Testing Polynomials for Vanishing on Cartesian Products of Planar Point Sets

Boris Aronov
Department of Computer Science and Engineering, Tandon School of Engineering, New York University, Brooklyn, NY 11201, USA
boris.aronov@nyu.edu

Esther Ezra
School of Computer Science, Bar Ilan University, Ramat Gan, Israel
ezraest@cs.biu.ac.il

Micha Sharir
School of Computer Science, Tel Aviv University, Israel
michas@tau.ac.il

Abstract

We present subquadratic algorithms, in the algebraic decision-tree model of computation, for detecting whether there exists a triple of points, belonging to three respective sets A, B, and C of points in the plane, that satisfy a certain polynomial equation or two equations. The best known instance of such a problem is testing for the existence of a collinear triple of points in $A \times B \times C$, a classical 3SUM-hard problem that has so far defied any attempt to obtain a subquadratic solution, whether in the (uniform) real RAM model, or in the algebraic decision-tree model. While we are still unable to solve this problem, in full generality, in subquadratic time, we obtain such a solution, in the algebraic decision-tree model, that uses only roughly $O(n^{28/15})$ constant-degree polynomial sign tests, for the special case where two of the sets lie on one-dimensional curves and the third is placed arbitrarily in the plane. Our technique is fairly general, and applies to any other problem where we seek a triple that satisfies a single polynomial equation, e.g., determining whether $A \times B \times C$ contains a triple spanning a unit-area triangle.

This result extends recent work by Barba et al. [4] and by Chan [7], where all three sets A, B, and C are assumed to be one-dimensional. While there are common features in the high-level approaches, here and in [4], the actual analysis in this work becomes more involved and requires new methods and techniques, involving polynomial partitions and other related tools.

As a second application of our technique, we again have three n-point sets A, B, and C in the plane, and we want to determine whether there exists a triple $(a, b, c) \in A \times B \times C$ that simultaneously satisfies two real polynomial equations. For example, this is the setup when testing for the existence of pairs of similar triangles spanned by the input points, in various contexts discussed later in the paper. We show that problems of this kind can be solved with roughly $O(n^{24/13})$ constant-degree polynomial sign tests. These problems can be extended to higher dimensions in various ways, and we present subquadratic solutions to some of these extensions, in the algebraic decision-tree model.

2012 ACM Subject Classification Theory of computation; Theory of computation → Computational geometry

Keywords and phrases Algebraic decision tree, Polynomial partition, Collinearity testing, 3SUM-hard problems, Polynomials vanishing on Cartesian products

Digital Object Identifier 10.4230/LIPIcs.SoCG.2020.8

Related Version A full version of the paper is available at `https://arxiv.org/abs/2003.09533`.

Funding *Boris Aronov*: Partially supported by NSF grant CCF-15-40656 and by grant 2014/170 from the US-Israel Binational Science Foundation.
Esther Ezra: Partially supported by NSF CAREER under grant CCF:AF-1553354 and by Grant 824/17 from the Israel Science Foundation.

36th International Symposium on Computational Geometry (SoCG 2020).
Editors: Sergio Cabello and Danny Z. Chen; Article No. 8; pp. 8:1–8:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Micha Sharir: Partially supported by ISF Grant 260/18, by grant 1367/2016 from the German-Israeli Science Foundation (GIF), and by Blavatnik Research Fund in Computer Science at Tel Aviv University.

Acknowledgements The authors wish to thank Adam Sheffer and Frank de Zeeuw for suggesting the transformation used for collinearity testing in Theorem 6.1. We also thank Jean Cardinal, John Iacono, Stefan Langerman, and Aurélien Ooms for useful discussions.

1 Introduction

General background. Let A, B, and C be three n-point sets in the plane. We want to determine whether there exists a triple of points $(a, b, c) \in A \times B \times C$ that satisfy one or two prescribed polynomial equations. An example of such a scenario, with a single vanishing polynomial, is to determine whether $A \times B \times C$ contains a collinear triple of points. This classical problem is at least as hard as the 3SUM problem [14], in which we are given three sets A, B, and C, each consisting of n real numbers, and we want to determine whether there exists a triple of numbers $(a, b, c) \in A \times B \times C$ that add up to zero.

The 3SUM problem itself, conjectured for a long time to require $\Omega(n^2)$ time, has recently been shown by Grønlund and Pettie [16] (with further improvements by Chan [7]) to be solvable in very slightly subquadratic time. Moreover, in the *linear decision-tree model*, in which we only count linear sign tests performed on the input points (and do not allow any other operation to access the input explicitly), Grønlund and Pettie have improved the running time to nearly $O(n^{3/2})$ (see also [13, 15] for subsequent slight speedups), which has been drastically further improved (still in the linear decision-tree model) to $O(n \log^2 n)$ time by Kane et al. [18].

In contrast, no subquadratic algorithm is known for the collinearity detection problem, either in the standard real RAM model (also known as the *uniform* model) or in the decision-tree model; see [4] for a discussion. In the uniform model, the problem can be solved in $O(n^2)$ time. The primitive operation needed to test for collinearity of a specific triple is the so-called *orientation test*, in which we test for the sign of a quadratic polynomial in the six coordinates of a triple of points in $A \times B \times C$ (see Eq. (1) below for the concrete expression). Consequently, it is natural (and apparently necessary) to use the more general *algebraic decision-tree model*, in which each comparison is a sign test of some constant-degree polynomial in the coordinates of a constant number of input points; see [6, 20] and below.

The problems, in more detail

In this paper we consider several variants of testing a polynomial, or polynomials, for vanishing on a triple Cartesian product. The main motivation for the present study is the aforementioned collinearity testing question. We present the problem in a wider context, where we are given three sets A, B, and C, each consisting of n points in the plane, and we consider two scenarios:

(a) **A single vanishing polynomial.** Given a single constant-degree irreducible 6-variate polynomial F, determine whether there exists a triple $(a, b, c) \in A \times B \times C$ such that $F(a, b, c) = 0$.

(b) **A pair of vanishing polynomials.** Given a pair F, G of constant-degree irreducible 4-variate polynomials, determine whether there exists a triple $(a, b, c = (c_1, c_2)) \in A \times B \times C$ such that $c_1 = F(a, b)$ and $c_2 = G(a, b)$.

We begin by studying the vanishing pair problem in (b), because our results are stronger for this setup, and show that, as can be expected, requiring the triple (a, b, c) to satisfy two equalities facilitates a more efficient solution. In contrast, the collinearity testing problem, as well as more general instances of a single vanishing polynomial in (a), seem harder to solve efficiently. As we spell out below, we can solve problems of the latter kind in subquadratic time, in the algebraic decision-tree model, only for restricted input sets.

We note that the vanishing pair problem in (b) is a special case of a more general question, in which F and G are 6-variate real polynomials, and the equations that we want to satisfy are $F(a, b, c) = G(a, b, c) = 0$. This general setting can also be handled by a more involved variant of the technique presented here, using standard tools from real algebraic elimination theory (as in [5]), but we will not consider this extension in the paper. (See also [4] for the treatment of this issue in a different, and simpler, context.)

A special (but natural) case of the problem with two polynomial constraints is where each of the sets A, B, C consists of n complex numbers, and we want to test the vanishing of a *single* constant-degree bivariate polynomial $H\colon \mathbb{C}^2 \to \mathbb{C}$ defined over the complex numbers; this is an extension of the problem studied by Barba et al. [4] over the reals. That is, the problem is to determine whether there is a triple $(a, b, c) \in A \times B \times C$ such that[1] $c = H(a, b)$. Two concrete instances of this question, involving testing for the existence of similar triangles that are determined by A, B, and C, will be used to present our technique.

Comments on the purely one-dimensional setup. Questions of the type studied here are (extensions to higher dimensions of) the algorithmic counterparts of the classical problems in combinatorial geometry, studied by Elekes and Rónyai [10] and by Elekes and Szabó [11], and later improved in [22, 21]. We comment on this connection in some detail in the full version of the paper [3]. In the setup studied in [10, 11, 22, 21], all three sets A, B, C are one-dimensional. This purely one-dimensional setup has recently been studied by Barba et al. [4], both for the algebraic decision tree model and for the uniform model, whose algorithm in the algebraic decision tree model runs in close to $O(n^{12/7})$ time. The same approach, combined with more involved algorithmic techniques, yields an algorithm in the uniform model that runs in $O(n^2(\log\log n)^{3/2}/\log^{1/2} n)$ time, which has been slightly improved to $O(n^2(\log\log n)^{O(1)}/\log^2 n)$ by Chan [7].

Given this apparent (polynomial) hardness of computation, our goal is thus to obtain a significantly subquadratic solution in the *algebraic decision-tree model*. Here we only pay for sign tests that involve the input point coordinates, where each such test determines the sign of some real polynomial of constant degree in a constant number of variables. All other operations cost nothing in this model, and are assumed not to access the input explicitly. For example, each orientation test used in collinearity detection examines the sign of the determinant (a quadratic polynomial in $a_1, a_2, b_1, b_2, c_1, c_2$)

$$\begin{vmatrix} 1 & a_1 & a_2 \\ 1 & b_1 & b_2 \\ 1 & c_1 & c_2 \end{vmatrix}, \tag{1}$$

for some triple of points $(a_1, a_2) \in A$, $(b_1, b_2) \in B$, $(c_1, c_2) \in C$.

[1] Over the reals, H induces two polynomial equations, one for the real part and the other for the imaginary part, so this is indeed a special case of the polynomial vanishing pair problem. Here too one may consider the more general case where H is trivariate and we test for $H(a, b, c) = 0$.

Concrete problems in the two-dimensional setup. Each of the two general questions studied here (of one or two vanishing polynomials) arises in various concrete problems in computational geometry. For the case of a single vanishing polynomial, collinearity testing is a fairly famous (or should we say, notorious) example. Other problems include testing for the existence of a triangle Δabc, for $(a, b, c) \in A \times B \times C$, that has a given area, or perimeter, or circumscribing disk of a given radius, and so on.

We consider two simple instances of the vanishing polynomial pair problem. In the first one, we are given sets A, B, and C, each of n points in the plane, none of which contains the origin, and wish to determine whether there exists a triple $(a, b, c) \in A \times B \times C$ such that the triangle spanned by o (the origin), b and c is similar to the triangle spanned by o, $e_1 = (1, 0)$, and a (with e_1 corresponding to b and a to c). As a matter of fact, and as easily verified, this instance can also be interpreted as having three sets A, B, C of *complex* numbers, and the goal is to determine whether there exists a triple $(a, b, c) \in A \times B \times C$ such that $c = ab$. This is a single complex quadratic equation that (a, b, c) has to satisfy, which translates to two real quadratic equations in the coordinates of a, b, c, when treated as points in the real plane.

In the second instance, we are given sets A, B, and C, each of n points in the plane, and a fixed triangle $\Delta = \Delta uvw$, and we want to determine whether there exists a triple $(a, b, c) \in A \times B \times C$ that spans a triangle similar to Δ, with a corresponding to u, b to v, and c to w. This instance too can be interpreted as having three sets A, B, C of *complex* numbers, and the goal is to determine whether there exists a triple $(a, b, c) \in A \times B \times C$ that satisfies a certain single *linear* equation over complex numbers, determined by Δ; see the full version [3] for these details.

We note that the first instance can also be turned into an instance that involves a single complex *linear* equation, simply by replacing every $z \in A \cup B \cup C$ by $\ln z$. As it turns out, complex linear equations can be handled more efficiently (see below for details), but we use these examples as showcases of the more general technique that we develop.

In the full version of this paper [3] we show that both versions of the triangle similarity testing problem are 3SUM-hard.

▶ Remark 1.1. Since the underlying vanishing complex polynomial can be assumed to be *linear* (in both instances), the analysis in the very recent work of Aronov and Cardinal [2] implies that both problems can be reduced to the classical real-3SUM problem, via a random projection technique, and then solved, in the linear decision-tree model, in $O(n \log^2 n)$ time, using the technique of Kane et al. [18].

Our results. After setting up the technical machinery that our analysis requires, in Sections 2 and 3, we first consider, in Section 4, the problem of testing for a vanishing pair of polynomials, which includes the triangle similarity testing problems. We show that such problems can be solved, in the algebraic decision-tree model, with $O(n^{24/13+\varepsilon})$ polynomial sign tests, for any $\varepsilon > 0$ (with the constant of proportionality depending on ε), where each test involves a polynomial of constant degree in a constant number of variables, which in general might be more involved than just orientation tests. For the analysis, we need to assume that the pair of polynomials F, G have "good fibers" (which they do in the triangle similarity testing problems) – see Sections 2 and 4 for details.

We then consider, in Section 5, the problem of '$2 \times 1 \times 1$-dimensional' collinearity testing, meaning that A is an arbitrary set of points in the plane, but each of B and C lies on some respective constant-degree algebraic curve γ_B, γ_C. We show that this restricted problem can be solved in the algebraic decision-tree model with $O(n^{28/15+\varepsilon})$ polynomial sign tests, for any

$\varepsilon > 0$ (where again the constant of proportionality depends on ε). The technique extends naturally to similar problems involving a single vanishing polynomial, such as determining whether $A \times B \times C$ spans a unit-area triangle.

We still do not have a subquadratic solution, even in the algebraic decision-tree model, to the unconstrained (referred to as $2 \times 2 \times 2$-dimensional) collinearity testing problem, or even for the more restricted $2 \times 2 \times 1$-dimensional scenario, where only C is constrained to lie on a given curve. The techniques that we use for the $2 \times 1 \times 1$ version can be extended to the general unconstrained (or less constrained) case, but they actually result in a *superquadratic* algorithm; see the full version of the paper for more details. As shown by Erickson and Seidel [12], if the *only* sign tests that we allow in the decision tree are orientation tests, then $\Omega(n^2)$ tests are needed in the worst case. The solution presented here uses other sign tests, making it more powerful (and more efficient).

We also consider, in the full version, extensions of both problems to higher dimensions. Specifically, we study collinearity testing in d dimensions, where we assume that each of B and C lies on a hyperplane. Our solution is based on projections of the input onto lower-dimensional subspaces, and achieves the same asymptotic performance as in the plane. This result is presented in Section 6. More general extensions to higher dimensions are discussed in the appendix of the full version [3].

▶ Remark 1.2. We are not aware of a simple extension of the analysis in the earlier work of Barba et al. [4] or of Chan [7] to the problems studied in this paper. A main technique in our arsenal is to consider the Cartesian product of polynomial partitionings, which we believe to be essential, mainly for the triangle similarity problems and their higher-dimensional extensions, as well as to higher-dimensional extensions of collinearity testing.

The $2 \times 1 \times 1$ case of problems involving a single vanishing polynomial, considered in Section 5, has an alternative subquadratic, albeit less efficient, solution, using simpler considerations, which somewhat resemble the analysis in [4]. We sketch this alternative technique in the full version of the paper [3]

We also comment that Chan [7] addresses several related geometric 3SUM-hard problems, among which is a variant of dual collinearity testing: Given three sets A, B, and C of line segments in the plane, where the segments in A are pairwise disjoint, and so are the segments in B, decide whether there exist a triple of segments in $A \times B \times C$ that meet at a common point. Although Chan's technique results in a slightly subquadratic algorithm in the RAM model, and is also claimed to yield a truly subquadratic algorithm in the algebraic decision-tree model, the disjointness assumptions significantly restrict the problem, so, to quote [7], "it remains open whether there is a subquadratic algorithm for the degeneracy testing for n lines in $\mathbb{R}^2$."

2 Preliminaries

Our analysis relies on planar polynomial partitioning and on properties of Cartesian products of pairs of them. For a polynomial $f\colon \mathbb{R}^d \to \mathbb{R}$, for any $d \geq 2$, the *zero set* of f is $Z(f) := \{x \in \mathbb{R}^d \mid f(x) = 0\}$. We refer to an open connected component of $\mathbb{R}^d \setminus Z(f)$ as a *cell*. The classical Guth-Katz result is:

▶ **Proposition 2.1** (Polynomial partitioning; Guth and Katz [17]). *Let P be a finite set of points in $\mathbb{R}^d$, for any $d \geq 2$. For any real parameter D with $1 \leq D \leq |P|^{1/d}$, there exists a real d-variate polynomial f of degree $O(D)$ such that $\mathbb{R}^d \setminus Z(f)$ has $O(D^d)$ cells, each containing at most $|P|/D^d$ points of P.*

Agarwal, Matoušek, and Sharir [1] presented an algorithm that efficiently computes[2] such a polynomial f, whose expected running time is $O(nr + r^3)$, where $r = D^d$.

Note that the number of points of P on $Z(f)$ can be arbitrarily large. For planar polynomial partitions, though, this can be handled fairly easily, by partitioning the algebraic curve $Z(f)$ into subarcs, each containing at most $|P|/D^2$ points (as do the complementary cells). We state this property formally and spell out the easy details in the full version of the paper.

Polynomial partitioning for Cartesian products of point sets in the plane. Solymosi and De Zeeuw [23] studied polynomial partitioning for Cartesian products of planar point sets. Given two finite sets P_1 and P_2 of points in the plane, a natural strategy to construct a partitioning polynomial for $P_1 \times P_2 \subset \mathbb{R}^2 \times \mathbb{R}^2$, a space that we simply regard as $\mathbb{R}^4$, is to construct suitable bivariate partitioning polynomials φ_1 for P_1 and φ_2 for P_2, as provided in Proposition 2.1, and then take their product $\varphi(x, y, z, w) \coloneqq \varphi_1(x, y)\varphi_2(z, w)$.

▶ **Corollary 2.2** (Polynomial partitioning of Cartesian product [23]). *The partition of $P_{1,2} \coloneqq P_1 \times P_2$ just described results in overall $O(D^4)$ relatively open cells of dimensions 2, 3, and 4, each of which contains at most $|P_{1,2}|/D^4$ points of $P_{1,2}$. The zero- and one-dimensional cells do not contain any point of $P_{1,2}$.*

The analysis in [23] also bounds the number of partition cells intersected by a two-dimensional algebraic surface S in $\mathbb{R}^4$, provided it has "good fibers." We define this notion:

▶ **Definition 2.3** (Good fibers). *(i) A two-dimensional algebraic surface S in $\mathbb{R}^4$ has* good fibers *if, for every point $p \in \mathbb{R}^2$, the fibers $(\{p\} \times \mathbb{R}^2) \cap S$ and $(\mathbb{R}^2 \times \{p\}) \cap S$ are finite. (ii) A two-dimensional algebraic surface S in $\mathbb{R}^3$ has* good fibers *if, for every point $p \in \mathbb{R}^2$, except for $O(1)$ exceptional points, the fiber $(\{p\} \times \mathbb{R}) \cap S$ is finite (it is one-dimensional for an exceptional point p), and for every point $q \in \mathbb{R}$, the fiber $(\mathbb{R}^2 \times \{q\}) \cap S$ is a one-dimensional variety (i.e., an algebraic curve).*

Note that in this definition we are only concerned with one specific decomposition of the underlying space into a product of two subspaces.

▶ **Proposition 2.4** (Cells intersected by a surface [23]). *Let S be a constant-degree two-dimensional algebraic surface in $\mathbb{R}^4$ that has good fibers. Then S intersects at most $O(D^2)$ two-, three-, and four-dimensional cells in the partitioning induced by $P_{1,2}$.*

Both Corolloary 2.2 and Proposition 2.4 have three-dimensional counterparts (for Cartesian products of a plane and a curve), presented in the full version of this paper.

3 Hierarchical Polynomial Partitioning

Even though we work in the algebraic decision-tree model, we still need to account for the cost of constructing the various polynomial partitionings (as it requires explicit access to the input points), which, if done by a straightforward application of the technique of [1], would be too expensive, as a naïve implementation of our technique needs to use polynomials of high, non-constant degree. We circumvent this issue by constructing a *hierarchical polynomial partitioning*, akin to the constructions of hierarchical cuttings of Chazelle [8] and Matoušek [19] from the 1990s. The material is rather technical, and we only state the resulting theorems, giving details in the full version.

[2] This polynomial forms a partition approximating the one in Proposition 2.1, and the constant of proportionality in the degree bound of [1] is slightly larger.

Roughly, we gain efficiency by constructing a hierarchical tree of partitions using constant-degree polynomials, until we reach subsets of the input point set of the desired size.

The actual hierarchical partitions that we will need are within a Cartesian product of either two planes or a plane and a one-dimensional curve, and are obtained by taking suitable Cartesian products of partitions constructed within each of these subspaces. We show that, up to n^ε factors, we achieve the same combinatorial properties as in a single-shot construction with a higher-degree polynomial, at a lower algorithmic cost. Specifically, we have the following results:

▶ **Theorem 3.1** (One set in the plane). *Let P be a set of n points in the plane, let $1 \le r \le n$ be an integer, and let $\varepsilon > 0$.*
(i) There is a hierarchical polynomial partition for P with $O((n/r)^{1+\varepsilon})$ bottom-level cells, each of which is associated *with at most r points of P which it contains. The hierarchy can be constructed in expected $O(n \log n)$ time.*
(ii) Any constant-degree algebraic curve γ reaches *at most $O((n/r)^{1/2+\varepsilon})$ cells at all levels of the hierarchy.*[3] *These cells can be computed within the same asymptotic time bound.*

▶ **Theorem 3.2** (Cartesian product of two planar point sets). *Let P_1, P_2 be two sets of points in the plane, each of size n, put $P_{1,2} = P_1 \times P_2 \subset \mathbb{R}^4$. Let $1 \le r \le n$ be an integer and $\varepsilon > 0$.*
(i) There is a hierarchical polynomial partition for $P_{1,2}$ with $O((n/r)^{2+\varepsilon})$ bottom-level cells, each of which is associated with a subset of at most r^2 points of $P_{1,2}$, which is the Cartesian product of a set of at most r points from P_1 and a set of at most r points from P_2, which it contains. The hierarchy can be constructed in expected $O(n \log n)$ time.
(ii) Any constant-degree two-dimensional algebraic surface S with good fibers reaches (in the same sense as in Theorem 3.1) at most $O((n/r)^{1+2\varepsilon})$ cells at all levels of the hierarchical partition of $P_{1,2}$. These cells can be computed within the same asymptotic time bound.

▶ **Theorem 3.3** (Cartesian product of a planar point set and a 1D set). *Let P be a set of n points in the plane, and let Q be a set of n points lying on a constant-degree algebraic curve $\gamma \subset \mathbb{R}^2$. Let $1 \le r, s \le \sqrt{n}$ be real parameters.*
(i) There is a hierarchical polynomial partition for $P \times Q \subset \mathbb{R}^2 \times \gamma$ into $O(n^{2+\varepsilon}/(rs)^{1+\varepsilon})$ bottom-level cells, for any $\varepsilon > 0$, each of which is associated with a subset of at most rs points of $P \times Q$, which is the Cartesian product of a set of at most r points from P and a set of at most s points from Q. The hierarchy can be constructed in expected $O(n \log n)$ time.
(ii) Any constant-degree two-dimensional surface S with good fibers reaches (in the same sense as above) at most $O\left(\frac{n^{3/2+\varepsilon}}{r^{1/2+\varepsilon}s^{1+\varepsilon}}\right)$ cells at all levels of the hierarchical partition of $P \times Q$. These cells can be computed within the same asymptotic time bound.

4 Testing for a Vanishing Pair of Polynomials

In this section we study problems of type (b), where A, B, and C are three sets of n points in the plane, and we seek a triple $(a, b, c) \in A \times B \times C$ that satisfies two polynomial equations. To simplify the presentation, we assume that they are of the form $c_1 = F(a, b)$, $c_2 = G(a, b)$, for $c = (c_1, c_2)$, where F and G are constant-degree 4-variate polynomials with *good fibers*, in the following sense: For any pair of real numbers κ_1, κ_2, the two-dimensional surface $\pi_{(\kappa_1,\kappa_2)} := \{(a, b) \in \mathbb{R}^4 \mid F(a, b) = \kappa_1,\ G(a, b) = \kappa_2\}$ has good fibers.

We fix a parameter $g \ll n$ (whose value will be set later), and apply Theorem 3.2(i) to the sets A, B, with $r = g$, to construct, in expected $O(n \log n)$ time two hierarchical planar polynomial partitionings, one for A and one for B, and combine them to obtain, a

[3] A curve γ is said to *reach* a cell τ if it intersects τ and all its ancestral cells – see the full version.

hierarchical four-dimensional polynomial partitioning for $A \times B$, so that each bottom-level cell ζ contains a Cartesian product $A_\zeta \times B_\zeta$ with $|A_\zeta|, |B_\zeta| \le g$, and the overall number of cells is $O((n/g)^{2+\varepsilon})$, for any prescribed $\varepsilon > 0$.

Let τ (resp., τ') be a bottom-level cell at the hierarchical partition of A (resp., of B). Put $A_\tau := A \cap \tau$ and $B_{\tau'} := B \cap \tau'$. The high-level idea of the algorithm is to sort lexicographically each of the sets $H_{\tau,\tau'} := \{(F(a,b), G(a,b)) \mid (a,b) \in A_\tau \times B_{\tau'}\}$, over all pairs of cells (τ, τ'). We then search with each $c = (c_1, c_2) \in C$ through the sorted lists of those sets $H_{\tau,\tau'}$ that might contain (c_1, c_2). We show that each $c \in C$ has to be searched for in only a small number of sets. As in all works on this type of problems, starting from [16], sorting the sets explicitly is too expensive. We overcome this issue by considering the problem in the algebraic decision-tree model, and by using an algebraic variant of *Fredman's trick*, extending those used in the previous algorithms for one-dimensional point sets [4, 16]. (Also, rather than carrying out the sorting in the lexicographical order, we do it in a primary round, in which we only sort the values of $F(a,b)$, followed by secondary rounds, in which we sort the values of $G(a,b)$, for each maximal block of equal values of F. For clarity of presentation, we only focus on F in the discussion below, while G is treated analogously and implicitly.)

Consider the step of sorting $\{F(a,b) \mid (a,b) \in A_\tau \times B_{\tau'}\}$. It has to perform various comparisons of pairs of values $F(a,b)$ and $F(a',b')$, for $a, a' \in A_\tau$, $b, b' \in B_{\tau'}$.

We consider $A_\tau \times A_\tau$ as a set of g^2 points in $\mathbb{R}^4$, and associate, with each pair $(b,b') \in B_{\tau'} \times B_{\tau'}$, the 3-surface $\sigma_{b,b'} = \{(a,a') \in \mathbb{R}^4 \mid F(a,b) = F(a',b')\}$. Let $\Sigma_{\tau'}$ denote the set of these surfaces. The arrangement $\mathcal{A}(\Sigma_{\tau'})$ has the property that each of its cells ζ (of any dimension) has a fixed sign pattern with respect to all these surfaces. That is, each comparison of $F(a,b)$ with $F(a',b')$, for any (a,b), $(a',b') \in A_\tau \times B_{\tau'}$, has a fixed outcome for all points $(a,a') \in \zeta$ (for a fixed pair b, b'). In other words, if we locate the points of $A_\tau \times A_\tau$ in $\mathcal{A}(\Sigma_{\tau'})$, we have available the outcome of all the comparisons needed to sort the set $\{F(a,b) \mid (a,b) \in A_\tau \times B_{\tau'}\}$.

Doing what has just been described is still too expensive (takes $\Omega(n^2)$ steps, in the algebraic decision-tree model) if implemented naïvely, processing each pair $\tau \times \tau'$ separately. We circumvent this issue, in the algebraic decision-tree model, by forming the unions $P := \bigcup_\tau A_\tau \times A_\tau$, and $\Sigma := \bigcup_{\tau'} \Sigma_{\tau'}$; we have $|P|,\ |\Sigma| = O(g^2 \cdot (n/g)^{1+\varepsilon}) = O(n^{1+\varepsilon} g^{1-\varepsilon})$. By locating each point of P in $\mathcal{A}(\Sigma)$, we get all the signs that are needed to sort all the sets $\{F(a,b) \mid (a,b) \in A_\tau \times B_{\tau'}\}$, over all pairs τ, τ' of cells, and the actual sorting costs nothing in our model, once the answers to all the relevant comparisons are known.

Searching with the points of C. We next search the structure with every $c = (c_1, c_2) \in C$. We only want to visit subproblems (τ, τ') where there might exist $a \in \tau$ and $b \in \tau'$, such that $F(a,b) = c_1$ and $G(a,b) = c_2$. To find these cells, and to bound their number, we consider the two-dimensional surface $\pi_{c=(c_1,c_2)} := \{(a,b) \in \mathbb{R}^4 \mid F(a,b) = c_1,\ G(a,b) = c_2\}$, and our goal is to enumerate the bottom-level cells $\tau \times \tau'$ in the hierarchical partition of $A \times B$ crossed by π_c. By assumption, π_c has good fibers, so, by Theorem 3.2(ii) (with $r = g$), we can find, in time $O((n/g)^{1+\varepsilon})$, the $O((n/g)^{1+\varepsilon})$ cells $\tau \times \tau'$ that π_c intersects.

Summing over all the n possible values of c, the number of crossings between the surfaces π_c and the cells $\tau \times \tau'$ is $O(n^{2+\varepsilon}/g^{1+\varepsilon})$, for any $\varepsilon > 0$. In other words, denoting by $n_{\tau,\tau'}$ the number of surfaces π_c that cross $\tau \times \tau'$, we have $\sum_{\tau,\tau'} n_{\tau,\tau'} = O(n^{2+\varepsilon}/g^{1+\varepsilon})$. Thus computing all such surface-cell crossings, over all $c \in C$, costs $O(n^{2+\varepsilon}/g^{1+\varepsilon})$ time. The cost of searching with any specific c, in the structure of a cell $\tau \times \tau'$ crossed by π_c, is $O(\log g)$ (it is simply a binary search over the sorted lists). Hence the overall cost of searching with the elements of C through the structure is (with a slightly larger ε) $O\left(\frac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right)$.

Preprocessing: Sorting the F and G values. In order to sort the F and G values, we follow a similar batched point location strategy as the one taken in [4]. That is, we perform $O(n^{1+\varepsilon}g^{1-\varepsilon})$ point location queries in an arrangement of $O(n^{1+\varepsilon}g^{1-\varepsilon})$ algebraic 3-surfaces of constant degree in $\mathbb{R}^4$. The output of this algorithm is a compact representation for the signs $F(a,b) - F(a',b')$ (where $a, a' \in A_\tau$, $b, b' \in B_{\tau'}$ over all pairs of cells τ, τ'), given as a disjoint union of complete bipartite graphs of the form $(P_\alpha \times \Sigma_\beta, \sigma)$, where $P_\alpha \subseteq P$, $\Sigma_\beta \subseteq \Sigma$, and $\sigma \in \{-1, 0, 1\}$ is the fixed sign of all points in P_α with respect to the 3-surfaces in Σ_β, where the sign of a point (a, a') with respect to a surface $\sigma_{b,b'}$ is positive (resp., zero, negative) if $F(a,b) > F(a',b')$ (resp., $F(a,b) = F(a',b')$, $F(a,b) < F(a',b')$). We show, in the following lemma, that the overall complexity of this representation, measured by the total size of the *vertex sets* of these graphs, as well as the time to construct it, is only $O\left((ng)^{8/5+\varepsilon}\right)$, where the $\varepsilon > 0$ here is slightly larger than the prescribed ε. Interestingly, as the proof of the lemma (in the full version) shows, this bound also holds in the uniform model. (G is handled by similar means.)

▶ **Lemma 4.1.** *One can perform batched point location of the points of P within the arrangement of Σ, and obtain the above complete bipartite graph representation of the output, in $O\left((ng)^{8/5+\varepsilon}\right)$ randomized expected time in the uniform model, for any prescribed $\varepsilon > 0$, where the constant of proportionality depends on ε and on the degree of F (and G).*

The overall algorithm. Combining the cost of this preprocessing stage with that of the construction of the hierarchical partitions for A and B, and of searching with the elements of C in the sorted order obtained (for free) from the complete bipartite graph representation, we get total expected running time of $O\left(n\log n + (ng)^{8/5+\varepsilon} + \dfrac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right)$. We now choose $g = n^{2/13}$, and obtain expected running time of $O\left(n^{24/13+\varepsilon}\right)$, where the implied constant of proportionality depends on the degrees of F and G and on ε, and the final ε is a (small) constant multiple of the initially prescribed ε. That is, we have shown:

▶ **Theorem 4.2.** *Let A, B, C be three n-point sets in the plane, and let F, G be a pair of constant-degree 4-variate polynomials with good fibers (in the sense defined at the beginning of this section). Then one can test, in the algebraic decision-tree model, whether there exists a triple $a \in A$, $b \in B$, $c = (c_1, c_2) \in C$, such that $c_1 = F(a,b)$ and $c_2 = G(a,b)$, using only $O\left(n^{24/13+\varepsilon}\right)$ polynomial sign tests (in expectation), for any $\varepsilon > 0$.*

▶ **Corollary 4.3.** *Let A, B, C be three sets, each of n complex numbers, and let H be a constant-degree bivariate polynomial defined over the complex numbers, with good fibers.*[4] *Then one can determine, in the algebraic decision-tree model, whether there exists a triple $(a,b,c) \in A \times B \times C$ such that $c = H(a,b)$, with only $O\left(n^{24/13+\varepsilon}\right)$ real-polynomial sign tests, in expectation, for any $\varepsilon > 0$.*

We can demonstrate this result on both instances of the triangle similarity testing problem, and show that, if A, B, C are n-point sets in the plane, so that the origin does not belong to $A \cup B \cup C$, then the following holds: One can determine, in the algebraic decision-tree model, whether there exists a triple $(a,b,c) \in A \times B \times C$ such that the triangle Δoe_1a is similar to the triangle Δobc, or that the triangle Δabc is similar to a given triangle Δuvw, with $O\left(n^{24/13+\varepsilon}\right)$ polynomial sign tests, in expectation, for any $\varepsilon > 0$. (In the full version [3] we show that the good-fiber property holds in this setting.)

[4] That is, the real and the imaginary parts of H are a pair of constant-degree 4-variate polynomials so that the intersection of their zero sets is a two-dimensional surface with good fibers.

▶ Remark 4.4. In the full version we describe a direct extension of the technique reviewed in this section to a single vanishing polynomial (which is different from the technique presented in the next section), and show that it results in a super-quadratic bound. In particular, this applies to the unconstrained collinearity testing problem in the plane.

5 Collinearity Testing and Related Problems: The Case of $2 \times 1 \times 1$ Dimensions

Let A, B, and C be three sets of points in the plane, but assume that B and C lie on respective constant-degree algebraic curves γ_B and γ_C. Our goal is to determine whether there exists a collinear triple of points in $A \times B \times C$, or more generally a triple (a, b, c) satisfying some prescribed constant-degree polynomial equation[5] $F(a, b, c) = 0$. To make the exposition easier to follow, we mostly focus on the collinearity testing problem.

Further simplifying, we assume that γ_B and γ_C are given in the parametric forms $\gamma_B(t) = (x_B(t), y_B(t))$ and $\gamma_C(s) = (x_C(s), y_C(s)))$, for $t, s \in \mathbb{R}$, where $x_B(t)$, $y_B(t)$, $x_C(s)$, $y_C(s)$ are constant-degree continuous algebraic functions, and that the sets A, B, and C are pairwise disjoint, for otherwise collinear triples exist trivially; the latter condition can be checked efficiently. A triple $(a = (a_1, a_2),\ b = (x_B(t), y_B(t)),\ c = (x_C(s), y_C(s)))$ is collinear if and only if

$$\begin{vmatrix} 1 & a_1 & a_2 \\ 1 & x_B(t) & y_B(t) \\ 1 & x_C(s) & y_C(s) \end{vmatrix} = 0, \text{ or}$$

$$x_B(t)y_C(s) - y_B(t)x_C(s) - a_1(y_C(s) - y_B(t)) + a_2(x_C(s) - x_B(t)) = 0.$$

While the theory can be developed for this general setting, we only consider here the special case where γ_C is a line, say the x-axis, so $\gamma_C(s) = (s, 0)$, and the last equality becomes:

$$-y_B(t)s + a_1 y_B(t) + a_2(s - x_B(t)) = 0 \quad \text{or} \quad s = \varphi(a, t) := \frac{a_1 y_B(t) - a_2 x_B(t)}{y_B(t) - a_2}.$$

Here φ is a constant-degree algebraic function; it is a linear rational function in a.

We fix a pair of parameters $g, h \ll n$ (whose values will be set later) and a parameter $\varepsilon > 0$, and apply Theorem 3.3(i) to the sets A, B, with the respective parameters $r = g, s = h$. Let τ (resp., τ') be a bottom-level cell in the resulting partition for A (resp., B). Put $A_\tau := A \cap \tau$ and $B_{\tau'} := B \cap \tau'$. In this analysis, somewhat abusing the notation, we regard B as a subset of $\mathbb{R}$, and denote by t the real parameter that parameterizes γ_B; in particular, we write $t \in B$ (resp., $t \in B_{\tau'}$) instead of $\gamma_B(t) \in B$ (resp., $\gamma_B(t) \in B_{\tau'}$). The number of bottom-level cells τ (and sets A_τ) is $O\left(\left(\frac{n}{g}\right)^{1+\varepsilon}\right)$, for any $\varepsilon > 0$, and the number of bottom-level cells τ' (and sets $B_{\tau'}$) is n/h.

The high-level idea of the algorithm is to sort each of the sets $\{\varphi(a, t) \mid (a, t) \in A_\tau \times B_{\tau'}\}$, over all pairs (τ, τ') of cells, and then to search with each $c = (s, 0) \in C$ (i.e., with the corresponding real s) through the sorted lists of only those sets that might contain s; this number is small, as argued below.

Again, sorting the sets explicitly is too expensive, and we use an instance of the algebraic variant of Fredman's trick, as in the previous section.

[5] Note that here F is (naturally) implicit, as opposed to the preceding section, where we used two explicit expressions $c_1 = F(a, b)$, $c_2 = G(a, b)$.

Preprocessing for batched point location. Consider the step of sorting $\{\varphi(a,t) \mid (a,t) \in A_\tau \times B_{\tau'}\}$, which has to perform various comparisons of pairs of values $\varphi(a,t)$ and $\varphi(a',t')$, for $a, a' \in A_\tau$, $t, t' \in B_{\tau'}$. We perform this task globally over all pairs (τ, τ') of cells.

We recurse by switching between a "primal" and a "dual" setups. In the primal, we view $P = \bigcup_\tau A_\tau \times A_\tau$ as a set of $O\left(\left(\frac{n}{g}\right)^{1+\varepsilon} \cdot g^2\right) = O(n^{1+\varepsilon} g^{1-\varepsilon})$ points in $\mathbb{R}^4$, and associate with each pair $(t,t') \in B_{\tau'} \times B_{\tau'}$, for each cell τ', the three-dimensional constant-degree algebraic surface $\sigma_{t,t'} = \{(a,a') \in \mathbb{R}^4 \mid \varphi(a,t) = \varphi(a',t')\}$. We let Σ be the collection of all these surfaces, over all cells τ', and have $|\Sigma| = n/h \cdot h^2 = O(nh)$.

In the dual, we view the pairs $(t,t') \in \bigcup_{\tau'} B_{\tau'} \times B_{\tau'}$ as points in the plane, and associate with each pair $(a,a') \in P$ the curve $\delta_{a,a'} = \{(t,t') \in \mathbb{R}^2 \mid \varphi(a,t) = \varphi(a',t')\}$. In each primal problem we need to perform batched point-location queries in an arrangement of (some subset of the) constant-degree algebraic 3-surfaces $\sigma_{t,t'}$ in $\mathbb{R}^4$, and in each dual problem we need to perform batched point location queries in an arrangement of (some subset of the) constant-degree algebraic curves $\delta_{a,a'}$ in $\mathbb{R}^2$. Initially we are in the primal, with $O(n^{1+\varepsilon} g^{1-\varepsilon})$ points and $O(nh)$ 3-surfaces.

If we could construct the full arrangement $\mathcal{A}$ of these surfaces and locate in it all these points, we would get the signs of all the differences $\varphi(a,t) - \varphi(a',t')$, for all (a,t), $(a',t') \in A_\tau \times B_{\tau'}$, over all pairs (τ,τ') of cells, from which we would get (for free) the sorted order of the sets $\{\varphi(a,t) \mid (a,t) \in A_\tau \times B_{\tau'}\}$, over all pairs (τ,τ'). However, a single-step construction of $\mathcal{A}$ is too expensive, so we replace it with the above "flip-flop" primal-dual processing, each time partitioning the (current version of the) arrangement using a polynomial of small degree, and thereby reduce the cost to that stated below.

The output of this preprocessing is a representation of $P \times \Sigma$ as a disjoint union of complete bipartite graphs, described in the following lemma (see the full version [3]):

▶ **Lemma 5.1.** *The above recursive batched point-location stage takes randomized expected time $O\left(n^{10/7+\varepsilon'} g^{6/7+\varepsilon'} h^{4/7}\right)$, also in the uniform model, where ε' is larger, by a small constant factor, than the prescribed ε.*

Searching with the points of C. We next search the structure with every $s \in C$ (identified with the point $(s,0)$ on the x-axis). For each $s \in C$, we only want to visit subproblems (τ,τ') where there might exist $a \in \tau$ and $t \in \tau'$ (not necessarily from $A_\tau \times B_{\tau'}$), such that $\varphi(a,t) = s$. We consider the two-dimensional surface $\pi_s \coloneqq \{(a,t) \in \mathbb{R}^3 \mid \varphi(a,t) = s\}$ and show that it has good fibers (details appear in the full version [3]).

We next proceed as follows. By Theorem 3.3(ii), choosing g and h to satisfy $\left(\frac{n}{g}\right)^{1/2} = \frac{n}{h}$, or $h = n^{1/2} g^{1/2}$, we ensure that π_s reaches $O\left(\frac{n^{1+\varepsilon}}{g^{1+\varepsilon}}\right)$ cells $\tau \times \tau'$. Summing over all the n possible values of s, the number of crossings between the surfaces π_s and the cells $\tau \times \tau'$ is $O\left(\frac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right)$. Denoting by $n_{\tau,\tau'}$ the number of surfaces π_s that cross $\tau \times \tau'$, we have $\sum_{\tau,\tau'} n_{\tau,\tau'} = O\left(\frac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right)$ and we can enumerate all such crossings in $O(n^{2+\varepsilon}/g^{1+\varepsilon})$ time.

The cost of searching with any specific s in the structure of a cell $\tau \times \tau'$ crossed by π_s, is $O(\log g)$. Hence the overall cost of searching with the elements of C through the structure is $O(n^{2+\varepsilon}/g^{1+\varepsilon})$, where ε is slightly larger than the originally prescribed one.

Combining this cost with that of the construction of the hierarchical polynomial partitioning, and the point-location preprocessing stage, we get overall expected time of

$$O\left(n\log n + n^{10/7+\varepsilon}g^{6/7+\varepsilon}h^{4/7} + \frac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right) = O\left(n\log n + n^{12/7+\varepsilon}g^{8/7+\varepsilon} + \frac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right).$$

We roughly balance the two last terms by choosing $g = n^{2/15}$, making the overall cost of the procedure $O\left(\frac{n^{2+\varepsilon}}{g^{1+\varepsilon}}\right) = O\left(n^{28/15+\varepsilon}\right)$.

A similar analysis, albeit somewhat more complicated, can handle the case where C is contained in a general constant-degree algebraic curve, rather than a line.[6] In summary, we thus obtain (see the full version for more details [3]):

▶ **Theorem 5.2.** *Let A, B, C be n-point sets in the plane, where B and C are each contained in some respective constant-degree algebraic curve γ_B, γ_C, and assume $B \cap \gamma_C = \emptyset$. Then one can test whether $A \times B \times C$ contains a collinear triple, in the algebraic decision-tree model, using only $O\left(n^{28/15+\varepsilon}\right)$ polynomial sign tests (in expectation), for any $\varepsilon > 0$, where the constant of proportionality depends on ε and on the degrees of γ_B, γ_C.*

Unit area triangles and other problems. This analysis can be extended to the *unit area triangle* problem, where we want to test for the existence of a triangle spanned by $A \times B \times C$ that has unit area, as well as to many other similar problems (e.g., does there exist a triangle spanned by $A \times B \times C$ that has circumradius 1, or inradius 1, or unit perimeter, and so on). All these variants can be solved with the same technique and within the same asymptotic performance bound as in Theorem 5.2, provided that the good-fiber property is satisfied.

6 Higher Dimensions

Collinearity in higher dimensions: The $d \times (d-1) \times (d-1)$ case. Let A, B and C be three sets of n points each, so that A is a set of points in $\mathbb{R}^d$ and each of B and C lies in a *hyperplane*. The goal is to test, in the algebraic decision tree model, whether $A \times B \times C$ contains a collinear triple. Our approach is to use a recursive chain of projections, which ultimately map the points in A, B, and C to some plane, so that B and C is each mapped to a set of points on some respective line, collinearity is preserved, and no new collinearity appears among the projected points. This is a variant of a projection technique described by De Zeeuw [9]. Deferring all the details to the full version of the paper, we obtain:

▶ **Theorem 6.1.** *Let A, B and C be three sets of n points each, where A is a set of points in $\mathbb{R}^d$ and B, C each lies in a respective hyperplane h_1, h_2. Assume that $h_1 \neq h_2$, $A \subset \mathbb{R}^d \setminus (h_1 \cup h_2)$ and $(B \cup C) \cap h_1 \cap h_2 = \emptyset$. Then one can test whether $A \times B \times C$ contains a collinear triple, in the algebraic decision tree model, by a randomized algorithm that succeeds with probability 1, and uses only $O\left(n^{28/15+\varepsilon}\right)$ polynomial sign tests (in expectation), for any $\varepsilon > 0$, where the constant of proportionality depends on ε and on d.*

We present in the full version [3] several initial results for more general extensions of both the single-polynomial and the polynomial-pair vanishing problems to higher dimensions. In the former setup, each of B and C is contained in an algebraic surface of codimension 1 and constant degree. Unlike the bound in Theorem 6.1, the bounds that we obtain deteriorate with d, but remain subquadratic for every d.

[6] For this extension one can apply *quantifier elimination* (see, e.g., [5]) and use a similar batched point-location mechanism, albeit with more complicated semi-algebraic sets (but still of constant complexity and same dimensionality).

References

1 Pankaj K. Agarwal, Jiří Matoušek, and Micha Sharir. On range searching with semialgebraic sets. II. *SIAM J. Comput.*, 42(6):2039–2062, 2013. doi:10.1137/120890855.

2 Boris Aronov and Jean Cardinal. Geometric pattern matching reduces to k-SUM, 2020. arXiv:abs/2003.11890.

3 Boris Aronov, Esther Ezra, and Micha Sharir. Testing polynomials for vanishing on Cartesian products of planar point sets, 2020. Full version of this paper. arXiv:2003.09533.

4 Luis Barba, Jean Cardinal, John Iacono, Stefan Langerman, Aurélien Ooms, and Noam Solomon. Subquadratic algorithms for algebraic 3SUM. *Discrete Comput. Geom.*, 61(4):698–734, 2019. doi:10.1007/s00454-018-0040-y.

5 Saugata Basu, Richard Pollack, and Marie-Françoise Roy. *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, second edition, 2006.

6 Michael Ben-Or. Lower bounds for algebraic computation trees (preliminary report). In *Proc. of the 15th Annual ACM Symposium on Theory of Computing*, pages 80–86, 1983. doi:10.1145/800061.808735.

7 Timothy M. Chan. More logarithmic-factor speedups for 3SUM, (median, +)-convolution, and some geometric 3SUM-hard problems. *ACM Trans. Algorithms*, 16(1):7:1–7:23, 2020. doi:10.1145/3363541.

8 Bernard Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete Comput. Geom.*, 9:145–158, 1993. doi:10.1007/BF02189314.

9 Frank de Zeeuw. Ordinary lines in space, 2018. arXiv:1803.09524.

10 György Elekes and Lajos Rónyai. A combinatorial problem on polynomials and rational functions. *J. Comb. Theory, Ser. A*, 89(1):1–20, 2000. doi:10.1006/jcta.1999.2976.

11 György Elekes and Endre Szabó. How to find groups? (and how to use them in Erdős geometry?). *Combinatorica*, 32(5):537–571, 2012. doi:10.1007/s00493-012-2505-6.

12 Jeff Erickson and Raimund Seidel. Better lower bounds on detecting affine and spherical degeneracies. *Discrete Comput. Geom.*, 13:41–57, 1995. doi:10.1007/BF02574027.

13 Ari Freund. Improved subquadratic 3SUM. *Algorithmica*, 77(2):440–458, 2017. doi:10.1007/s00453-015-0079-6.

14 Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom.*, 5:165–185, 1995. doi:10.1016/0925-7721(95)00022-2.

15 Omer Gold and Micha Sharir. Improved bounds for 3SUM, k-SUM, and linear degeneracy. In *25th Annual European Symposium on Algorithms*, pages 42:1–42:13, 2017. Also in arXiv:1512.05279. doi:10.4230/LIPIcs.ESA.2017.42.

16 Allan Grønlund and Seth Pettie. Threesomes, degenerates, and love triangles. *J. ACM*, 65(4):22:1–22:25, 2018. doi:10.1145/3185378.

17 Larry Guth and Nets Hawk Katz. On the Erdős distinct distances problem in the plane. *Ann. Math. (2)*, 181(1):155–190, 2015. doi:10.4007/annals.2015.181.1.2.

18 Daniel M. Kane, Shachar Lovett, and Shay Moran. Near-optimal linear decision trees for k-SUM and related problems. *J. ACM*, 66(3):16:1–16:18, 2019. doi:10.1145/3285953.

19 Jiří Matoušek. Range searching with efficient hierarchical cuttings. *Discrete Comput. Geom.*, 10(2):157–182, 1993. doi:10.1007/BF02573972.

20 Franco P. Preparata and Michael Ian Shamos. *Computational Geometry—An Introduction*. Texts and Monographs in Computer Science. Springer, 1985. doi:10.1007/978-1-4612-1098-6.

21 Orit E. Raz, Micha Sharir, and Frank de Zeeuw. Polynomials vanishing on Cartesian products: the Elekes-Szabó theorem revisited. *Duke Math. J.*, 165(18):3517–3566, 2016 . doi:10.1215/00127094-3674103.

22 Orit E. Raz, Micha Sharir, and József Solymosi. Polynomials vanishing on grids: the Elekes-Rónyai problem revisited. *Amer. J. Math.*, 138(4):1029–1065, 2016. doi:10.1353/ajm.2016.0033.

23 József Solymosi and Frank de Zeeuw. Incidence bounds for complex algebraic curves on Cartesian products. In *New Trends in Intuitive Geometry*, volume 27 of *Bolyai Soc. Math. Stud.*, pages 385–405. János Bolyai Math. Soc., Budapest, 2018.