithm for this problem in the plane; in fact, the problem is 3SUM-hard [26]. Indeed, since a point p being correctly classified translates to h^* lying in the appropriate (closed) halfspace bounded by p^*, our problem is equivalent to finding the "deepest" point (i.e., a point contained in the maximum number of halfspaces) in an arrangement of n halfspaces. The two-dimensional version of this problem is known to be 3SUM-hard [26]. (A reduction that produces a less degenerate arrangement, corresponding to disjoint sets of red and blue points in the primal, can be carried out along the lines of the argument in [4], where the more general problem of finding the maximum depth in a disk arrangement is shown to be 3SUM-hard.)

An $O(nk \log^2 n)$ time algorithm to compute the space of all classifiers misclassifying up to k points in the plane, for a given k (or, equivalently, finding all feasible points with up to k constraints removed; see below) was presented in [17]. A different approach was taken in [25] to compute k_{OPT} in $O(nk_{\text{OPT}} \log k_{\text{OPT}} + n \log n)$ time.

An equivalent restatement of the dual problem is to consider the set of the (closed dual) halfspaces corresponding to correctly classified points and ask the following question: what is the minimum number k, so that if k of these halfspaces are eliminated, the remaining halfspaces have a common intersection? Phrased differently, given an infeasible linear program, what is the minimum number of constraints that need to be removed to make it feasible? The above question is closely related to the *linear programming with violations* problem [11,39]: given a set of n linear constraints, an integer $k < n/2$, and a linear function $f(\cdot)$ to maximize, find the point x that attains the largest value $f(x)$, while satisfying all but at most k of the given constraints, or reports that no such point exists. This problem has been extensively studied. Several papers on linear programming with violations directly find the minimum number k_{OPT} of constraints that need to be removed to ensure feasibility. For example, Chan [11] presents a randomized algorithm that runs in $O((n + k_{\text{OPT}}^2) \log n)$ expected time in the plane and $O(n \log n + k_{\text{OPT}}^{11/4} n^{1/4} \log^c n)$ expected time in $d = 3, 4$.

Another approach to the problem is to use approximation, as exact solutions, especially when the optimal value k_{OPT} is comparable to n, seem expensive. In [4], a number of algorithms are constructed for approximating the depth of the deepest point in a variety of circumstances. In particular, an approximate solution to the linear programming with violators problem is described, giving an $O(n \log(\varepsilon^{-1} \log n) + (\varepsilon^{-1} \log n)^{O(1)})$ expected time algorithm (for $d = 2, 3$) and $O(n(\varepsilon^{-2} \log n)^{d+1})$ expected time algorithm, for $d > 3$, which will, with high probability, find a hyperplane that misclassifies at most $(1+\varepsilon)k_{\text{OPT}}$ points. Recall that the best hyperplane misclassifies k_{OPT} points.

8. Conclusions

We have described a collection of related problems and solve them using a collection of old, new, and borrowed techniques. The algorithms produce exact solutions, but suffer from the fact that the computational complexity grows sharply with the dimension of the problem. The interesting open question that remains unresolved is to develop algorithms to find optimal classifiers with computational complexity that is less sensitive to the dimension.

Acknowledgments

We thank Sariel Har-Peled for his expert advice on walking in arrangements and looking for depth, not necessarily at the same time. We thank the two anonymous referees for their suggestions and comments.

The first author was partially supported by grant No. 2006/194 from the U.S.-Israel Binational Science Foundation, NSA MSP Grant H98230-06-1-0016, and NSF Grant CCF-08-30691. The second author was partially supported by project MTM2008-05866-C03-01. The third author was partially supported by project NSERC Postdoctoral Fellow, IRDF. The fourth author was partially supported by an NSERC of Canada Discovery Grant. The fifth author was partially supported by projects MTM2009-07242, Gen Cat DGR2009GR1040, and ESF EUROCORES programme EuroGIGA-ComPoSe IP04-MICINN Project EUI-EURC-2011-4306. The sixth author's research was supported in part by MTM2006-03909 (Spain) and SEP-CONACYT of México, Proyecto 80268.

References

[1] P.K. Agarwal, B. Aronov, T.M. Chan, M. Sharir, On levels in arrangements of lines, segments, planes, and triangles, Discrete Comput. Geom. 19 (1998) 315–331.

[2] P.K. Agarwal, L.J. Guibas, S. Har-Peled, A. Rabinovitch, M. Sharir, Computing the penetration depth of two convex polytopes in 3D, Nordic J. Comput. 7 (2000) 227–240.

[3] E.M. Arkin, F. Hurtado, J.S.B. Mitchell, C. Seara, S.S. Skiena, Some lower bounds on geometric separability problems, Internat. J. Comput. Geom. Appl. 16 (2006) 1–26.

[4] B. Aronov, S. Har-Peled, On approximating the depth and related problems, SIAM J. Comput. 38 (2008) 899–921.

[5] K.P. Bennett, O.L. Mangasarian, Robust linear programming discrimination of two linearly inseparable sets, Optim. Methods Softw. 1 (1) (1992) 23–34.

[6] M. Blum, R.W. Floyd, V. Pratt, R. Rivest, R. Tarjan, Time bounds for selection, J. Comput. System Sci. 7 (1973) 448–461.

[7] J.D. Boissonnat, J. Czyzowicz, O. Devillers, J. Urrutia, M. Yvinec, Computing largest circles separating two sets of segments, Internat. J. Comput. Geom. Appl. 10 (1) (2000) 41–53.

[8] S.A. Cameron, R.K. Culley, Determining the minimum translational distance between two convex polyhedra. In: Proc. Int. Conf. on Robotics and Automation, 1986, pp. 591–596.

[9] T.M. Chan, Remarks on the k-level algorithms in the plane, Manuscript, 1999.

[10] T.M. Chan, Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus, Internat. J. Comput. Geom. Appl. 12 (2002) 67–85.

[11] T.M. Chan, Low-dimensional linear programming with violations, SIAM J. Comput. 34 (2005) 879–893.

[12] T.M. Chan, A dynamic data structure for 3-D convex hulls and 2-D nearest neighbor queries, J. ACM 57 (3) (2010) Article 16.

[13] T.M. Chan, Random sampling, halfspace range reporting, and construction of $(\leq k)$-levels in three dimensions, SIAM J. Comput. 30 (2000) 561–575.

[14] B. Chazelle, An optimal convex hull algorithm in any fixed dimension, Discrete Comput. Geom. 10 (1993) 377–409.

[15] K.L. Clarkson, New applications of random sampling in computational geometry, Discrete Comput. Geom. 2 (1987) 195–222.

[16] K.L. Clarkson, P.W. Shor, Applications of random sampling in computational geometry, II, Discrete Comput. Geom. 4 (1989) 387–421.

[17] R. Cole, M. Sharir, C.K. Yap, On k-hulls and related problems, SIAM J. Comput. 16 (1987) 61–77.

[18] C. Cortés, J.M. Díaz-Báñez, P. Pérez-Lantero, C. Seara, J. Urrutia, I. Ventura, Bichromatic separability with two boxes: a general approach, Journal of Algorithms, Algorithms in Cognition, Informatics and Logic 64 (2–3) (2009) 79–88.

[19] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, 2000.

[20] T.K. Dey, Improved bounds on planar k-sets and k-levels, Discrete Comput. Geom. 19 (1998) 373–382.

[21] D. Dobkin, J. Hershberger, D. Kirkpatrick, S. Suri, Computing the intersection-depth of polyhedra, Algorithmica 9 (1993) 518–533.

[22] R. Duda, P. Hart, D. Stork, Pattern Classification, John Wiley and Sons, Inc., 2001.

[23] H. Edelsbrunner, Algorithms in Combinatorial Geometry, in: EATCS Series Monographs in Theoretical Computer Science, vol. 10, Springer, 1987.

[24] D. Elizondo, Searching for linearly separable subsets using the class of linear separability method, in: IEEE International Joint Conference on Neural Networks, 2004, pp. 955–960.

[25] H. Everett, J.-M. Robert, M. van Kreveld, An optimal algorithm for the $(\leq k)$-levels, with applications to separation and transversal problems, Internat. J. Comput. Geom. Appl. 6 (1996) 247–261.

[26] A. Gajentaan, M.H. Overmars, On a class of $O(n^2)$ problems in computational geometry, Comput. Geom. Theory Appl. 5 (1995) 165–185.

[27] S. Har-Peled, Taking a walk in a planar arrangement, SIAM J. Comput. 30 (2000) 1341–1367.

[28] M.E. Houle, Algorithms for weak and wide separation of sets, Discrete Appl. Math. 45 (1993) 139–159.

[29] M.E. Houle, G.T. Toussaint, Computing the width of a set, IEEE Trans. Pattern Anal. Mach. Intell. 10 (1988) 761–765.

[30] F. Hurtado, M. Noy, P.A. Ramos, C. Seara, Separating objects in the plane by wedges and strips, Discrete Appl. Math. 109 (2000) 109–138.

[31] A. Karam, G. Caporossi, P. Hansen, Arbitrary-norm hyperplane separation by variable neighbourhood search, IMA J. Manag. Math 18 (2007) 173–189.

[32] Y.J. Kim, M.C. Lin, D. Manocha, Incremental penetration depth estimation between convex polytopes using dual-space expansion, IEEE Trans. Vis. Comput. Graphics 10 (2004) 152–163.

[33] Y.J. Kim, M.A. Otaduy, M.C. Lin, D. Manocha, Fast penetration-depth computation for physically-based animation, in: Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2002, pp. 23–31.

[34] J. Matoušek, M. Sharir, S. Smorodinsky, U. Wagner, k-sets in four dimension, Discrete Comput. Geom. 35 (2006) 177–191.

[35] N. Megiddo, Linear programming in linear time when the dimension is fixed, J. ACM 31 (1) (1984) 114–127.

[36] J. O'Rourke, S.R. Kosaraju, N. Megiddo, Computing circular separability, Discrete Comput. Geom. 1 (1) (1986) 105–113.

[37] M.H. Overmars, J. van Leeuwen, Maintenance of configurations in the plane, J. Comput. System Sci. 23 (1981) 166–204.

[38] J. Rico, Dynamic planar convex hull, Ph.D. Dissertation, BRICS, Aarhus, Denmark, 2002.

[39] T. Roos, P. Widmayer, k-Violation linear programming, Inform. Process. Lett. 52 (1994) 109–114.

[40] B. Schölkopf, A.J. Smola, Learning with Kernels, The MIT Press, 2002.

[41] M. Sharir, S. Smorodinsky, G. Tardos, An improved bound for k-sets in three dimensions, Discrete Comput. Geom. 26 (2001) 195–204.

[42] H.U. Simon, On the complexity of working set selection, Theoret. Comput. Sci. 382 (3) (2007) 262–279.

[43] G. Tóth, Point sets with many k-sets, Discrete Comput. Geom. 26 (2001) 187–194.

[44] G.T. Toussaint, Solving geometric problems with the rotating calipers, in: Proc. IEEE MELECON'83, Athens, Greece, 1983, A10.02/1–4.