MASTER'S THESIS

Master of Science in Advanced Mathematics and Mathematical Engineering

Title: Two problems in Computational Geometry

Author: Nicolau Oliver Burwitz

Advisors: Clemens Huemer and Carlos Seara

Department: Departament de Matemàtiques

Academic year: 2022-2023

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

Universitat Politècnica de Catalunya
Facultat de Matemàtiques i Estadística

Master in Advanced Mathematics and Mathematical Engineering

Master's thesis

Two problems in Computational Geometry

Nicolau Oliver Burwitz

Supervised by Clemens Huemer and Carlos Seara

June, 2023

Thanks to my parents, for their patience and love. To my grandparents, for their strength.

Abstract

Two different problems belonging to computational geometry are studied in this thesis. The first problem studies: given a set S of n points in the plane in general position, how close are four points of S to being **cocircular**. We define three measures to study this question, the *Thales*, *Voronoi* and *Determinant* measures.

We present bounds on the Thales almost-cocircularity measure over a point set. Algorithms for computing these measures of cocircularity are presented as well. We give a reduction from computing cocircularity using the Determinant measure to the 4SUM problem.

The second problem studies: given two sets R and B of red and blue points respectively, how to compute the **bichromatic discrepancy** using *boxes* and *circles*. The bichromatic discrepancy is defined as the difference between the number of red points and blue points inside the shape. We present a comparison of algorithms in the existing literature for the two shapes. Bichromatic discrepancy in axis-parallel boxes .vs non-axis-parallel boxes is also compared.

Furthermore, we also present a new algorithm for disk discrepancy in higher dimensions, based on existing literature. We also relate existing problems in separability with existing output sensitive algorithms for bichromatic discrepancy using boxes.

Keywords

General position, cocircularity, algorithms, Erdős–Szekeres theorem
Discrepancy, coarseness, boxes, disks, 3SUM-hard

MSC 2020

Primary 52-08, 52C10. Secondary 52-02, 52-06, 51N20.

Contents

1. Introduction

Both problems presented in this thesis belong to the field of Computational Geometry, that studies the inherent complexity of geometric problems under various models of computation. It lies at the intersection of Theoretical Computer Science, Geometry (usually Euclidean) and Combinatorics. Analyzing the computational complexity of a problem usually entails a study of its combinatorial complexity, followed by the design of algorithms using geometric tools and properties to achieve efficiency and elegance.

The problem studied in Section 2 focuses on points in the plane in general position. We study how close are four points from a set S of n points of being cocircular. We investigate the combinatorial complexity of this problem and show algorithms to detect cocircularities. This problem is very related to analogous results on finding collinearities in a point set, that has been widely studied. In Section 2.1 we define three measures for cocircularity, the *Thales* cocircularity, the *Voronoi* cocircularity, and the *Determinant* cocircularity. We present properties, similarities and differences among them. In Section 2.5 we present some algorithms to compute various of the measures of cocircularity.

Section 3 is a comparative analysis of existing literature about a family of problems on chromatic discrepancy. Given a bichromatic set of red and blue points, the discrepancy of a shape is the absolute value of the difference of red and blue points that lie inside the shape. The families of problems we are concerned with, are finding which specific shape among a family of them has maximal discrepancy.

The main goal is to understand similarities and differences among existing techniques to compute the maximal discrepancy. Several approaches exist for each problem in various dimensions, and properly comparing them compiles the state of the art. We start by presenting existing results on axis-parallel boxes discrepancy in Section 3.2. In Section 3.3 we present existing and new results on disk discrepancy. In Section 3.4 we briefly explore the non-axis-parallel boxes discrepancy. And finally, in Section 3.5 we compile all the cited and new results in a final comparison and conclusion.

Even if there are some casual similarities between both problems studied in this thesis, each should be treated as independent of the other, as that has been the working methodology. But certain specific links and similarities have fortuitously appeared, mostly concerning the reduction of some problems to 4SUM and the use of techniques involving capable arcs.

Both problems result from collaborating with the UPC Research Group on Discrete, Combinatorial and Computational Geometry (DCCG), and are accepted publications to the XX Spanish Meeting on Computational Geometry (EGC'23) [1, 24] [1].

The first problem on cocircularity was initiated at the TOPPING workshop in Barcelona, July 2022. The authors of the submitted paper are Andrea de las Heras, Guillermo Esteban, Delia Garijo, Clemens Huemer, Antoni Lozano, Nicolau Oliver and David Orden. I also thank Mercè Claverol, Dolores Lara, Alejandra Martínez, and Pablo Perez for their valuable discussions and contributions.

The second problem on bichromatic discrepancy is the continuation of my collaboration with Carlos Seara, who also supervised my bachelor's thesis on bichromatic separability. This previous work with him is also an accepted publication on EGC'23. The topics of bichromatic discrepancy and separability have a lot of overlap, and some of the surveyed results on discrepancy are related to results in Oliver and Seara [25].

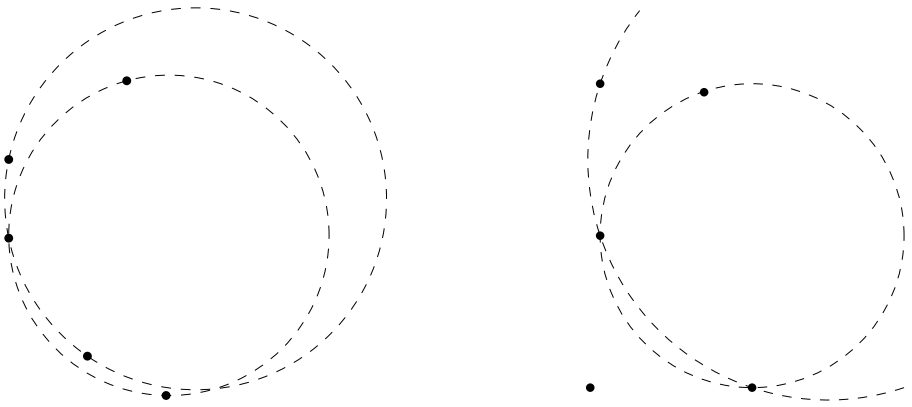
[1] https://egc23.web.uah.es/

2. Measuring cocircularity

Most algorithms in Computational Geometry have as input geometric objects such as segments, polygons, triangulations... In these cases, each of increasing complexity, the underlying building block are points and relations over these. Most algorithms in Computational Geometry require the input points to be in general position.

Definition 2.1. A set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ of points in the Euclidean plane R^2 is in **general position** if:

- No three points exist in S belonging to the same line.
- No four points exist in S belonging to the same circle.



(a) Points that are nearly cocircular.

(b) Points that are less cocircular.

Figure 1: Two examples of sets S of five points in general position.

This requirement simplifies the design of algorithms as most degenerate situations arise from collinearity, but also from cocircularity. In some cases, the problems solved by the algorithms aren't even well defined if the points aren't in general position. Some famous examples taught in undergraduate courses include the Bentley–Ottmann algorithm for finding all crossings of a set of segments, and the problem of computing the Delaunay triangulation. For the first example there exists an algorithm easy to implement if one omits degeneracies, and the second problem is ill-defined for points not in general position.

But it is rarely guaranteed, especially in real life data-sets, that the input set of points is in general position. This prompts the study of bounds that relate both collinearity and cocircularity, with the number of input points. The central question being: how close/far is any set S of n points of containing four that are almost collinear/cocircular.

As the goal is to study these properties over arbitrary sets S of points in the plane, we will first mention some very well known results from Ramsey Theory and their applications in geometry. This branch of combinatorics focuses on proving the existence of certain substructures in sufficiently large sets. A canonical example in graph theory is the existence of a large monochromatic clique in any coloring of the complete graph of sufficiently large size. In layman terms, it seeks to find order among chaos.

A very important result by Erdős and Sekeres [17] states that for every set S points in the plane, with bounded size $|S| = 2^{n+o(n)}$ by Suk [2], there exists a convex polygon of n sides with vertices in S. This is the famous "happy ending" theorem. From this, a known bound on how close points form S are to being collinear can be deduced as follows:

Lemma 2.2. *The largest angle defined by any triplet of S is bounded from below by $\pi \cdot (1 - 2/n)$.*

Proof. Let the collinearity of a set of points be measured by the largest angle formed by any three points of the set. Consider that the n inner angles of the convex n-gon implied by the result of Erdős and Sekeres must sum $\pi(n-2)$. Then at least one of these n angles is at least $\pi(n-2)/n$. And the largest angle defined by any triplet of S is bounded from below by $\pi \cdot (1 - 2/n)$. □

The entire proof of Erdős and Sekeres [17] improves this bound to $\pi \cdot (1 - 1/n)$, and holds for sets S of 2^n points . Notice the two relevant statements made in the proof. First, the definition of a measure of collinearity. Second, a bound on that measure. This is relevant to clarify, as other intuitive notions of collinearity might exist. Namely, we could opt to measure collinearity using the thinnest strip that contains 3 points of S. This notion is not equivalent to the one presented above, and illustrates the variety of interpretations that will also appear when studying cocircularity.

Following closely this latter interpretation of collinearity, we could choose to measure cocircularity using the smallest width annulus that contains 4 points of S. This overlaps with a very well studied algorithmic problem, the problem of computing the annulus of smallest width that contains S. In García-López et al. [19] they define the roundness of a point $x \in \mathbb{R}$ as $\mathcal{RS}(x) = d(x, CN(x)) - d(x, FN(x))$, where $CN(x)$ is the closest point (neighbor) in S from x, and $FN(x)$ the furthest. Both $CN(x)$ and $FN(x)$ are clearly related to Voronoi diagrams of S, yet another overlap of tools and techniques with our results. García-López et al. [19] find minimums for the roundness function.

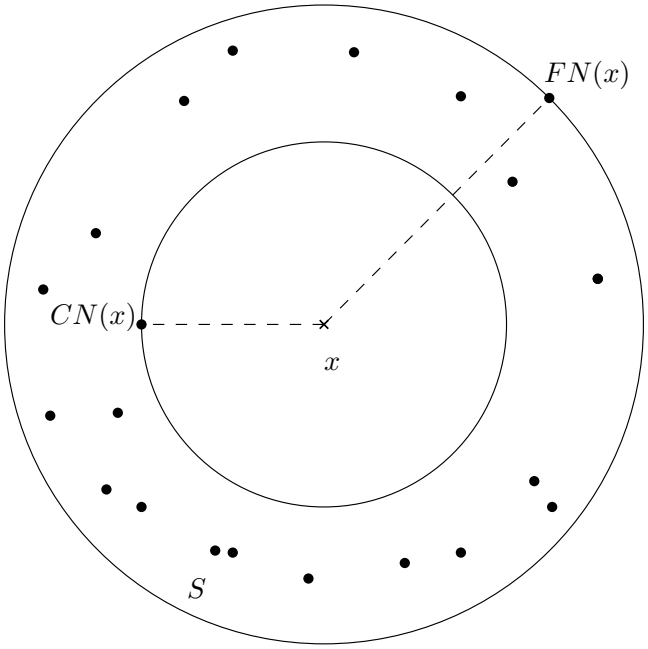


Figure 2: Example of annulus containing S centered at x, indicating the closest neighbor and furthest neighbor of x.

2.1 Basics on Measures for Cocircularity

In order to study how close are four points from S to being cocircular, we define several measures of cocircularity. For this, we recall the formal definition of measures, simplified to measures over points only:

Definition 2.3. Let S be any set of points on the Euclidean plane, a function $\mathcal{C} : S \to \mathbb{R} \cup \{-\infty, \infty\}$ is a measure if the following conditions hold:

1. Non negativity: $\forall S \quad \mathcal{C}(S) \geq 0$
2. $\mathcal{C}(\emptyset) = 0$
3. Countable additivity: $\forall S, \forall S' \subseteq S \quad \mathcal{C}(S') \leq \mathcal{C}(S)$

The measures we present yield 0 if the set of points contains a quadruplet of cocircular points. To meet the previous conditions, we should formally define our measures using their reciprocal ($1/\mathcal{C}(S)$). For ease of presentation, this is omitted. In other words, we could understand our measures as defining non-cocircularity. Despite this detail, we will refer to them as measures of cocircularity for simplicity. To further clarify, let us introduce the most basic and intuitive measure of cocircularity:

Definition 2.4. Let the Binary cocircularity of a set of points S be:

$$\mathcal{B}(S) = \begin{cases} 0 & \exists A, B, C, D \in S \text{ s.t. they are cocircular} \\ 1 & \text{otherwise} \end{cases}$$

This measure captures the basic behavior we are interested in, but it tell us nothing on how "close" non-cocircular points are from being cocircular. Still, it is useful as a baseline to build upon. In order to study how close are four points from S to being cocircular we need functions with a continuous image, unlike the discrete image of the Binary measure. It is necessary to define a measure that contains an interval $[0, x]$ with $x \in \mathbb{R}^+ \cup \infty$ as image space. This can be quite counter-intuitive, and leads to questions like:

1. What are the most non-cocircular sets of four points?
2. Perturb four cocircular points slightly so that they do not lie on a common circle any more. Does the radius of the circle impact the measure of cocircularity?
3. Are collinear points also cocircular? If so, how does this interact with the previous question.
4. Should the points be considered in any particular order? What if the points are in a non-convex position?

Different intuitions on cocircularity yield different answers on these questions, and consequently result in different measures of cocircularity. We present the following three measures of cocircularity (in order of appearance in this thesis): **Thales** measure, **Determinant** measure, and the **Voronoi** measure.

The measures will differ widely, but some properties are desirable among them all. The first property all measures of cocircularity ought to have, is compatibility with the Binary cocircularity. If there is a quadruplet of cocircular points in the set S, any measure of cocircularity over S should be 0.

$$\forall \mathcal{C}, \forall S \quad \mathcal{B}(S) = 0 \implies \mathcal{C}(S) = 0$$

Binary cocircularity is the agreed upon, sensible and coherent baseline. Different measures may improve on it, but must always retain this compatibility.

The second one is that the measure is local, meaning that it is defined in terms of the quadruplets of points, and not in terms of more points in S. In formal terms, let $\mathcal{C}(S)$ be any measure of cocircularity on a set S of points, then:

$$\mathcal{C} \text{ is local} \iff \forall S, \exists \{A, B, C, D\} \subseteq S \quad \mathcal{C}(S) = \mathcal{C}(\{A, B, C, D\})$$

There must be at least one quadruplet in the set S that determines the value of the measure of cocircularity on S, and the remaining points do not strictly affect the measures.

All the measures we present are local measures of cocircularity, and are compatible with the Binary measure.

Definition 2.5. The Thales/Determinant/Voronoi measure of a set S of points is the minimum of the Thales/Determinant/Voronoi measure among all 4-tuples of points of S.

2.2 The Thales measure

The first measure we present is inspired on the measure of collinearity presented by Erdős and Sekeres. If collinearity is to be measured by the largest angle defined by a triplet of points, cocircularity should be measured in terms of the angles defined by a quadruplet of points. Let $\{A, B, C, D\}$ be such quadruplet.

In order to study the relation among the angles defined by the quadruplet of points, we make use of the inscribed angle theorem, also known as the Thales' theorem:

Theorem 2.6. Inscribed angle theorem:

A circle with center O that crosses the endpoints of the segment $\overline{AB}$, and has inscribed angle $\angle ApB = \theta$, must have as central angle $\angle AOB = 2\theta$. The circle is the locus of all points p with $\angle ApB = \theta$, or $\angle ApB = \pi - \theta$ if and only if p and the center O lie on the opposite side of the segment $\overline{AB}$. See Figure 3.

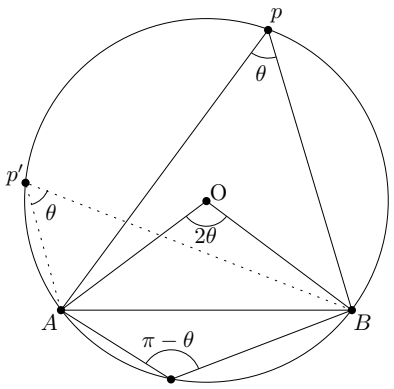


Figure 3: *Capable arc* supported on $\overline{AB}$ and of inscribed angle θ

Because it is a very useful tool, we name this locus of points by its misnomer in Spanish *"capable arc"*.

Definition 2.7. The *capable arc* of $\overrightarrow{AB}$ and angle θ is the locus of all points p with $\angle ApB = \theta$ if p is to the left of $\overrightarrow{AB}$, otherwise $\angle ApB = \pi - \theta$.

Given a quadruplet $\{A, B, C, D\}$ consider w.l.o.g the segment $\overline{AB}$. Using Thales' theorem we can understand the angles among $\{A, B, C, D\}$ as defining capable arcs. To compare how cocircular are the points C and D, with respect to $\overline{AB}$, it is intuitive to compare $\angle ACB$ with $\angle ADB$.

Corollary 2.8. *Applying the inscribed angle theorem to our quadruplet, if C and D lie on the same side of the segment $\overline{AB}$, they are cocircular if and only if $\angle ACB = \angle ADB$. Otherwise they are cocircular if and only if $\angle ACB = \pi - \angle ADB$*

The cocircularity of $\{A, B, C, D\}$ is achieved if equality is reached. But if not, we must compare the two angles $\angle ACB$ and $\angle ADB$. This comparison can be accomplished through diverse functions, but the one we found most intuitive is the absolute value of the difference.

Definition 2.9. Let $\{A, B, C, D\}$ be a quadruplet of points. We define the Thales cocircularity as:

$$\mathcal{T}(A, B, C, D) = \min_{P}\{\min\{|\alpha - \beta|, |\alpha - (\pi - \beta)|\}\}.$$

Where $\alpha = \angle ACB$ and $\beta = \angle ADB$, and the minimum $\min_P$ is taken over all permutations P of the four points $\{A, B, C, D\}$. See Figure 4.

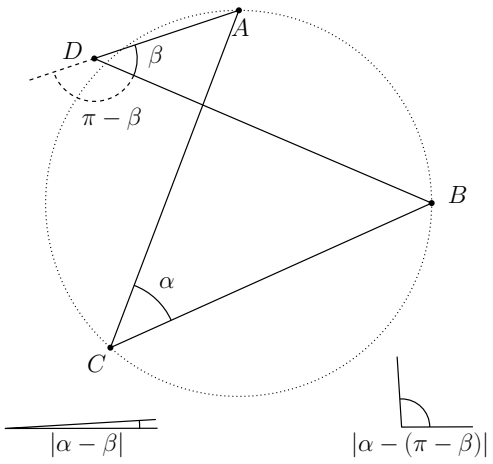


Figure 4: Angles used in Thales cocircularity.

The minimum over all permutations P ensures that all pair of points are considered as the segment subtending the two angles α and β. Notice that symmetries on the measure result in lots of permutations yielding the same result.

Also notice that the minimum of the difference with and without the supplementary angle $(\pi - \beta)$ ensures that angles are compared appropriately. As remarked before, the capable arc of an angle on one side of the segment would correspond to the capable arc of the supplementary angle on the other side. Given a point set and its Thales measure, one can also visualize it through the use of *lunes*.

Definition 2.10. Let D_1 and D_2 be two distinct not-tangent disks in the Euclidean plane that have non-empty intersection $D_1 \cap D_2 \neq \emptyset$. Let A and B be the two points in the intersection of the circles bounding the disks. We define the *lune* L supported on $\overline{AB}$ corresponding to D_1 and D_2 as:

$$L = (D_1 \setminus D_1 \cap D_2) \cup (D_2 \setminus D_1 \cap D_2).$$

Let $\{A, B, C, D\}^{\min}$ be the quadruplet of the point set S with smallest angle difference according to Definition 2.9. Then a pair of capable arcs, w.l.o.g passing through A and B, defines a lune between them that must be empty of points in S. So we can alternatively define the Thales measure of cocircularity as the empty lune of smallest angle difference supported on points of S.

As for the extremal values of the measure, it follows from the definition that

$$\forall S : 0 \leq \mathcal{T}(\mathcal{S}) \leq \pi$$

as two angles can't differ more than π, otherwise the supplementary of one of them yields a difference smaller than π. But this analysis of the extremal values of the Thales measure can be refined. All the inner angles of the quadrilateral formed by a quadruplet (A, B, C, D) must sum up to 2π. So by the pigeonhole principle, there must be a pair of angles α, β that sum at most $\pi/2$. This immediately yields the better range of possible values of $\mathcal{T}(\mathcal{S})$:

$$\forall S : 0 \leq \mathcal{T}(\mathcal{S}) \leq \frac{\pi}{2}$$

.

The extremal configuration $\{A, B, C, D\}^{\max}$ in Figure 5 has Thales measure $\mathcal{T}(A, B, C, D) \to \frac{\pi}{2}$ as $\epsilon \to 0$, so it is asymptotically tight.

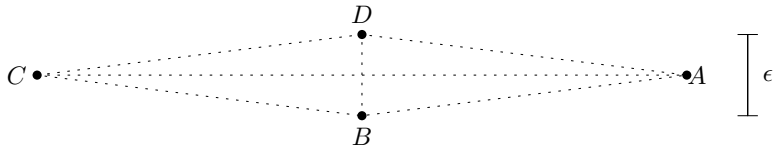


Figure 5: Quadruplet with extremal Thales measure.

2.2.1 Bounds on the Thales measure

We present two upper bounds on the Thales measure of cocircularity. For the simpler first bound, we will fix a pair of points $\ell, r \in S$ as a segment, and consider the measure of the remaining points with respect to this segment.

Proposition 2.11. *For any set S of n points and any two points $\ell, r \in S$ there exist other $a, b \in S$ such that $\mathcal{T}(\ell, r, a, b) \leq \pi/(n-3)$.*

Proof. Fix two points ℓ, r. The $n-2$ remaining points are candidates for a, b. When measuring, consider only the permutation (ℓ, r, a, b) so the angles measured are $\angle \ell a r$ and $\angle \ell b r$.

The $n-2$ angles subtended by the segment $\overline{\ell r}$ lie in the interval $[0, \pi]$, by definition. Consider the angles of the remaining points of S with respect to $\overline{\ell r}$. Take the supplementary angle of the points that lie to the left of $\vec{\ell r}$, in order to make them comparable to the remaining points of S.

Subdivide $[0, \pi]$ into $n-3$ sub-intervals of equal length. By the pigeonhole principle, there must be two points a, b whose angles $\angle \ell a r$ and $\angle \ell b r$ fall in the same interval.

Thus $\exists a, b \in S \setminus \{\ell, r\}$ such that $|\angle \ell a r - \angle \ell b r| \leq \pi/(n-3)$. □

The bound can be thought of through the visualization of lunes again. The bound is on the size of the lune with smallest angular difference, supported on $\overline{\ell r}$, that contains two points in $S \setminus \{\ell, r\}$.

We can show a better bound for sets of points S is in convex position:

Theorem 2.12. *For any set S of n points in convex position, there exist $A, B, C, D \in S$ such that $\mathcal{T}(A, B, C, D) \in O(\frac{1}{n^2})$.*

Proof. Let the points of S be $p_1, \ldots, p_n$ in the clockwise order they are encountered along the boundary of the convex hull of S. Let $S_1 = \{p_1, \ldots p_{n/2-1}\}$ be the first half and $S_2 = \{p_{n/2}, \ldots, p_n\}$ be the second half.

For each p_i from S_1 build a triangulation T_i of $S_2 \cup \{p_i\}$ where we connect p_i to all vertices of S_2 in a fan. Then p_i is incident to $\frac{n}{2}$ triangles in T_i whose angles at p_i sum less than π. At least half of these $\frac{n}{2}$ angles are less than $\frac{4\pi}{n}$, we label them "small" angles.

Summing over all $i = 1 \ldots, \frac{n}{2} - 1$, we obtain at least $\left(\frac{n}{2} - 1\right) \cdot \frac{n}{4}$ small angles incident to a point from S_1 in $\bigcup_{i=1}^{\frac{n}{2}-1} T_i$. Each small angle lies in a triangle formed by a point of p_i and two consecutive points of S_2. By the pigeon-hole principle, at least one edge formed by two consecutive points of S_2 subtends at least $\frac{\left(\frac{n}{2}-1\right)\cdot\frac{n}{4}}{\frac{n}{2}} = \frac{n}{4} - 2$ angles that are less than $\frac{4\pi}{n}$. Let this edge be ab.

Similar to the proof of Proposition 2.11, we subdivide the interval $[0, \frac{4\pi}{n}]$ into $\frac{n}{4} - 3$ disjoint intervals of equal length. Again by the pigeon-hole principle, at least two of the $\frac{n}{4} - 2$ small angles, that must be less than $\frac{4\pi}{n}$, fall into the same interval. Consequently, there exist two angles subtended by the edge ab whose difference is at most:

$$\frac{\frac{4\pi}{n}}{\frac{n}{4} - 3} = \frac{16\pi}{n(n-12)} \in O\left(\frac{1}{n^2}\right).$$

□

Some constants of this proof can be improved easily if instead of using triangulations from S_1 and S_2, we triangulate the entire S. But the presented proof using these two sets hits at possible future lines of research, as we try and generalize the bound to general point sets.

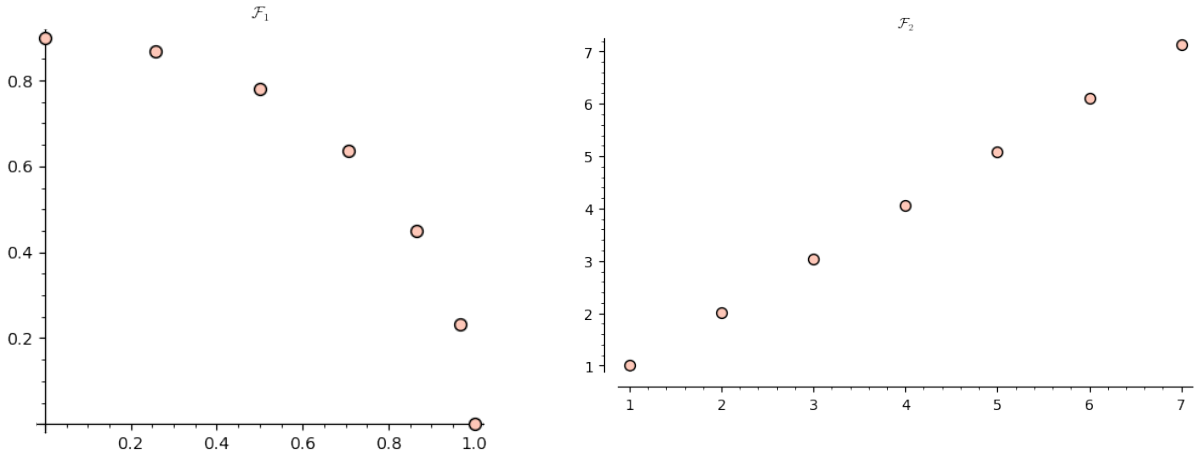
It can be shown that if given a non-convex S, one finds a start polygon defined by the points in S_1, such that S_2 lay inside its kernel, one can replicate the previous proof, obtaining the same bound for a non-convex S. Unfortunately it remains an open question to generalize this bound to non-convex S.

The question of finding lower bounds remains a work in progress. The strategy we attempted was finding a family of points that was asymptotically far from cocircular. This involved a lot of trial and error. As analytically studying all these families was unfeasible due to time and knowledge constraints, we screened empirically the candidates. We computed the Thales cocircularity for the families of points, plotting the results against the number of points.

The long list of the families studied include points equispaced on: polynomials of low degree (parabola, quadratics) including polynomials with degree $d = 1 \pm \epsilon$ and Bézier curves, the logarithm function, conics (ellipses and hyperbolas), cycloids, various spirals (logarithmic, hyperbolic, littus spiral, Fermat spiral), points in the integer grid with special interest in permutations of the $x = y$ diagonal points.

After all this blind search, with lots of interesting empirical results[2] some of the more promising families had some counter-intuitive properties.

Our initial bias was that families with lots of variations, like the latter points in the grid, would exhibit very good results, but would be difficult to analyze analytically. We can only confirm the latter. Much simpler families resulted in quite good results. We present only two, the family $\mathcal{F}_1$ the near circular ellipse, and $\mathcal{F}_2$ the near linear polynomial. See Figure 6.



(a) Coordinates of equispaced points on one quadrant of a near-circular ellipse $(x, y) = (a\cos(t), b\sin(t))$ $0 \leq t \leq \pi/2 \quad a = 1, b = 0.9$.

(b) Equispaced points on a near-linear polynomial $y = x^{1.1}$.

Figure 6: Two of the studied families for lower bounds.

As the results for both families are similar, we present the ones yielded by the near-circular ellipse. We study how cocircular these points sets are with respect to the number of points n, trying to visualize if they have promising asymptotic behavior. See Figure 7. These families that are very close to cocircular/collinear, still yield interesting results. We highlighting the important property that they always fail to be exactly cocircular in a predictable fashion. Recall that in the case of the near-linear polynomial, collinearity is treated as cocircularity by the Thales measure.

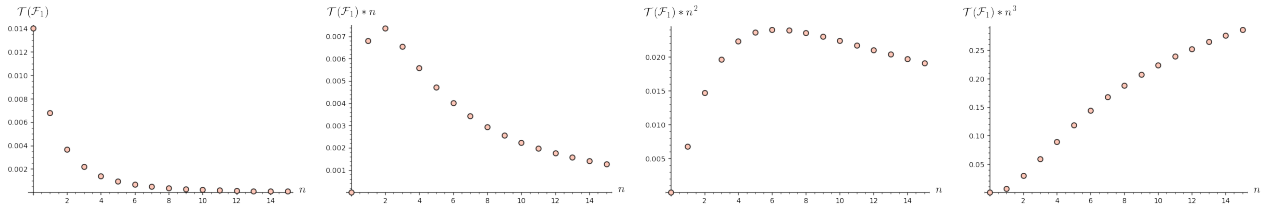


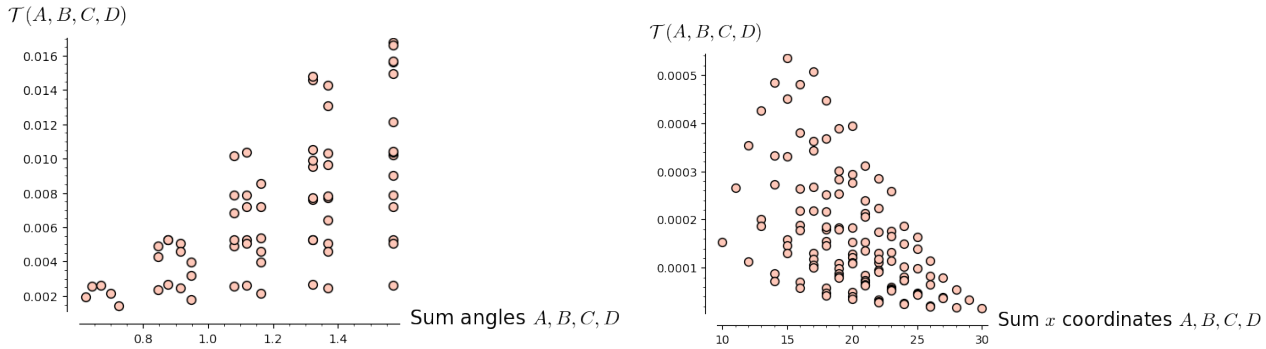
Figure 7: Thales cocircularity of the near-circular ellipse $\mathcal{T}(\mathcal{F}_1)$ against the number of points n.

To conduct a very rough analysis of the results, we multiply them by n until we find a positive tendency. As we can see on the rightmost plot, the tail of the function seems to be of the order $\Omega(\frac{1}{n^3})$. But this could be easily misleading, as we have very few points.

[2] That we don't present for the sake of brevity.

Another bad sign is that there is a change in monotonicity for the two middle plots, suggesting that eventually the plotted tendency starts decreasing. These are very rough results, but already suggest that it might be interesting potential to continue working on this family.

The next logical step would be to identify the quadruplet of points that is most cocircular in the two families. This is where we have found very promising results. See Figure 8. This immediately yielded very clear and noticeable patterns for both families, indicating that it was always the same recognizable tuple that was minimizing the Thales measure. This is very good news as, if proved analytically, would simplify a lot the analysis of the measure for these families.



(a) Thales cocircularity of all quadruplets $\mathcal{T}(A,B,C,D)$ $A,B,C,D \in \mathcal{F}_1$ for $n=8$ against the sum of the parameter t of each point in the quadruplet $t_A+t_B+t_C+t_D$.

(b) Thales cocircularity of all quadruplets $\mathcal{T}(A,B,C,D)$ $A,B,C,D \in \mathcal{F}_2$ for $n=9$ against the sum of the x-coordinates of each point in the quadruplet $x_A+x_B+x_C+x_D$.

Figure 8: Analysis to identify the most cocircular quadruplet of $\mathcal{F}_1$ and $\mathcal{F}_2$.

All these results are very preliminar, and shouldn't be taken as rigorous conjectures. But they had to be mentioned as they represent the most time consuming experiments designed for this thesis.

2.3 The Determinant measure

The Determinant cocircularity uses a very common tool to check if four points are cocircular. It is well known that four points A, B, C, D in the plane lie on a common circle if and only if their lifted projections to the paraboloid $(x,y) \mapsto (x,y,x^2+y^2)$ are coplanar. Testing coplanarity of four points $p,q,r,s \in \mathbb{R}^3$ is canonically done using the following determinant:

$$\det \begin{pmatrix} x_p & y_p & z_p & 1 \\ x_q & y_q & z_q & 1 \\ x_r & y_r & z_r & 1 \\ x_s & y_s & z_s & 1 \end{pmatrix}$$

This basic operand is ubiquitous and is the basic primitive for lots of more complex algorithms and data-structures in computational geometry, as shown in Guibas and Stolfi [20].

Definition 2.13. The Determinant cocircularity of four points $\mathcal{D}(A,B,C,D)$ with coordinates $A=(x_A,y_A)$, $B=(x_B,y_B)$, $C=(x_C,y_C)$, $D=(x_D,y_D)$ is the absolute value of the determinant:

$$\mathcal{D}(A,B,C,D) = \left| \det \begin{pmatrix} x_A & y_A & x_A^2+y_A^2 & 1 \\ x_B & y_B & x_B^2+y_B^2 & 1 \\ x_C & y_C & x_C^2+y_C^2 & 1 \\ x_D & y_D & x_D^2+y_D^2 & 1 \end{pmatrix} \right|.$$

This measure is invariant under translation, but under a scaling by a factor c, the determinant varies in a factor of c^4. This among other properties make this measure somewhat unwieldy to study analytically, but very practical for the decision problem of deciding if for points are exactly cocircular. In the following section, we consider the Determinant cocircularity to show a relation to the 4SUM problem.

2.3.1 Reduction from 4SUM

The k-SUM problem is stated as follows: given a set $S \subset \mathbb{Z}$ of different n integers, decide if there exists a k-tuple of distinct elements $(s_1, \dots, s_k) \in S$ whose sum is zero, $\sum_{i=1}^{k} s_i = 0$. This is a prominent problem for $k = 3$, referred as the 3SUM problem, and that can be restated as finding a triplet $(a, b, c) \in S$ of distinct integers that satisfy the equation $a + b = c$. A famous conjecture is that the time complexity of 3SUM is $\Omega(n^{2-o(1)})$, even in expectation [21].

Introduced in Gajentaan and Overmars [18], a complexity class of problems in computational geometry has been build upon 3SUM. Since, many geometric problems have been reduced to 3SUM, they call such problems 3SUM-hard. Of specific interest to us, they show that the problem of detecting collinear points in a set is 3SUM hard.

There exist easy quadratic-time algorithms to solve both 3SUM and 4SUM in the integer RAM model. The generic quadratic-time algorithm for 4SUM uses hashing. Under the real RAM model there exists a similar algorithm without hashing of complexity $O(n^2 \log n)$. We prove that cocircularity for the Determinant measure is 4SUM-hard.

Proposition 2.14. *Given n integers $[x_1, x_2, \dots, x_n]$ from the 4SUM problem. Let S be the set of n points $x_i \mapsto (x_i, x_i^2)$ on the parabola $y = x^2$. Then S contains four cocircular points if and only if the x-coordinates of these four points sum* 0.

Proof. Let A, B, C, D be four different points on the parabola $y = x^2$ with coordinates:

$$A = (p, p^2), B = (q, q^2), C = (r, r^2), D = (s, s^2).$$

The Determinant cocircularity measure then simplifies the absolute value of:

$$\left| \det \begin{pmatrix} p & p^2 & p^2+p^4 & 1 \\ q & q^2 & q^2+q^4 & 1 \\ r & r^2 & r^2+r^4 & 1 \\ s & s^2 & s^2+s^4 & 1 \end{pmatrix} \right| =$$

$$-(r-s)\cdot(q-s)\cdot(q-r)\cdot(p-s)\cdot(p-r)\cdot(p-q)\cdot(p+r+s+q).$$

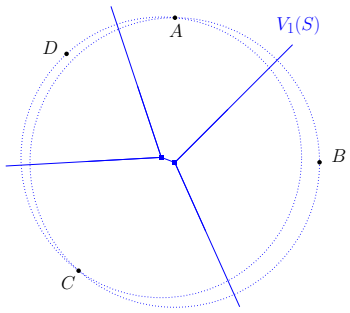
Then the determinant is zero if and only if $p + r + s + q = 0$. □

This measure could be extended to higher dimensions, as all the tools used are straightforward to generalize. For any dimension d, there is a determinant test to decide if $d + 1$ points in $\mathbb{R}^d$ are in general position.

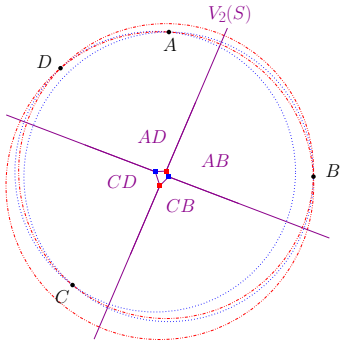
2.4 The Voronoi measure

Definition 2.15. Let $V_k(S)$ be the order-k Voronoi diagram of a point set S in the Euclidean plane. It is defined as a partition of the plane into (convex) cells whose points have the same k closest points of S.

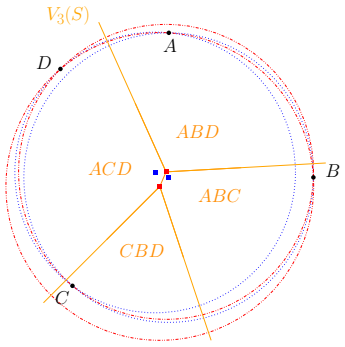
See Figure 9 for an example of Voronoi diagrams of different orders.



(a) Order-1 Voronoi diagram of four almost cocircular points.



(b) Order-2 Voronoi diagram, where the unbounded cells indicate the closest pair of points and the bounded cell has AC as the closest pair of points.



(c) Order-3 Voronoi diagram, where the unbounded cells indicate the closest triplet of points.

Figure 9: Voronoi diagrams of order 1,2 and 3. Centers of circles are depicted as red or blue squares.

It is well known that the order-1 Voronoi diagram of four cocircular points is composed of one vertex of degree four and four rays from it, the vertex being the center of the circle [7]. A slight perturbation making the cocircularity disappear leads to the Voronoi diagram being composed by a short segment joining two vertices of degree three and two rays from each of those vertices, where each vertex is the center of a circle through three of the four points considered, with none of them in the interior. See Figure 9a.

Note that, in addition to those two circles with centers at vertices of the Voronoi diagram, there are two more circles through three of the four points, containing the remaining fourth point in the interior. The centers of such circles are vertices of the order-2 and the order-3 Voronoi diagrams of the four points. See Figures 9b and 9c. Since in a general point set S it can happen that, out of the four circles defined by three points in a 4-tuple, none of them is empty of other points of S, we will also consider the order-k Voronoi diagrams for $k > 1$. It is well known that the endpoints of an edge of $V_k(S)$ are the centers of two circles with either $k-1$ or $k-2$ points of S inside the circle.

Definition 2.16. The Voronoi cocircularity $\mathcal{V}(A, B, C, D)$ is the length of the shortest edge among the Voronoi diagrams all orders of points A, B, C, D. If any vertex among the Voronoi diagrams has degree strictly greater than three, then $\mathcal{V}(A, B, C, D)$ is defined to be 0.

This measure is invariant under translation and rotation. But under scaling by a factor c, the length of the shortest Voronoi edge scales linearly with c. This means that the Voronoi and Thales cocircularity measures of a point set may be different. We show in Figure 10 an example where the set S under both measures doesn't yield the same four-tuple of points minimizing the cocircularity. It is no coincidence that the quadruplets minimizing the measures have circles of very different radius. As shown above with scaling, the Voronoi measure has a preference for small radius circles, the Thales measure is agnostic.

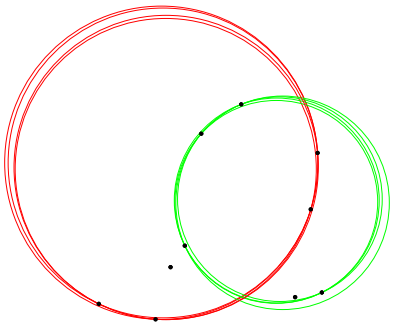


Figure 10: Green (resp., red) circles pass through the four most cocircular points of a set S of $n = 10$ points according to Voronoi (resp., Thales) measure.

The endpoints of an edge of $V_k(S)$ are the centers of two circles that each go through three points in S. Virtue of being an edge for some $V_k(S)$, it is also known that two of each three points are common to both circles. This two points that lie on both circles are very analogous to the supporting segment of the capable arcs in the Thales measure. Following the comparison, these other two remaining points that are not common to the two circles, are very analogous to the ones defining the capable arcs in the Thales measure.

We give a formula for the distance between the centers of two circles with two common points, in terms of the length of the supporting segment and the angles of the capable arcs. Hence, the following result also allows us to relate the length of an edge of with angles among the points.

Proposition 2.17. *Let A, B, C, D be four points in the plane. Let α and β be the angles $\angle CAD$ and $\angle CBD$, respectively. Let C_A, C_B be the centers of the circles through C, A, D and C, B, D, respectively. Then, if C_A and C_B are on the same side of the line $\overline{CD}$:*

$$|C_A C_B| = \frac{|CD|}{2}(\big||\cot\beta| - |\cot\alpha|\big|).$$

$$\textit{Otherwise, } |C_A C_B| = \frac{|CD|}{2}(|\cot\beta| + |\cot\alpha|).$$

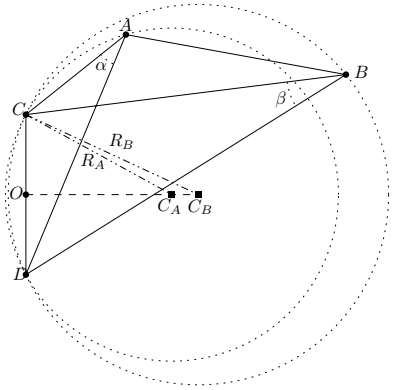


Figure 11: Illustration of Proposition 2.17.

Proof. Suppose, without loss of generality, that C_A and C_B are on the positive x-axis. Let d_1 and d_2 be, respectively, the distance from C_A and C_B to the origin O, see Figure 11. Then, by the Pythagorean Theorem, we know that

$$|C_A C_B| = d_2 - d_1 = \sqrt{R_B^2 - \left(\frac{|CD|}{2}\right)^2} - \sqrt{R_A^2 - \left(\frac{|CD|}{2}\right)^2}, \tag{1}$$

where R_A and R_B are the radii of the circles through C, A, D and C, B, D respectively.

It is well known that $R_A = \frac{|CD|}{2\sin\alpha}$ and $R_B = \frac{|CD|}{2\sin\beta}$. Thus, Equation (1) equals

$$\begin{aligned}
&\sqrt{\left(\frac{|CD|}{2\sin\beta}\right)^2 - \left(\frac{|CD|}{2}\right)^2} - \sqrt{\left(\frac{|CD|}{2\sin\alpha}\right)^2 - \left(\frac{|CD|}{2}\right)^2} \\
=&\frac{|CD|}{2}\left(\sqrt{\frac{1}{\sin^2\beta} - 1} - \sqrt{\frac{1}{\sin^2\alpha} - 1}\right) \\
=&\frac{|CD|}{2}\left(\sqrt{\csc^2\beta - 1} - \sqrt{\csc^2\alpha - 1}\right) \\
=&\frac{|CD|}{2}(|\cot\beta| - |\cot\alpha|).
\end{aligned}$$

However, if C_A and C_B are on different sides of the x-axis, $|C_A C_B| = d_2 + d_1$. Hence $|C_A C_B| = \frac{|CD|}{2}(|\cot\beta| + |\cot\alpha|)$. □

2.5 Algorithms

Another line of research is to detect these cocircularities. This intersects with a lot of applications in computer vision on detecting circles, from quality control in production lines all the way to biomedical image diagnostics, like classification of brain aneurysm type. Without going into detail, the circle Hough Transform (CHT) is a technique based on voting to find circles of known radius in images. For each point in the image (usually the edges as detected by a threshold Gaussian filter), it "votes" the pixels at the fixed radius distance as candidates for centers. If there is a circle with the fixed radius centered at one pixel, it will receive a lot of votes. This general technique of voting using an accumulator matrix[3] is very common in Hough Transforms. This inspires our approach on computing approximations of cocircularity.

The decision problem is: Are there 4 cocircular points? The optimization problem is: Find the 4-tuple of points that minimizes the measure.

2.5.1 Inversions

Proposition 2.18. *There is an $O(n^3)$ time algorithm that decides whether there exist four cocircular points in a set of n points, using inversions.*

An *inversion* transformation is determined by two parameters: The center of inversion O and the radius of inversion R. Two points P and P' are said to be inverses of each other if P and P' lie in the same half-line with origin in O and the Euclidean distances $|OP|$ and $|OP'|$ satisfy $|OP| \cdot |OP'| = R^2$.

[3] An accumulator matrix is just a matrix of the size of the image that aggregates all the votes cast by each pixel.

We need the following property of inversions:

Property 2.19. Given a center O and radius R of inversion, any circle containing O is inverted into a line.

So, if we perform inversion at a point A that is cocircular with B, C, D, then inverting B, C, D results in the three points being collinear. Property 2.19 allows us to propose Algorithm 1:

Algorithm 1 Detecting cocircularities with inversions

Input: set S of n points.
for every point $p_i \in S$ **do**
- Invert all points of $S \backslash \{p_i\}$ with respect to the center p_i and radius $r = 1$.
- Execute as subroutine a sweep-line algorithm to detect collinearities.

end for

The computational cost of detecting collinearities is $O(n^2)$, see Edelsbrunner and Guibas [14], and it is executed in a loop with $O(n)$ iterations. The total computational cost is $O(n^3)$ time and $O(n)$ space. We remark that inversions do not preserve a relationship between the measures of cocircularity and collinearity. Thus we are only able to solve the decision problem of detecting cocircularities using inversions.

2.5.2 Higher order Voronoi diagrams

Proposition 2.20. *The Voronoi cocircularity of a set S of n points in general position can be calculated in $O(n^3)$ time.*

The definition of the Voronoi cocircularity of S is the size of the shortest edge among $V_k(S)$, for $k = 1, \dots, n-1$. It is known that the $V_k(S)$, for all k, can be calculated in time $O(n^3)$, see [16, 23]. It then just remains to check the length of each edge in each Voronoi diagram. Since the number of edges in a Voronoi diagram of order k is at most $O(k(n-k))$, traversing all of them takes $O(n^3)$ time. In case S is not in general position, then the $V_k(S)$ for some k will have a vertex of degree strictly greater than three[4]. This can be detected once all higher order Voronoi diagrams of S have been obtained.

So the decision problem for detecting four cocircular points can be solved in $O(n^3)$ time using Voronoi diagrams. If the points are in general position, traversing all edges to find the shortest one solves the optimization problem for the Voronoi measure.

2.5.3 Approximations using Hashing

We present two more algorithms. We show that whether there exist four cocircular points in a set of n points can be decided in $O(n^3 \log n)$ time in the worst case or, using hashing, in average $O(n^3)$ time.

Both algorithms can find a solution, thus solving the optimization problem. The algorithms work by storing the radius and center of the circle defined by each triplet of points, and then detecting collisions. In the first case, by sorting the circles, which contributes with the additional factor of $\log n$.

The particularity is that these algorithms are not based on the measures we presented. But some relationship seems to exist between the measures of cocircularity presented and the difference in radius and center of two circles. As mentioned at the start of this Section 2.5, more practical methods like the circle Hough Transform approximate cocircularity through a discretization of the space of all possible centers and radii of the circles. This encourages our study of ϵ-approximations to cocircularity.

[4] We can informally consider vertices of degree strictly greater than three as edges of length 0.

Definition 2.21. Four points p_1, p_2, p_3, and p_4 are ϵ-*cocircular* if there are two circles $C_1 = (c_1, r_1)$, $C_2 = (c_2, r_2)$ (where c_i, r_i are the center and radius of C_i, resp., $1 \leq i \leq 2$) defined by two different selections of three points from $\{p_1, p_2, p_3, p_4\}$ such that $|c_1 - c_2| \leq \epsilon, |r_1 - r_2| \leq \epsilon$.

Proposition 2.22. *There is an algorithm working in $O(n^3)$ time on average that decides whether there exist four cocircular points in a set S of n points.*

Proof. Our algorithm uses hashing with separate chaining. In particular, H represents a hash table of point sets indexed by triples of points. Let f be the function that, given a set T of three points, returns the triple (x, y, r) s.t. (x, y) and r are the center and radius, resp., of the circle defined by the points in T. Consider the operations:

- **Insertion**: $H.insert(T)$ adds information T with key $f(T)$ to H.
- **Search**: $H.search(T)$ returns $\emptyset$ if H does not contain information for key $f(T)$ or a set R s.t. $f(T) = f(R)$, otherwise.

The method is shown in Algorithm 2. Since insertion and search have constant average cost, the for loop and the initialization of H give an $O(n^3)$ cost on average for the whole algorithm.

Algorithm 2 Exact cocircularity with hashing

```
Input: set S of n points.
let H be a hash table as above of size n^3
initialize H with all entries containing ∅
for every distinct T ∈ (S choose 3) do
    if H[T] = ∅ then
        H[T] ← T
    else
        return H[T] ∪ T
    end if
end for
return "no cocircular points"
```

□

Proposition 2.23. *There is an $O(n^3 \log n)$ algorithm that decides whether there exist four cocircular points in a set of n points.*

Proof. In Algorithm 3, the hash table of Algorithm 2 has been substituted by a vector V of size n containing 4-tuples (x, y, r, T), where n is the number of points; here T represents a set of three points, and (x, y) and r are the center and radius resp. of the circle defined by the points in T. We use a stable sorting algorithm (like *mergesort*) and apply it to V with respect to the first, second, and finally third component of the vector. Since the sorting algorithm is stable, after the three iterations the entries corresponding to the same circle are contiguous in V and can be detected in linear time. The cost of the first and third for loops is $O(n^3)$, while the second for loop has a cost $O(n^3 \log n^3) = O(n^3 \log n)$, which means that sorting is the determining cost factor of Algorithm 3 in the worst case.

□

Algorithm 3 Exact cocircularity without hashing

Input: set S of n points.
Let V be a size n^3 vector of tuples (x, y, r, T) for reals x, y, r and a three point set T
$i \leftarrow 1$
for every distinct $T \in \binom{S}{3}$ **do**
  Let (x, y) and r be the center and radius resp. of the circle defined by the points in T
  $V[i] \leftarrow (x, y, r, T)$
  $i \leftarrow i + 1$
end for
Sort(V)
for $i = 1$ to $n^3 - 1$ **do**
  if $V[i].x = V[i+1].x$ and $V[i].y = V[i+1].y$ and $V[i].r = V[i+1].r$ **then**
    return $V[i].T \cup V[i+1].T$
  end if
end for
return "no cocircular points"

Proposition 2.24. *There is an algorithm working in $O(n^3)$ time on average that, given n points in general position and $\epsilon > 0$, decides whether there exist four ϵ-cocircular points.*

This follows from slight modifications to the previous Algorithms 2 and 3. For all operations, considering discretizing the values of (x, y) and r to the closest value of the form $k \in \mathbb{Z}$ to $k \cdot 2\epsilon$. Also consider the discretization to the closes value $k \in \mathbb{Z}$ to $k \cdot 2\epsilon + \frac{\epsilon}{2}$. This corresponds to discretizing to the closest multiple of 2ϵ, and discretizing to the closes multiple plus an offset. If two values of (x, y) or r differ by at most ϵ, then have the same value after *either* discretizations. This results in some false positives, but we can easily discard them by checking the real difference of the values. See a visual representation of these discretizations in Figure 12.

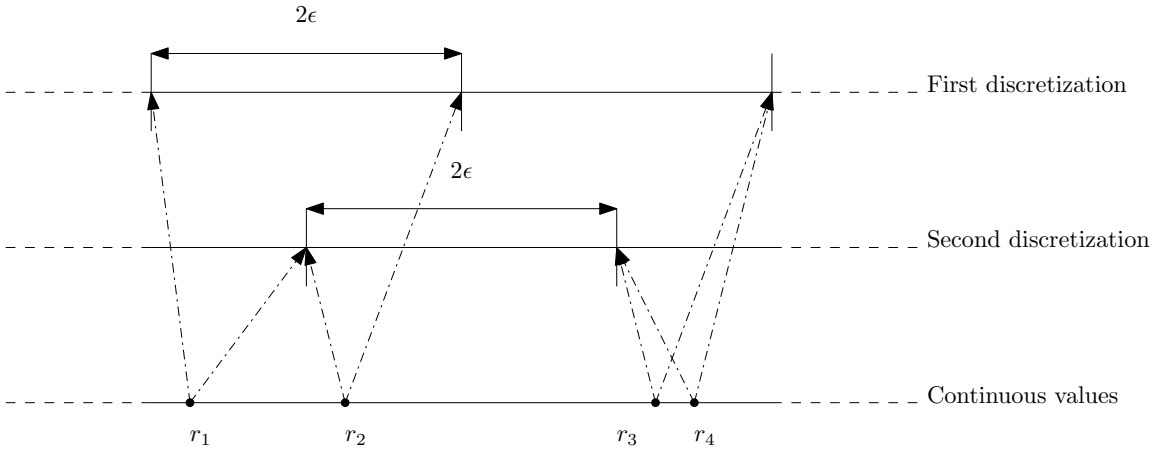


Figure 12: Two values that differ by at most ϵ are equal under at least one discretization.

So, executing the algorithms twice, once per discretization, comparing if a tuple is returned by either execution, and discarding false positives, is enough to detect ϵ-cocircularities.

2.6 Conclusions

We initiated the study of almost cocircularity in point sets. As an extension of analogous results in collinearity, we first discuss different measures for cocircularity that we considered to be very natural, though other measures could be studied as well. For each measure we present a different kind of result, this is because each measure is related to very different concepts.

For the *Thales* measure, we mainly showed an asymptotic upper bound of $O(1/n^2)$, and some preliminary work on a lower bound. Also some minor results on extremal configurations. For the *Determinant* measure we showed a reduction to 4SUM. This is specially relevant as it will also appear at the end of Subsection 3.3.1. For the *Voronoi* measure we showed an equation relating the an edge in a Voronoi diagram of with angles among the determining points.

After analytically studying the measures, we present several algorithms to compute the measures of cocircularity. We also present algorithms to decide if the points are in general position. And also present algorithms to decide if the points are in general position within a certain tolerance ϵ. This latter algorithms, some of witch use hashing, are specially suited to tackle the practical problems of detecting cocircularities in the setting of computer vision.

Several questions remain open, and we plan to continue this line of research. Some major ones already commented are:

Open problem 2.25. Is there a sub-cubic algorithm for the decision problem of detecting cocircularities?

Open problem 2.26. Can the bound for the Thales measure for convex point sets be extended to arbitrary point sets in general position?

Open problem 2.27. Find families of point sets that are "far away" from having four cocircular points.

3. Comparing bichromatic discrepancy for boxes and spheres

Let S be a bichromatic set of n points in d dimensions. Let R and B be a the set of red and blue points from S, respectively, so $S = R \cup B$. The coloring 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 $\mathcal{SH}$ as:

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

That is, the discrepancy of the shape is the number of red points minus the number of blue points of S contained in the shape, including the boundary. Let $\mathcal{F}$ be a family of shapes. The maximum bichromatic discrepancy of a family of shapes $\mathcal{F}$, where $\mathcal{SH} \in \mathcal{F}$, is defined as:

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

The main goal of this section is to compare existing algorithms and approaches to solve the problem of computing the maximum bichromatic discrepancy of a given set S using various families of shapes. The shapes considered here are boxes and disks, in various dimensions d. In the case of disks/spheres, we provide the natural extension to higher dimensions of known results by Bereg et al. [5].

3.1 Applications

The problem of computing discrepancy is present in several areas of computer science. Three major applications mentioned in the introduction of the paper by Dobkin et al. [13] are the Agnostic PAC-Learning, ϵ-Approximations, and Sampling Patterns in Graphics. In order to understand why is it worthwhile to consider and compare discrepancy over boxes and disks, we briefly summarize some of these applications.

PAC-Learning

In the case of Probably Approximately Correct - Learning, the geometric shape used in the computation of the discrepancy corresponds to the hypothesis that the model is trying to fit. Models in PAC-Learning try to choose a generalization function (or some parameters) from a family of functions provided to them. These functions are called hypothesis, as they should in broad terms predict the color (label) of the points.

Computing the discrepancy can be roughly understood as finding the parameters of the shape such that the data points are partitioned into highly monochromatic regions. We can measure this in a quantitative way by measuring the true error (or discrepancies) resulting from the hypothesis. For a more detailed definition of true error, and in general PAC-Learning, we refer to the already mentioned introduction of the paper by Dobkin et al. [13].

The insight is that simple hypothesis (axis-parallel rectangles) yield very good hypothesis (prediction-rules), so exploring discrepancy for basic shapes is of practical interest. This alone we find a sufficiently important application to explore non-axis-parallel rectangles, but maybe not disks, as they are hypothesis of substantially greater complexity.

Sampling Patterns and human vision

To justify our interest in studying the discrepancy using disks we turn to the last mentioned application, Sampling Patterns in Graphics. In the context of computer generated images, a very common technique is ray-tracing. This consists on approximating the image of a certain scenery by computing a discrete number of light-paths. Computing the true image would involve computing all the light paths present in the scene. And it would not be all that useful, as in the end the each pixel from the image is an interpolation of all light-paths crossing the pixel.

The main idea is that for each pixel we have a finite sample of light-paths from which we compute the color values of the pixel, but picking these samples in uniform patters yields visible biasing artifacts. The aim of studying discrepancy is that it has been shown to yield results in the design of these patters.

Related to images and vision, computing the maximal discrepancy corresponds to finding "blobs" or regions that have a high concentration of one color, that are chromatically unbalanced. The intuition is that these regions would strike the eye as monochromatic. This inspired our extension from squares to circles.

The square shape of pixels in digital sensors is closely related with the original results in axis-parallel boxes discrepancy. If we instead suppose 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 (blurring due to the cornea and lens) it suggests studying discrepancy on circles instead of rectangles. This visual metaphor of finding "blobs" also links to the study of measures of well-blended points.

There is a lot literature about discrepancy, and the books by Matousek [22] and Chazelle [8] cover in depth the topic. Related articles that also relate to the ideas we cover are Bereg et al. [6] and Díaz-Báñez et al. [9, 10].

3.2 Discrepancy for axis-parallel boxes

We first introduce the approach presented by Dobkin et al. [13], and Gunopulos [11], to compute the maximal boxes bichromatic discrepancy in $1d$ and $2d$. They develop two algorithms, the first for $1d$ solves the problem in optimal time if the input is sorted, and the second uses the first as a subroutine. This is relevant, as it turns out that indeed we can use the $1d$ algorithm to solve discrepancy is several related shapes. Formal proofs of the cited lemmas in this section are provided in the citations, for the sake of brevity we do not include them[5].

3.2.1 1d: Intervals

As boxes and disks both define intervals in the 1-dimensional ($1d$) case, the results from Dobkin et al. [13] apply to both shapes. We specially want to highlight some lemmas for the $1d$ case, as they provide the properties that are fundamental for the algorithms in further sub-sections.

Observation 3.1. [13] Given a set of points S in $1d$, any interval $[\ell, r]$ where $\ell, r \notin S$ can be shrunk until $\ell, r \in S$ without changing its discrepancy.

This first observation allows us to consider only intervals with endpoints in S, this discretizes and simplifies our search.

[5] The proofs are relatively trivial or intuitive and don't involve any idea or technique that is not mentioned in this section.

Lemma 3.2. *[13] Given an interval $[\ell, r]$ on our 1d setting, the discrepancy $\Delta([\ell, r]) = \Delta([\ell, m]) + \Delta((m, r])$ where $m \in [\ell, r]$ and $m \notin S$. See Figure 13.*

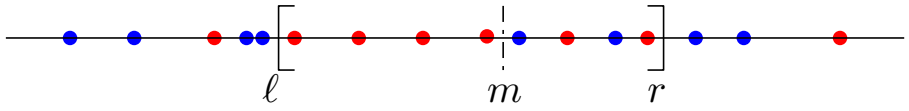


Figure 13: Example points in 1d with discrepancy 4.

This trivial observation is crucial for all the proposed algorithms. It can be rephrased verbosely as: "Computing the discrepancy of an interval can be divided (and conquered) by partitioning the interval into sub-intervals and computing their discrepancy". The recursive structure of computing the discrepancy is further elaborated later in Sub-section 3.2.2.

Lemma 3.3. *[13] Given an interval $[\ell, r]$ on our 1d setting, let the maximum discrepancy interval be $[a, b] \in [\ell, r]$. Then, for any $m \in [a, b]$, the interval $[a, m]$ maximizes the discrepancy among all intervals in $[\ell, m]$ that have m as the right endpoint. Analogously for the discrepancy among all intervals in $[m, r]$ with m as the left endpoint.*

Similarly to Lemma 3.2, this one states that "Computing the maximal discrepancy of an interval can also be divided into sub-interval computations". Lemma 3.3 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 3.4. [13] Given the maximum discrepancy of two consecutive intervals $[\ell, m]$ and $[m, r]$ and their maximum discrepancy with the fixed endpoint m, we can compute the maximum discrepancy for their union interval $[\ell, r]$ in constant time.

We can understand this observation as a way of computing the maximum discrepancy interval $[a, b]$ in Lemma 3.3. If we compute the maximum discrepancy of any interval with fixed right or left endpoint, it will allow us to merge it with adjoined intervals that have a common endpoint.

From Observation 3.4 we can intuitively see how we can build a tree of the points to represent all possible intervals. Because the basic object we are storing in the tree are intervals, the obvious implementation of a tree data-structure to use is a segment tree, or it's cousin the interval tree. For an overview of both data structures and more, see the staple book by Berg et al. [7]. Specifics on what tree implementation we use are not that relevant, as we just want to capture the hierarchical structure of the intervals. Regardless of the implementation of tree we choose, we will augment the data structure with the maximal discrepancy for each node of the tree. But it will also store the left/right fixed endpoint maximal discrepancy, finally realizing Observation 3.4.

Building this tree takes $O(n \log n)$ time and $O(n)$ space, allows insertions and deletions of points in $O(\log n)$ time, and queries of the discrepancy of an interval in $O(\log n)$ time. So the tree allows us to solve the static 1d maximum discrepancy, but this tree also allows for a online algorithm. Updates need only to traverse a $O(\log n)$-length path from the new leaf point (or deleted leaf) to the root of the tree. This allows us to solve the online 1d maximum discrepancy.

Theorem 3.5. *[13] Given a set S of n red and blue points, 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.*

3.2.2 2d: Boxes

The key strategy to tackle the $2d$ setting, is to find projections back to $1d$. The axis-parallel boxes is a perfect example of this technique. We could repeat most of the lemmas and observations from the previous Sub-section 3.2.1 adapted to $2d$ boxes, but instead go straight to stating the algorithm.

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 14.

Lemma 3.6. *[13] 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.*

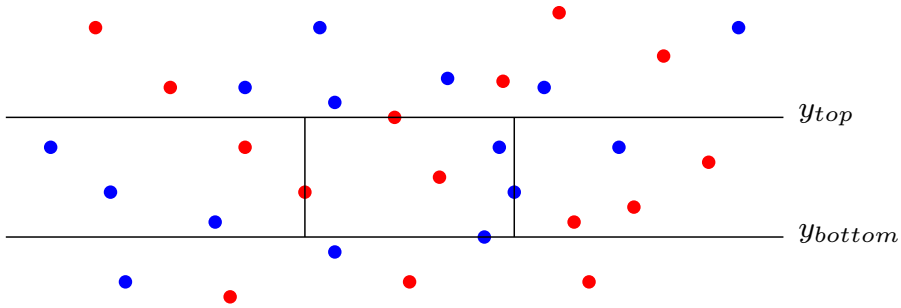


Figure 14: Discrepancy with boxes in $2d$.

This projection allow us to use the previous results to design algorithms. The only missing concept not used for this algorithm is to traverse the $\Theta(n^2)$ strips in order. In this case it would suffice to use the lexicographical order of the pairs of points. This would allow for the computation of the next strip as an update to the previous strip. This detail is important later in this section. Regardless of it, the following is an outline of the algorithm step by step:

Algorithm 4 Axis-parallel boxes discrepancy by Dobkin et al. [13]

Input: set S of n bichromatic points
Sort S by y-coordinate obtaining the order $p_1, \dots, p_n$.
for each p_i **do**
  Initialize the segment tree with only p_i, this represents the strip that only contains p_i.
  for each p_j such that $i < j$ **do**
    Update the segment tree by inserting p_j.
    If the new maximum discrepancy is larger than the seen so far, record the box.
  end for
end for

Sorting has cost $O(n \log n)$ time[6]. Updating has cost $O(\log n)$ time. Both loops have $O(n)$ iterations, so the total complexity is $O(n^2 \log n)$ time and $O(n)$ space.

[6]The for loops iterate in this order.

As a summary of the key ideas, we found a suitable projection from our $2d$ shape back to the $1d$ intervals, allowing us to use the static $1d$ algorithm. Furthermore, we found an ordering so that we could traverse efficiently the strips, such that from one strip to the next we needed only to perform insertion/deletion updates on the tree. This would allow us to also exploit the dynamic $1d$ algorithm, and it will become the main theme in the comparison between this previous results and the following ones.

This algorithm is straightforward to extend to higher dimensions, for each new dimension the "strip" is determined by an extra pair of points. In $2d$ the pair of points determines the interval of y-coordinates of the strip. In $3d$ an extra pair of points would determine the interval of z-coordinates of the strip. For each extra dimension above $1d$ we need to traverse all pairs of points, so the complexity is $O(n^{2(d-1)} \log n)$ time and $O(n)$ space.

Recall the relevant detail of traversing the strips in order. This results from the fact that the divide-and-conquer strategy we used on the x-axis can be fully replicated in the y-axis. This suggests dropping the quadratic for loop from Algorithm 4, and instead opting for a fully recursive algorithm. Still, the recursive algorithm would need to sort the input set of points by all its coordinates. This and more improvements where shown by Barbay et al. [4] to result in an $O(n^2)$-time algorithm, or even faster under some parametrizations of the input. They also extend this approach to higher dimensions, resulting in an $O(n^d)$ algorithm. This running time is tight up to sub-polynomial factors, as proven by Backurs et al. [3].

One important parametrization studied in Barbay et al. [4] relates with existing results on separability. They let δ be the number of (without loss of generality) horizontal strips that are monochromatic and pairwise disjoint.

Theorem 3.7. *[4] The Maximum-Weight Box problem admits a solution in $SORT(n) + O(\delta n)$ time and $O(n)$ space on instances of n points composed of δ strips.*

Here the weights refer to the values of the χ mapping we mention in the start of Section 3. The inclusion of different weights generalizes the bichromatic discrepancy problem, it allows for different weights per each point, so color is no longer relevant. And $SORT(n)$ refers to the time complexity of sorting the points, that differs under different models of computation; for the sake of simplicity we assume it to be $O(n \log n)$ time as in the comparison model.

As mentioned in Oliver and Seara [25] computing the number of parallel monochromatic strips that are pairwise disjoint is equivalent to computing the number of parallel lines needed to separate the point set S. This relationship of the parametrization by Barbay et al. [4] becomes even more relevant in the latter Section 3.4.

3.3 Discrepancy for disks

Disks and boxes are equivalent to intervals in $1d$, but disks in $2d$ do not satisfy the analogous of Lemma 3.2. There is no easy way to decompose a disk into smaller disks. Analogously, Lemma 3.3 doesn't hold. We can't divide-and-conquer, there is not a recursive structure we can exploit. Nevertheless, we can still apply the key strategy presented in Subsection 3.2.2, finding projections back to $1d$. The projection we present is equivalent to the one used by Bereg et al. [5].

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 15. In fact, because the centers are vertices of some Voronoi diagram $V_k(S)$, the bisector is divided into edges of some $V_k(S)$. Two consecutive segments in the bisector belong to consecutive $V_k(S)$ and $V_{k\pm1}(S)$. Regardless, we propose the following concept of *oriented angle* to sort all points with respect to to $\overline{p_i p_j}$.

Definition 3.8. 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 $\overrightarrow{p_i p_j}$, negative otherwise.

Recall the Inscribed Angle Theorem 2.6 and the capable arc Definition 2.7 from Subsection 2.2. The oriented angle corresponds to taking the supplementary angle of the capable arc through p_k supported on $\overline{p_i p_j}$, but with sign, to distinguish points from either side of the segment.

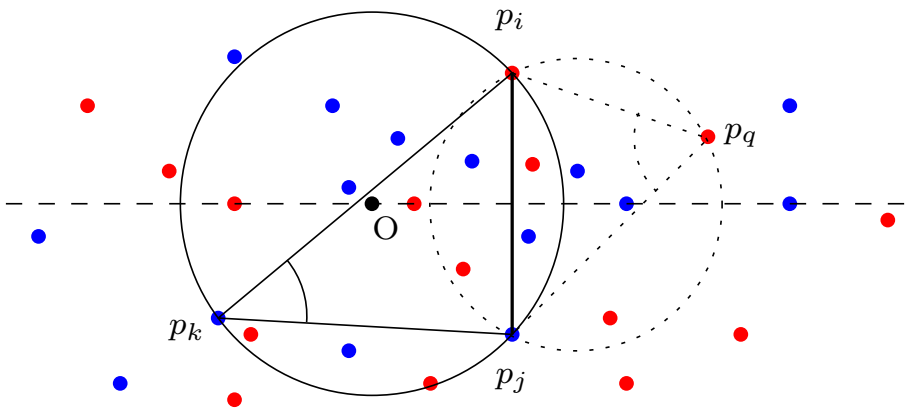


Figure 15: Discrepancy with disks in $2d$. Oriented angle α_k^{ij} is negative, and oriented angle α_q^{ij} is positive.

If we order the points in S by the oriented angle, the *furthest* left point in Figure 15 has the smallest oriented angle. The *furthest* right point has the largest oriented angle. Notice also that given two points p_k and p_q on the left half-plane defined by the line passing through p_i and p_j, we say that $p_k \preceq p_q$ if the radius of the circle passing through p_i, p_j, p_k is smaller that the radius of the circle passing through p_i, p_j, p_q, see Figure 16. This allows us to talk about the furthest left point. In a similar way, we can consider the order of the points on the right half-plane and talk about the furthest right point. It can be the case that several points on the left half-plane (or analogously on the right half-plane) are cocircular with respect to p_i and p_j, but this is not a problem because in one dimension there can be coincident points.

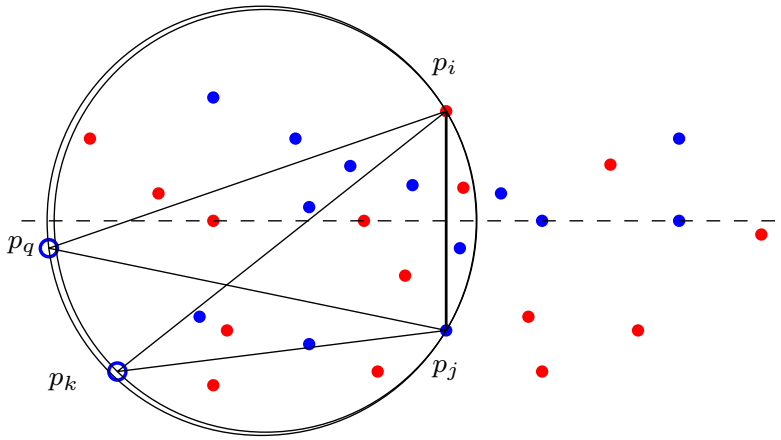


Figure 16: Sorting the points with respect to p_i and p_j: $p_k \preceq p_q$.

Points *close* to segment $\overline{p_i p_j}$ have oriented angle close to 0. Traversing all the points $p_k \in S$ by their oriented angle $\alpha_k^{ij} = \pm(\pi - \angle p_i p_k p_j)$, from smallest to greatest, can be visualized as sliding the center O of a disk over the bisector of $\overline{p_i p_j}$, from left to right.

For the fixed points $p_i, p_j \in S$, let

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

be the list (in fact, a multi-set) of oriented angles of the points p_k with respect to p_i, p_j. Each α_k^{ij} retains the color of the point it represents. If the oriented angle wasn't define to take the supplementary angle of $\angle p_i p_k p_j$, then the order of the list Λ^{ij} wouldn't correspond to the order induced by sliding the center O over the bisector.

Let $[\ell, r]_{\Lambda^{ij}} = [\forall \alpha_k^{ij} \in \Lambda^{ij} : \ell \leq \alpha_k^{ij} \leq r] \subseteq \Lambda^{ij}$, where $\ell, r \in [-\pi, \pi]$ and $[\ell, r]_{\Lambda^{ij}}$ is the sub-list of oriented angles that lie inside the $[\ell, r]$ interval.

Definition 3.9. The inverse of an angle α_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.

It follows from the Inscribed Angle Theorem 2.6 that the intervals of angles corresponding to disks 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 β, and also oriented angle β, where β 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 Λ^{ij}.

Lemma 3.10. *The disk discrepancy of Λ^{ij} is equal to a constant with respect to β, plus the interval discrepancy of the inverse angles with fixed endpoint 0.*

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

Proof. The proof is straightforward from Lemma 3.2, and the constant results easily from the inversion process. □

Definition 3.11. 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 3.10 we have the following result.

Theorem 3.12. *The disk discrepancy of a list of angles is equal to the interval discrepancy of its projection, with the restriction of containing the interval $[-\pi, 0]$,*

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

Proof. Notice that as we point out above, the inversion is defined to change only the points with negative oriented angle. So the projection $\mathcal{P}$ will only affect those points. The projection has joined the two terms of the sum in Lemma 3.10 □

Theorem 3.12 now provides a tool to solve the maximum discrepancy of disks in $2d$ by solving maximum discrepancy of intervals in $1d$. Similar tools allow to solve the discrepancy of lunes, but this will not be elaborated for the sake of brevity. Further work in progress is being done on applying analogous techniques to solve the ellipse/conic discrepancy.

3.3.1 Algorithm for disk discrepancy

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. The remaining points can be sorted by their oriented angle with respect to the fixed pair. These angles are then projected via $\mathcal{P}$. Using this projection, the algorithm is straightforward:

Algorithm 5 Disk discrepancy

Input: set S of n bichromatic points.
for each pair $(p_i, p_j) \in S \times S$ such that $i \neq j$ **do**
  Compute the list of oriented angles Λ^{ij} of all points with respect to p_i, p_j.
  Compute the projection $\mathcal{P}(\Lambda^{ij})$.
  Sort Λ^{ij}
  Compute the maximum discrepancy interval with the restriction: $\Delta(\mathcal{P}(\Lambda^{ij})) = [-\pi, \beta], \quad \beta \in [0, \pi]$.
end for

To compute the maximum discrepancy interval with the fixed left endpoint, it would be enough to modify slightly the algorithm for interval discrepancy. Recall that we already stored this information in the tree. But fixing the left endpoint simplifies the $1d$ case dramatically, and a simple traversal of all right endpoints it is enough.

Thus, computing the list of oriented angles and their projection costs $O(n)$ time. Sorting the oriented angles costs $O(n \log n)$ time, and computing the maximum discrepancy interval costs $O(n)$ time. The loop has $O(n^2)$ 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. [5].

In comparison to this approach, a faster and more general algorithm exists by Dobkin and Eppstein [12]. 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. [15]. Their resulting complexity is $O(n^3)$ time for disks, and $O(n^5)$ time for ellipses, and both with $O(n)$ space. The drawback is that in practice implementing a topological sweep can be non-trivial. Even if the algorithm we presented is a $O(\log n)$ factor slower, it may be easier to implement in practice for dimension $d = 2$, and also for any dimension $d \geq 3$.

It is straightforward to generalize Algorithm 5 and the one by Dobkin and Eppstein [12] to higher dimensions $d \geq 3$. See Figure 17 for a visual example in $d = 3$. The respective complexities are $O(n^{d+1} \log n)$ and $O(n^{d+1})$ time and $O(n)$ space.

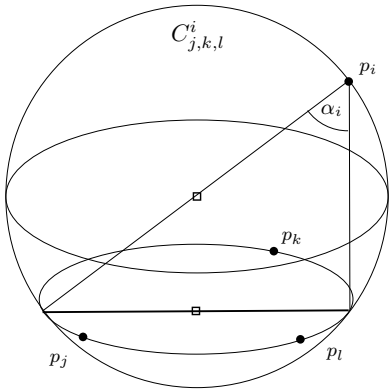


Figure 17: Generalization of oriented angle in 3d.

We think that finding the maximum discrepancy disk in $2d$ could be 4SUM hard. The reduction can be done using Proposition 2.14 from the Subsection 2.3.1 of this thesis, and Theorems 6, 7 and Lemma 12 in Bereg et al. [6]. We are almost done with the proof of this result, but we leave it as further work for a journal version of this research.

3.4 Discrepancy for non-axis-parallel 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 3.13. *Computing the maximum discrepancy box in a fixed strip with a given direction $\vec{v}$ is equivalent to finding the maximum discrepancy interval of the points inside the strip projected onto a directed line with direction $\vec{v}$.*

This is just a generalization of Lemma 3.6, and allows us to reuse the results for axis-parallel boxes.

Lemma 3.14. *A set S of n points in $2d$ has $O(n^2)$ non-equivalent linear projections onto $1d$.*

Notice that the number of non-equivalent projections is at most the half of the number of lines defined by two bichromatic points. See Figure Figure 18.

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 $O(n^4)$ time. We are currently studying other approaches to improve this trivial complexity. We present a brute force trivial algorithm:

The last instruction is computed by executing the algorithm presented in Barbay et al. [4], in $O(n^2)$ time and $O(n)$ space. The loop has $O(n^2)$ iterations so the total complexity is $O(n^4)$ time and $O(n)$ space.

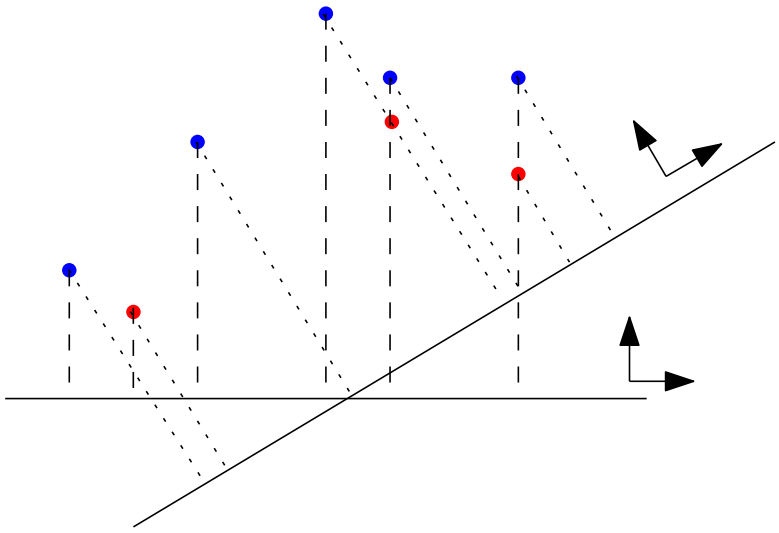


Figure 18: Two projections on a line.

Algorithm 6 Non-axis-parallel boxes discrepancy

Input: set S of n bichromatic points.
for each pair $(p_i, p_j) \in S \times S$ such that $i \neq j$ **do**
  Compute the clockwise "closest" direction to $\vec{v} = \overrightarrow{p_i p_j}$ such that $p_{i_x} \neq p_{j_x}$.
  Rotate the axes till the x-axis is parallel to $\vec{v}$.
  Compute the maximum discrepancy axis-parallel box.
end for

3.4.1 Some observations for non-axis-parallel boxes

As mentioned in Oliver and Seara [25], computing the minimum number of parallel and pairwise disjoint monochromatic strips for a given bichromatic set S is equivalent to computing the minimum number of parallel lines needed to separate the point set S into monochromatic subsets. This relationship with the parametrization by Barbay et al. [4] becomes even more relevant by the following discussion.

The relationship is that computing the parameter δ in Barbay et al. [4] is a well known problem in separability. As mentioned in Oliver and Seara [25], there is an algorithm that computes for all directions $\vec{v}$, the number of lines parallel to $\vec{v}$ that are needed to separate the points into monochromatic strips, i.e., a generalized parameter δ. This algorithm sweeps the point-line dual, keeping monochromatic-levels to count the monochromatic strips. Independently of these details, its time complexity is $O(n^2 \log n)$, so it is not worth it to precompute the parameter δ.

The relevant connection is that, if there is a faster algorithm running in $O(\delta n \log n)$ time, then it would be possible to precompute δ with no overhead[7].

Notice that on non-axis-parallel boxes computing the parameter δ can not be done in linear time per dimension with a trivial traversal of the points.

[7] Recall we are assuming that the cost of sorting SORT(n) in Barbay et al. [4] is $O(n \log n)$

3.5 Conclusions

In Subsection 3.2.2 we followed the key strategy by Dobkin et al. [13]. We found that there are two relevant parameters that condition the time complexity:

- The first parameter is the number of non-equivalent $1d$ projections we extract from the input. This constitutes our search space.
- The second parameter whether there is an order in this search space. If this order allows for a constant number of updates from one $1d$ projection to the next, then we can use the dynamic algorithm for $1d$ intervals. This incurs a $O(\log n)$ factor. Otherwise we must recompute from scratch the discrepancy of each $1d$ projection using the static algorithm. This incurs an extra $O(n)$ factor with respect to the dynamic algorithm.

We can evaluate these parameters for the shapes studied to show how the complexities are related.

Shape	Search space	Order	Time complexity	Space complexity
Boxes	$O(n^2)$	Yes	$O(n^2 \cdot \log n)$	$O(n)$
Disks	$O(n^2)$	No	$O(n^2 \cdot n \log n)$	$O(n)$
Non-axis-parallel Boxes	$O(n^4)$	Yes	$O(n^4 \cdot \log n)$	$O(n)$

But, as cited above, better algorithms exist in the literature. The following table illustrates the time complexities of the best known algorithms for computing the discrepancies for the shapes studied.

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

Possible further work is finding a better method, maybe exploiting advanced data structures in the $2d$ setting. Specifically in the case of non-axis-parallel 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.

In a unexpected coincidence, lots of concepts already seen in Section 2 reappeared. Mostly because of the use of concepts like the capable arc, but also in the case of the 4SUM-hardness, or the topological sweep and paraboloid projection. This highlights how the same conceptual tools are reused from one problem to the other.

Lots of open questions remain open, but some of the ones mentioned above are:

Open problem 3.15. Is there a faster non-trivial algorithm for non-axis-parallel boxes in $d \geq 2$?

Open problem 3.16. Formally prove that the maximum bichromatic disk discrepancy problem in $2d$ is 4SUM-hard.

Open problem 3.17. Relate the discrepancy of a set of points with the monochromatic fraction of edges in the Delaunay triangulation of the set of points. Is this a suitable measure of "well-blended points"?

4. Bibliography

[1] A. de las Heras, G. Esteban, D. Garijo, C. Huemer, A. Lozano, N. Oliver and D. Orden. Measuring cocircularity in a point set. *XX Spanish Meeting on Computational Geometry, EGC'23*, Santiago de Compostela, Spain, 3-5 July, (2023).

[2] A. Suk. On the Erdős-Szekeres convex polygon problem. *Journal of the American Mathematical Society*, Voll. 30(4), (2016), pp. 1047–1053. https://doi.org/10.1090/jams/869

[3] A. Backurs, N. Dikkala and C. Tzamos. Tight hardness results for maximum weight rectangles. *ArXiv abs/1602.05837*, (2016).

[4] 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.

[5] S. Bereg, O. Daescu, M. Zivanic and T. Rozario. Smallest maximum-weight circle for weighted points in the plane. *15th International Conference on Computational Science and Applications, (ICCSA 2015)*, Banff, Alberta, Canada, (2015), pp. 244–253.

[6] 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.

[7] M. de Berg, O. Cheong , M. van Kreveld, and M. Overmars. *Computational Geometry*. Springer Berlin Heidelberg, (2008). https://doi.org/10.1007/978-3-540-77974-2.

[8] B. Chazelle. *The discrepancy method in computational geometry*. Handbook of Discrete and Computational Geometry, CRC Press 44, (2004), pp. 983–996.

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

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

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

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

[13] 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.

[14] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, 1986, pp. 389–403.

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

[16] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15(2), (1986), pp. 341–363.

[17] P. Erdős and G. Szekeres. On some extremum problems in elementary geometry. *Ann. Univ. Sci. Budapest*, 3-4, (1960/1961), pp. 53–62.

[18] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Computational Geometry: Theory and Applications*, 5(3), (1995), pp. 165–185. https://doi.org/10.1016/0925-7721(95)00022-2.

[19] J. García-López, P. A. Ramos, and J. Snoeyink. Fitting a set of points by a circle. *Discrete and Computational Geometry*, 20, (1998), pp. 389–402.

[20] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams, *ACM Trans. Graph.* , 4(2), (1985), pp. 74–123

[21] T. Kopelowitz, S. Pettie, and E. Porat. Higher lower bounds from the 3SUM conjecture. *In Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms (SODA '16)*, pp. 1272–1287.

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

[23] K. Mulmuley. On levels in arrangements and Voronoi diagrams. *Discrete and Computational Geometry*, 6, (1991), pp. 307–338.

[24] N. Oliver and C. Seara. Comparing square and circular bi-chromatic discrepancy. *XX Spanish Meeting on Computational Geometry, EGC'23*, Santiago de Compostela, Spain, 3-5 July, (2023).

[25] N. Oliver and C. Seara. On strip separability. *XX Spanish Meeting on Computational Geometry, EGC'23*, Santiago de Compostela, Spain, 3-5 July, (2023).