# Comparing box and disk bichromatic discrepancy

Nicolau Oliver Burwitz*[1] and Carlos Seara†[1]

[1]Universitat Politècnica de Catalunya

**Abstract**

In this paper we consider the problem of computing the discrepancy of a bichromatic point set by using boxes and disks and comparing the respective algorithmic complexities.

## 1 Introduction

Let $S$ be a bichromatic $d$-dimension $n$-point set. Let $R$ and $B$ be a the set of red and blue points from $S$ respectively, so $S = R \cup B$. The colouring of the points in $S$ is expressed in the mapping $\chi : S \to \{-1, 1\}$, where blue points are negative and red points are positive. Let us define the bichromatic discrepancy of a geometric shape $\mathcal{SH}$:

$$\Delta(\mathcal{SH}) = \sum_{x \in (\mathcal{SH} \cap S)} \chi(x).$$

This is, the discrepancy of the shape is the number of red points minus the number of blue points. The maximum bichromatic discrepancy of a family of shapes $\mathcal{SH} \in \mathcal{F}$ is defined as:

$$\text{Max}\Delta(S, \chi, \mathcal{F}) = \max_{\mathcal{SH} \in \mathcal{F}} |\Delta(\mathcal{SH})|.$$

The main goal of this paper is to compare existing algorithms and approaches to solve the problem of computing the maximum bichromatic discrepancy of the set $S$ using various families of shapes. The shapes considered are boxes and disks, in various dimensions.

Applications of computing discrepancy are present in several areas of computer science. Three major ones are mentioned in the introduction of the paper by Dobkin et al. [9] are the Agnostic PAC-Learning, $\epsilon$-Approximations, and Sampling Patterns in Graphics.

Sampling patterns are used in ray-tracing for rendering digital images. If the pattern is ill designed it yields visible biasing artefacts. Computing the boxes discrepancy in $2d$ is related to the design of good patterns.

If instead of pixels we instead assume more circular shaped receptors, as the human eye's photo-receptors roughly are, and furthermore take into account other optical effects natural to human eyesight, it suggests studying discrepancy on disks instead of boxes.

There is a lot literature about discrepancy, and the books by Matousek [13] and Chazelle [5] cover in depth the topic. For related results see Bereg et al. [4] and Díaz-Báñez et al. [6, 7].

## 2 Boxes discrepancy

We first introduce the approach presented by Dobkin et al. [9], and Gunopulos [11], to compute the boxes bichromatic discrepancy in $1d$ and $2d$.

### 2.1 $1d$: Intervals

As boxes and disks both define intervals in the 1-dimension ($1d$) case, the results from Dobkin et al. [9] apply to both shapes. We specially want to highlight some results for the $1d$ case, as they provide the properties that are fundamental for the algorithms in further sections.

**Lemma 1** *[9] Given an interval $[l, r]$ on our $1d$ setting, the discrepancy $\Delta([l, r]) = \Delta([l, m]) + \Delta([m, r])$ where $m \in [l, r]$ and $m \notin S$. See Figure 1.*

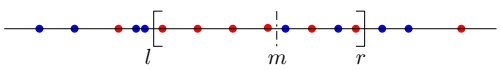


Figure 1: Example points in $1d$ with discrepancy 4.

This allows us to divide and conquer the computation of the discrepancy.

**Lemma 2** *[9] Given an interval $[l, r]$ on our $1d$ setting, let the maximum discrepancy interval be $[a, b] \in [l, r]$. Then for any $m \in [a, b]$ the interval $[a, m]$ maximises the discrepancy among all intervals in $[l, m]$ that have $m$ as the right endpoint. Analogously, for the discrepancy among all intervals in $[m, r]$ with $m$ as the left endpoint.*

*Email: nicolau.oliver@estudiantat.upc.edu

†Email: carlos.seara@upc.edu. Supported by project PID2019-104129GB-I00/ MCIN/ AEI/ 10.13039/501100011033.

Lemma 2 introduces the connection between computing the maximum discrepancy in intervals with some fixed endpoints and computing the maximum discrepancy among all intervals. It specifically implies:

**Observation 3** *[9] Given the maximum discrepancy of two consecutive intervals $[l, m]$ and $[m, r]$ with the fixed endpoint $m$, we can compute the maximum discrepancy for their union interval $[l, r]$.*

From Observation 3 we can intuitively see how we can build a segment tree of the points to represent all possible intervals. Computing this tree takes $O(n \log n)$ time and allows us to solve the static $1d$ maximum discrepancy.

But this tree also allows for a dynamic algorithm. Updates need only to traverse a $O(\log n)$-LENGTH path from the new leaf (or deleted leaf) to the root of the tree. This allows us to solve the dynamic $1d$ maximum discrepancy.

**Theorem 4** *[9] The maximum discrepancy for intervals in $1d$ can be computed in $O(n \log n)$ time (linear if input is sorted) and $O(n)$ space. Computing updates after insertion/deletion of a point can be done in $O(\log n)$ time and $O(n)$ space.*

### 2.1.1 Axis-parallel boxes

The key strategy to tackle the $2d$ setting, is to find projections back to $1d$. The axis-parallel boxes is a perfect example.

Fix the $y$-coordinates of the axis-parallel box. We have $\Theta(n^2)$ pairs to choose from, and each of them defines a horizontal strip. Because we fixed them, the $y$-coordinates of the points inside the strip become irrelevant. See Figure 2.

**Lemma 5** *[9] Computing the maximum discrepancy box in a fixed horizontal strip is equivalent to finding the maximum discrepancy interval of the points inside the strip projected onto the $x$-axis.*

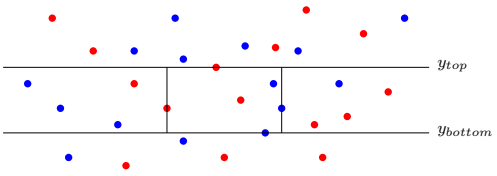


Figure 2: Discrepancy with boxes in $2d$.

These projections allow us to use the previous results to design algorithms. Let $S$ again be the input set of $n$ points. An outline of the algorithm step by step is:

1. Sort $S$ by $y$-coordinate[1], obtaining the order $p_1, \ldots, p_n$.
2. For each $p_i$ do:
   (a) Initialise the segment tree with only $p_i$, this represents the strip that only contains $p_i$.
   (b) For each $p_j$ such that $i < j$ do:
      i. Update the segment tree by inserting $p_j$.
      ii. If the new maximum discrepancy is larger than the one seen so far, record the box.

Step 1 has cost $O(n \log n)$ time. Step 2.b.i has cost $O(\log n)$ time. Step 2.a and 2.b.ii have cost $O(1)$ time. Both loops 2 and 2.b have $O(n)$ iterations, so the total complexity is $O(n^2 \log n)$ time and $O(n)$ space. This algorithm is straightforward to extend to higher dimensions, for each new dimension the "strip" is determined by two extra points, so the complexity is $O(n^{2(d-1)} \log n)$ time and $O(n)$ space.

This can be improved by applying divide and conquer to the $y$-axis, after sorting the input by both coordinates as a pre-computation. This and more improvements where shown by Barbay et al. [2] to result in an $O(n^2)$ time algorithm, or even faster under some parametrizations of the input. They extend this approach to higher dimensions, resulting in an $O(n^d)$ time algorithm. This running time is tight up to subpolynomial factors, as proven by Backurs et al. [1].

## 3 Disk discrepancy

Disks are equivalent to intervals in $1d$, but disks in $2d$ do not satisfy the analogous of Lemma 1. There is no easy way to decompose a disk into smaller disks. Analogously, Lemma 2 doesn't hold.

Nevertheless, we can still apply the key strategy presented in Subsection 2.1.1, finding projections back to $1d$. The projection we present is equivalent to the one used by Bereg et al. [3].

For $p_i, p_j \in S$ consider all disks that pass through them. All their centers lie on the bisector of the segment $\overline{p_i p_j}$. See Figure 3.

**Definition 6** *The* oriented angle $\alpha_k^{ij} \in [-\pi, \pi]$ *of a point $p_k \in S$ with respect to $p_i, p_j \in S$ is the supplementary angle of $\angle p_i p_k p_j$. It is positive if $p_k$ is to the right of the directed line $\overrightarrow{p_i p_j}$, otherwise negative.*

If we order the points in $S$ by the oriented angle, the *furthest* left point in Figure 3 has the smallest oriented angle. The *furthest* right point has the largest oriented angle. Points *close* to segment $\overline{p_i p_j}$ have oriented angle close to 0.

[1]The following for loops iterate in this order.

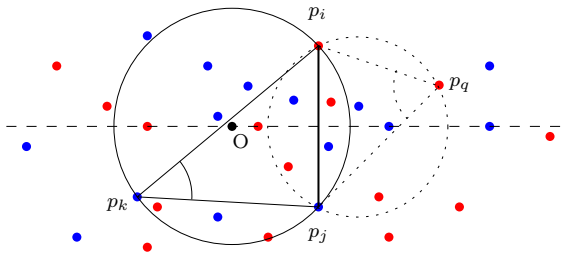


Figure 3: Discrepancy with disks in $2d$. Oriented angle $\alpha$ is negative, and oriented angle $\beta$ is positive.

Traversing all the points $p_k \in S$ by their oriented angle $\alpha_k^{ij} = \pm(\pi - \angle p_i p_k p_j)$ can be visualised as sliding the center O of the disk over the bisector from left to right.

Thus, for fixed points $p_i, p_j \in S$, let

$$\Lambda^{ij} = [\alpha_1^{ij}, \ldots, \alpha_k^{ij}, \ldots, \alpha_{n-2}^{ij}]$$

be the list (in fact, a multi-set) of oriented angles of the points $p_k$ with respect to $p_i, p_j$. Each $\alpha_k^{ij}$ retains the colour of the point it represents.

**Definition 7** *The inverse of an angle $\alpha_k^{ij}$ is:*

$$inv(\alpha_k^{ij}) = \begin{cases} \text{if } \alpha_k^{ij} < 0 : swap\_color(\pi - |\alpha_k^{ij}|) \\ \text{else } \alpha_k^{ij}. \end{cases}$$

Analogously, the inverse of a list

$$inv(\Lambda^{ij}) = [\,\forall k : inv(\alpha_k^{ij})\,]$$

is just the list of the inverse of its elements.

In the circular discrepancy, the intervals of angles must be of the form $[\beta - \pi, \beta]_{\Lambda^{ij}}$ where $\beta \in [0, \pi]$. This interval represents the disk through $p_i, p_j$ with inscribed angle $\beta$, where $\beta$ is positive; so it lies to the right of $\overrightarrow{p_i p_j}$. Let $[0, \beta]_{inv(\Lambda^{ij})}$ be the same interval/disk over the inverse angles of $\Lambda^{ij}$.

**Lemma 8** *The disk discrepancy of $\Lambda^{ij}$ is equal to a constant with respect to $\beta$, plus the interval discrepancy of the inverse angles with fixed endpoint 0.*

$$\Delta([\beta - \pi, \beta]_{\Lambda^{ij}}) = \Delta([-\pi, 0]_{\Lambda^{ij}}) + \Delta([0, \beta]_{inv(\Lambda^{ij})}).$$

**Definition 9** *The projection of a list of angles is:*

$$\mathcal{P}(\Lambda^{ij}) = [\,\forall k : \{\alpha_k^{ij} \cup inv(\alpha_k^{ij})\}\,].$$

In few words, the projection is duplicating the negative angles with its inverses. So a negative red angle is duplicated by inserting its positive value in blue. As a consequence of Lemma 8 we have the following result.

**Theorem 10** *The disk discrepancy of a list of angles is equal to the interval discrepancy of its projection, with the restriction of containing the interval $[-\pi, 0]$,*

$$\Delta([\beta - \pi, \beta]_{\Lambda^{ij}}) = \Delta([-\pi, \beta]_{\mathcal{P}(\Lambda^{ij})}), \quad \beta \in [0, \pi].$$

The algorithm for circular discrepancy starts by fixing two points. We have $O(n^2)$ pairs to choose from, and each of them defines a bisector. Because we fixed the points, the remaining points can be sorted by their oriented angle. These angles are then projected via $\mathcal{P}$. Using this projection, the algorithm is straightforward:

1. For each pair $(p_i, p_j) \in S \times S$ such that $i \neq j$ do:
   (a) Compute the list of oriented angles $\Lambda^{ij}$ of all points with respect to $p_i, p_j$.
   (b) Compute the projection $\mathcal{P}(\Lambda^{ij})$.
   (c) Compute the maximum discrepancy interval with the restriction:
   $$\Delta(\mathcal{P}(\Lambda^{ij})) = [-\pi, \beta], \quad \beta \in [0, \pi].$$

To compute Step 1.c it is enough to modify slightly the algorithm for interval discrepancy.

Steps 1.a and 1.b have cost $O(n)$ and Step 1.c has cost $O(n \log n)$ time. The Loop 1 is $O(n^2)$ iterations so the total complexity is $O(n^3 \log n)$ time and $O(n)$ space. This is equivalent to the complexity of the algorithm presented by Bereg et al. [3].

In comparison to this approach, a faster and more general algorithm exists by Dobkin and Eppstein [8]. Their approach extends to shapes bounded by algebraic curves, such as circles and ellipses in $2d$. Lifting the points to the paraboloid, in order to compute the discrepancy inside the disks they compute the lifted points below the corresponding plane, using the topological sweep algorithm by Edelsbrunner et al. [10]. Their resulting complexity is $O(n^3)$ time for disks, and $O(n^5)$ time for ellipses.

We think that finding the maximum discrepancy disk in $2d$ could be 4-SUM hard. The reduction can be done using Proposition 11 in Heras et al. [12], and Theorems 6, 7 and Lemma 12 in Bereg et al. [4].

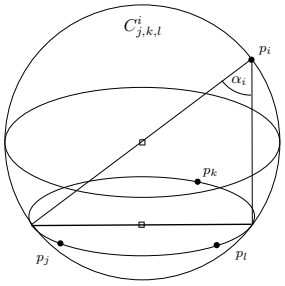


Figure 4: Generalization of the oriented angle in 3d.

It is straightforward to generalise to $d > 2$ the algorithm we present above and the one by Dobkin and Eppstein [8]. See Figure 4.The respective complexities are $O(n^{d+1} \log n)$ and $O(n^{d+1})$ time and $O(n)$ space.

## 4 Unoriented boxes

In contrast to disks in $2d$, boxes with arbitrary orientations in $2d$ can be decomposed into smaller boxes of that same orientation. Again this impacts positively the complexity of the algorithm.

**Lemma 11** *Computing the maximum discrepancy box in a fixed strip with orientation $\vec{v}$ is equivalent to finding the maximum discrepancy interval of the points inside the strip projected onto a line with direction $\vec{v}$.*

This is just a generalisation of Lemma 5, and allows us to reuse the results for axis parallel boxes.

**Lemma 12** *A set $S$ of $n$ points in $2d$ has $O(n^2)$ unique linear projections onto $1d$.*

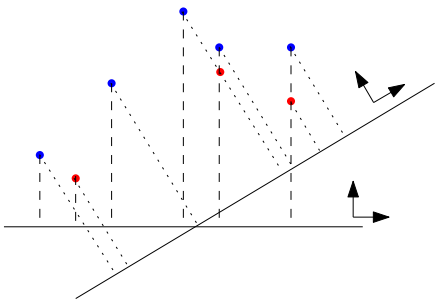

Figure 5: Two projections on a line.

Furthermore, the projections of the set $S$ can be sorted by their angle. Two consecutive projections differ by the swap of two points. This can be processed in two insertion/deletion updates using the dynamic algorithm for intervals in $1d$. The trivial algorithm reusing the boxes discrepancy algorithm is $O(n^4 \log n)$ time. We are currently studying other approaches to improve this trivial complexity.

## 5 Conclusions

The following table illustrates the time complexities of the algorithms for computing the discrepancies for boxes, disks and unoriented boxes.

| | Boxes | Disks | Unoriented Boxes |
|---|---|---|---|
| $d = 1$ | $O(n \log n)$ | | |
| $d = 2$ | $O(n^2)$ | $O(n^3)$ | $O(n^4 \log n)$ |
| $d \geq 3$ | $O(n^d)$ | $O(n^{d+1})$ | ? |

**Open problem 13** *Is the maximum bichromatic disk discrepancy problem in $2d$ 4-SUM hard?*

Exploiting advanced data structures in the $2d$ setting could be promising. Specifically in the case of oriented boxes, we attempted fruitlessly to use quad-trees to extend the algorithm for discrepancy on intervals in $1d$. The hope was that quad-trees of the points can be rotated with only $O(n^2)$ updates.

**Open problem 14** *Is there a faster algorithm for unoriented boxes in $d \geq 2$?*

## References


[1] A. Backurs, N. Dikkala and C. Tzamos. Tight Hardness Results for Maximum Weight Rectangles. *ArXiv abs/1602.05837*, (2016).

[2] J. Barbay, T.M. Chan, G. Navarro and P. Pérez-Lantero. Maximum-weight planar boxes in $O(n^2)$ time (and better). *Information Processing Letters*, Vol. 114(8), (2014), pp. 437–445.

[3] S. Bereg, O. Daescu, M. Zivanic and T. Rozario. Smallest Maximum-Weight Circle for Weighted Points in the Plane. *ICCSA*, (2015), pp. 244–253.

[4] S. Bereg, J.M. Díaz-Báñez, D. Lara, P. Pérez-Lantero, C. Seara, and J. Urrutia. On the coarseness of bicolored point sets. *Computational Geometry: Theory and Applications*, 46(1), (2013), pp. 65–77.

[5] B. Chazelle. *The Discrepancy Method in Computational Geometry*. Handbook of Discrete and Computational Geometry, CRC Press 44, (2004), pp. 983–996.

[6] J.M. Díaz-Báñez, R. Fabila, P. Pérez-Lantero, I. Ventura. New results on the coarseness of bicolored point sets. *Information Processing Letters*, 123, (2017), pp. 1–7.

[7] J.M. Díaz-Báñez, M.A. López, C. Ochoa, P. Pérez-Lantero. Computing the coarseness with strips or boxes. *Discrete Applied Mathematics*, 224(19), (2017), pp. 80–99.

[8] D.P. Dobkin and D. Eppstein. Computing the Discrepancy. *9th Annual Symposium on Computational Geometry*, (1993).

[9] D.P. Dobkin, D. Gunopulos, and W. Maass. Computing the maximum bichromatic discrepancy, with applications to computer graphics and machine learning. *Journal of Computer and Systems Sciences*, 52(3), (1996), pp. 453–470.

[10] H. Edelsbrunner and L.J. Guibas. Topologically sweeping an arrangement. *Journal of Computer and System Sciences*, Vol. 38(1), (1989), pp. 165–194.

[11] D. Gunopulos. Computing the Discrepancy. *Thesis in Princeton University*, Princeton, N.J. (1995).

[12] A. de las Heras, G. Esteban, D. Garijo, C. Huemer, A. Lozano, N. Oliver and D. Orden. Measuring cocircularity in a point set. *ECG23* (2023).

[13] J. Matoušek. *Geometric Discrepancy: An Illustrated Guide*. Springer-Verlag, (1999).