Annales Mathematicae et Informaticae
44 (2015) pp. 35–46
http://ami.ektf.hu

Solving computational problems in real algebra/geometry

James H. Davenport*

University of Bath (U.K.)
J.H.Davenport@bath.ac.uk

Submitted September 16, 2014 — Accepted January 26, 2015

Abstract

We summarise some computational advances in the theory of real algebra/geometry over the last 15 years, and list some areas for future work.

Keywords: Real algebraic geometry, cylindrical algebraic decomposition

1. Introduction

Real Algebra and Geometry have many computational applications. In theory they solve many problems of robot motion planning [34], though practice is not so kind [38]. Understanding the (real) geometry of branch cuts [16] is crucial to issues of complex function simplification [3]. A very powerful technique in computation real geometry is *Cylindrical Algebraic Decomposition*, introduced in [12] to solve problems of Quantifier Elimination.

Notation 1.1. We assume that the initial problem posed has m polynomials, of degree (in each variable separately) at most d, in n variables.

*Thanks to Russell Bradford, Matthew England, Nicolai Vorobjov, David Wilson (Bath); Chris Brown (USNA); Scott McCallum (Macquarie); Marc Moreno Maza (UWO) and Changbo Chen (CIGIT), and to the referees. This talk grew out of an invitation to speak at ICAI 2014 in Eger, and the author is grateful to the organisers. The underlying work was supported by EPSRC grant EP/J003247/1.

2. Quantifier elimination

A key technique we are going to use is *Quantifier Elimination*: throughout, $Q_i \in \{\exists, \forall\}$. Given a statement

$$\Phi := Q_{k+1}x_{k+1} \dots Q_n x_n \phi(x_1, \dots, x_n),$$

where ϕ is in some (quantifier-free, generally Boolean-valued) language $\mathcal{L}$, the Quantifier Elimination problem is that of producing an equivalent

$$\Psi := \psi(x_1, \dots, x_k) : \qquad \psi \in \mathcal{L}.$$

In particular, $k = 0$ is a decision problem: is Φ true?

The Quantifier Elimination problem is critically dependent on the language $\mathcal{L}$ and the range of the variables x_i. For example

$$\forall n : n > 1 \Rightarrow \exists p_1 \exists p_2 \left(p_1 \in \mathcal{P} \wedge p_2 \in \mathcal{P} \wedge 2n = p_1 + p_2\right)$$
$$[\text{where } m \in \mathcal{P} \equiv m > 1 \wedge \forall p \forall q \left(m = pq \Rightarrow p = 1 \vee q = 1\right)]$$

is a statement of Goldbach's conjecture in the language of the natural numbers with, naïvely, seven quantifiers (five will do if we use the same quantifiers for the two instances of $\mathcal{P}$).

In fact, quantifier elimination is impossible over the natural numbers [29]. From this it follows that it is impossible over the real numbers if we allow unrestricted[1] trigonometric transcendental functions in $\mathcal{L}$, since $n \in \mathbf{Z}$ is equivalent to $\sin(n\pi) = 0$. The function sin satisfies a second-order (or coupled pair of first-order) differential equation(s), and there are positive results provided we restrict ourselves to Pfaffian functions, i.e. solutions of triangular systems of first-order partial differential equations with polynomial coefficients. Exploring this is beyond the scope of this paper: see [24].

However, quantifier elimination is possible for semi-algebraic (polynomials and inequalities) $\mathcal{L}$ over $\mathbf{R}$ [35]. Note that we need to allow inequalities: the quantifier-free form of $\exists y : y^2 = x$ is $x \geq 0$, and $\exists y : xy = 1$ has the quantifier-free form $x \neq 0$. Formally we define the language of *real closed fields*, $\mathcal{L}_{\mathrm{RCF}}$, to include the natural numbers, $+, -, \times, =, >$ and the Boolean operators. Then $\exists y : y^2 = x$ eliminates the quantifier to $(x > 0) \vee (x = 0)$ and $\exists y : xy = 1$ eliminates the quantifier to $(x > 0) \vee (0 > x)$.

It is worth noting that in practice we nearly always treat $\neq$ as a first-class citizen, and indeed this is necessary when proceeding via regular chains (Section 3.3).

3. Cylindrical algebraic decomposition

Unfortunately, the complexity of Tarski's method is indescribable (in the sense that no tower of exponentials can describe it) and we had to wait for [12] for a remotely

[1] Note that the undecidability comes from the fact that the function $\sin : \mathbf{R} \to \mathbf{R}$ has infinitely many zeros. Restricted versions are a different matter: see [24, (h) p. 214].

plausible method.

3.1. Collins' method

Collins proceeds via a *cylindrical algebraic decomposition*, which is (almost) what it says: a decomposition of $\mathbf{R}^n$ into cells $C_{\mathbf{i}}$ indexed by n-tuples of natural numbers (so $\mathbf{R}^n = \bigcup_{\mathbf{i}} C_{\mathbf{i}}$ and $\mathbf{i} \neq \mathbf{j} \Rightarrow C_{\mathbf{i}} \cap C_{\mathbf{j}} = \emptyset$), which is (semi-)algebraic in the sense that every $C_{\mathbf{i}}$ is defined by a finite set of equalities and inequalities of polynomials in the x_i and which is *cylindrical*, meaning that, for all $k < n$, if π_k is the projection operator onto the *first* k coordinates, then, for all $\mathbf{i}, \mathbf{j}$, $\pi_k(C_{\mathbf{i}})$ and $\pi_k(C_{\mathbf{j}})$ are either equal or disjoint. Collins constructs a cylindrical algebraic decomposition which is *sign-invariant* for the polynomials in ϕ, i.e. on each cell, every polynomial is identically zero, or everywhere positive, or everywhere negative.

The construction and use of such a decomposition is roughly[2] described below.

1 Let $\mathcal{S}_n$ be the polynomials in ϕ (m polynomials, degree $\leq d$, n variables).

2 Compute $\mathcal{S}_{n-1}$ ($\Theta(d^3m^2)$ polynomials, degree $\Theta(2d^2)$, $n-1$ variables), such that, over a cylindrical algebraic decomposition of $\mathbf{R}^{n-1}$ sign-invariant for the polynomials in $\mathcal{S}_{n-1}$, the polynomials in $\mathcal{S}_n$ are collectively delineable, meaning each branch of each of them is defined by a continuous algebraic function of $x_1, \ldots, x_{n-1}$, and the branches of all polynomials are either equal or disjoint;

3 and $\mathcal{S}_{n-2}$ ($\Theta((2d^2)^3(d^3m^2)^2)$ polynomials, degree $\Theta(2(2d^2)^2)$, $n-2$ variables) satisfying a similar condition;

$\vdots$ continue

n and $\mathcal{S}_1$ ($\leq (2d)^{3^n} m^{2^{n-1}}$ polynomials, degree $\leq \frac{1}{2}(2d)^{2^{n-1}}$, 1 variable) satisfying a similar condition.

$n+1$ Isolate the N_1 roots of S_1, decomposing $\mathbf{R}^1$ into N_1 zero-dimensional points and $N_1 + 1$ one-dimensional regions. Pick a sample point in each one-dimensional region.

$n+2$ Over each root, or at the sample points for each interval between roots, isolate roots of S_2, and pick a sample point between each adjacent pair of roots.

$\vdots$ continue

$2n$ Over each cell in the decomposition of $\mathbf{R}^{n-1}$, isolate roots of $\mathcal{S}_n$, pick a sample point between each adjacent pair of roots, and hence make our decomposition of $\mathbf{R}^n$.

[2]For example, we ignore the growth in coefficient sizes. This can be (and is in [12]) tracked in detail, but doesn't affect the general argument.

So S_n has invariant signs on each region of $\mathbf{R}^n$, and $\phi(x_1, \ldots, x_n)$ has invariant truth on each region. Therefore all questions about ϕ reduce to the values of ϕ at the sample points.

$2n+1$ Evaluate the truth of Φ on each region R of $(x_1, \ldots, x_k)$-space, by looking at the values of ϕ at the sample points lying above the sample point of R, and combining them according to the quantifiers in Φ.

$2n+2$ The quantifier-free form Ψ of Φ is then the disjunction of the definitions of those regions of $(x_1, \ldots, x_k)$-space for which Φ is true.

The time complexity ends up being bounded [12, Theorem 16] by

$$O\left(m^{2^{n+6}}(2d)^{2^{2n+8}}\right).$$

While running time is one measure of complexity, it depends on the various sub-algorithms, and in practice depends on a lot of implementation details. A more refined analysis [15] of the complexity of step $n+1$ (and its knock-on effects on the subsequent steps), for example, reduces the complexity bound[3] (not the actual time) to $O\left(m^{2^{n+\not{6}4}}(2d)^{2^{2n+\not{8}6}}\right)$. Hence practitioners in the field of cylindrical algebraic decomposition tend to concentrate on the number of cells in the final decomposition. This has several advantages [6].

- It can be directly compared across systems, irrespective of hardware or software details.
- Most applications do significant amounts of post-processing on the cells, so the complexity of the post-processing is dependent on the number of cells (and on the complexity of the descriptions of the cells and their sample points, which a simple count doesn't capture directly: however experience shows that if algorithm A generates more cells than algorithm B on a given problem, algorithm A's descriptions are at least as complex).
- For a given problem and software/hardware system, the number of cells and the processing time tend to be closely correlated: a point first made explicitly in [18].
- The known *lower* bounds on complexity [17, 4] are in fact lower bounds on the number of cells.

The number of cells produced by Collins' method is bounded, by an analysis similar to [6], by

$$O\left(m^{2^n}(2d)^{2\cdot 3^n}\right).$$

[3] This may seem like a trivial improvement. In fact, the new bound is the fourth root of the old one, an improvement that would be viewed as massive in other contexts.

3.2. McCallum's improvements

Definition 3.1. The order of f at a point x is the least k such that at least one partial derivative of f of order k does not vanish at x.

McCallum [30] introduced a new projection operator for producing $\mathcal{S}_{i-1}$ from $\mathcal{S}_i$. In fact, he constructs a stronger cylindrical algebraic decomposition, *order-invariant* for the polynomials in ϕ, i.e. on each cell, every polynomial is identically zero of constant order, or everywhere positive, or everywhere negative. Clearly every order-invariant decomposition is sign-invariant, but the converse is not true. This approach has three major features.

pro Despite the fact that we are constructing a richer final object, the new projection sets are much smaller, and our analysis [6], based on the key result [30, Lemma 6.1.1] shows that the number of cells is bounded by

$$2^{n2^{n-1}} m^{2^n-1} d^{n2^{n-1}},$$

where the key improvement[4] is in the exponent of d, being of the form $n2^n$ rather than 3^n.

con The projection might not always work. There is a technical condition, known as *well-oriented* in [30], which is only discovered in phases $n+2, \ldots, 2n-1$, when a polynomial in $\mathcal{S}_k$ turns out to vanish identically on a cell of nonzero dimension. In these circumstances we can either revert to using an improvement [26] of the full Collins method (with its attendant costs), or, as suggested in [30] but to the best of the author's knowledge never implemented, whenever, in the *projection* phases, an $\mathcal{S}_i$ contains a polynomial that *might* nullify, add its partial derivatives with respect to each variable to the set $\mathcal{S}_i$. Again to the best of the author's knowledge, the complexity of this has never been analysed.

In theory, well-orientedness ought to occur "with probability 1". However, humans don't pose random problems, and the experience of the author and his Bath colleagues is that well-orientedness can frequently fail to occur, especially when solving problems coming from simplification, as in [3].

odd Step $2n+2$ may run into difficulties, a problem first pointed out in [8]. The roots of $\mathcal{S}_1$ isolated in step $n+1$ are of the form "the (unique) root α of $p(x)$ lying in (β, γ)", where $\beta, \gamma \in \mathbf{Q}$ (in practice $\in \mathbf{Z}[1/2]$). This statement is in our language $\mathcal{L}_{\mathrm{RCF}}$ ($p(x) = 0 \wedge x > \beta \wedge \gamma > x$). However, the branches of $\mathcal{S}_2$ (and other S_i) are in the form "that branch of $p(x_1, x_2)$ such that $p(\alpha, x_2)$ lies in (β, γ)", where $\beta, \gamma \in \mathbf{Q}$, and this is not in $\mathcal{L}_{\mathrm{RCF}}$. We could equally

[4] An improved analysis of [30, Lemma 6.1.1] in fact gives

$$2^{2^{n+1}-2n} (md)^{2^n-1}, \tag{3.1}$$

reducing the exponent of d further.

describe it as "the third real branch of $p(x_1, x_2)$ when $x_1 \in (\alpha_1, \alpha_2)$", but again this statement is not in $\mathcal{L}_{\mathrm{RCF}}$. Now by Thom's Lemma [14], we can describe this branch in terms of the signs of p *and its derivatives*, but, whereas these derivatives are in the Collins projection, they are not in the McCallum projection. However, when it comes to step $2n+2$, we can just add these, so the additional cost is negligible.

3.3. Regular chains methods

The production of *Cylindrical Algebraic Decompositions by Regular Chains* was first introduced in [11], and an improved version (essentially of the first step) was presented in [9]. Unlike the previous methods, they go via complex space, essentially as:

1. construct a cylindrical decomposition of $\mathbf{C}^n$ which is *zero/nonzero-invariant* for the polynomials in ϕ;

2. refine this to a cylindrical algebraic decomposition of $\mathbf{R}^n$, which will therefore be sign-invariant for the polynomials in ϕ;

3. for the same reasons as those described under **odd** in the previous section, if necessary add extra derivatives to be able to express the quantifier-free result in $\mathcal{L}_{\mathrm{RCF}}$ [10].

Not much is known about the theoretical complexity of regular chain computation in general, but this method does seem to be[5] at least competitive with, and often better than, our implementation [22] of [30] using the same Maple technology and libraries[6].

4. Equational constraints

It is often the case that ϕ contains equations: can we make use of these? [31] (based on [13]) was the first to show how. He defined an *equational constraint* $h = 0$ as an equality logically implied by ϕ, i.e. $\phi(x_1 \ldots, x_n) \Rightarrow h(x_1, \ldots, x_n) = 0$. For the sake of simplicity, we assume that ϕ actually has the form $(h = 0) \wedge \hat{\phi}$, and that $\hat{\phi}$ involves $m-1$ polynomials g_i (therefore m in all). We will not ask for an order-invariant decomposition, but rather an *equationally sign-invariant* decomposition: one where the signs of f and the g_i are constant on every cell **where** $f = 0$ — it is quite possible that cells where $f \neq 0$ have a mixture of behaviours of the g_i. We also need h to have main variable x_n. Compared with equation (3.1), this generates [6] at most

$$2^{2^{n+1}-2n}(m+1)^{2^{n-1}-1}d^{2^n-1}$$

[5] Compare the columns RC-Inc-CAD and PL-CAD in [19, Table 1].

[6] `www.regularchains.org`.

cells: replacing m^2 by $m+1$ in a single projection, and hence reducing the double exponent in the overall complexity.

This process is generalised in [5] to consider ϕ of the form

$$\underbrace{(f_1 = 0 \wedge g_{1,1} > 0 \wedge \cdots)}_{\phi_1} \vee \underbrace{(f_2 = 0 \wedge \cdots)}_{\phi_2} \vee \cdots \vee \underbrace{(f_k = 0 \wedge g_{k,1} > 0 \wedge \cdots)}_{\phi_k}. \quad (4.1)$$

This *does* have an equational constraint, viz. $\prod_i f_i = 0$, but this is of degree kd if the f_i have degree d. What [5] produces is rather a *Truth Table Invariant* decomposition (TTICAD), in which the truth of each ϕ_i is invariant on each cell. Practically this shows further savings, and the complexity is analysed in [6]. It uses a strict subset of the projection sets $\mathcal{S}_i$ generated with the equation constraint $\prod_i f_i = 0$ (for example not having $\text{res}_{x_n}(f_2, g_{1,1})$), so is clearly an improvement.

This method is further generalised in [6] to the case where not every ϕ_i in (4.1) has an equality. The savings are less spectacular than when every ϕ_i has an equality, but much more spectacular than the McCallum projection. The complexity [6] depends on the shape of ϕ.

These methods have also been applied to the Regular Chains approach to constructing cylindrical algebraic decompositions [2], again with good experimental results, but no complexity analysis. However, a curious feature comes up here. The method of [9] is incremental: one starts with the trivial decomposition, invariant for $\emptyset$, and adds polynomials. [2] build on this, but prune the cylindrical decomposition being created according to the Boolean structure of ϕ. This means that the same ϕ, but processed in a different order, can produce different cylindrical algebraic decompositions, and different (though of necessity logically equivalent) quantifier-free forms. This is analysed in [19].

5. So I have a problem in $\mathcal{L}_{\text{RCF}}$: what do I do?

A first remark is that this is a common phenomenon: according to [1], 47% of the problems in the Tokyo University mathematics entrance examination *can be posed* in $\mathcal{L}_{\text{RCF}}$. Most of these, however, *are actually posed* in terms of elementary geometry, and translating these into forms involving the fewest number of variables (note how all the complexity formulae are doubly-exponential in n, as are the lower bounds [17, 4]) is non-trivial: an obvious translation might have twelve variables, whereas we "really only need" three.

There are also many choices of the precise way the problem is formulated: see [7] in general for projection-based methods, [19] for Regular Chains methods, and [38] for a specific example in robot motion planning.

Even after doing this, in fact, you probably have a problem that can be expressed in $\mathcal{L}_{\text{RCF}}$ in many ways. If our goal is quantifier elimination, then step $2n+1$ implies that the variables must be projected (in terms of the cylindricity property, even if we are using Regular Chains methods) in the order implied by the quantifiers. But this still leaves much unspecified, e.g. $x_1, \ldots, x_k$ can be projected

in any order, and we can apply $\forall x \forall y \equiv \forall y \forall x$ (and its $\exists$ equivalent). Other applications, such as robot motion planning [34] and simplification [3] impose no constraints on the variable order. Hence an $\mathcal{L}_{\mathrm{RCF}}$-problem with n variables may have up to $n!$ possible instantiations as a cylindrical algebraic decomposition problem. This problem was first studied systematically in [18], who produced and evaluated various heuristics to pick the best order. Having observed that no one heuristic is best for all problems, [25] used machine learning on a large set of problems (using McCallum-style methods) to see if machine learning could predict which heuristic to use in a given setting, and found that this did indeed do better than any fixed choice of heuristic. [21] explored the choice of variable ordering for Regular Chains methods. The theoretical importance of variable ordering is shown by [4, Theorem 7] who produce a family of systems that has doubly-exponentially many cells in one ordering, and a constant number (independent of n) in another. Conversely [4, Theorem 8] there are problems that are doubly exponential for all orders.

If a problem involves equalities, then, as well as the simplifications in Section 4, we can also use the equalities to perform algebraic simplification of each other, and of the appropriate inequalities. This was first noticed in the case of branch cuts by Phisanbut [33, 32], where a simple cut like $\Re(z) < 0 \wedge \Im(z) = 0$ becomes $f_{\Re}(z) < 0 \wedge f_{\Im}(z) = 0$, but $f_{\Re}(z)$ is only relevant when $f_{\Im}(z) = 0$. This was generalised in [36], who observed that it is *often* helpful, but not always. They evaluated several heuristics, including those from [18], to determine which formulation of a given problem to solve.

In the case of Projection/Lifting methods, whether Collins or McCallum, it is nearly always the case that the lifting step $(n+1, \ldots, 2n)$ are by far the most expensive. Within these, it is the lifting of cells of non-full dimension that is the most expensive component, since in general this will involve algebraic number manipulations. Hence [37] suggest using a variant of these steps that *only* lifts the full-dimensional cells, picking the ordering (or other formulation choices) that minimises this, and then recovering the lower-dimensional cells. Further improved lifting techniques in the presence of equational constraints are described in [20].

6. Conclusion

6.1. Overview

In the fifteen years since [31], there has been a great deal of work on algorithms for cylindrical algebraic decomposition, both for quantifier elimination and other problems. Notably the Regular Chains method [11, 9] has appeared as a competitor to the traditional Projection/Lifting approach of [12] and his school. Nevertheless, there are at least as many open questions as there were before, and some are given below.

6.2. Open questions

1. Produce some complexity results for the Regular Chains method (Section 3.3). It follows from the work of that section, and the fact that quantifier elimination is doubly exponential [17, 4] that Regular Chain computation (of the kind used in these computations) must be, but this has not been explored systematically.

2. Formalise the "we only need" issue of writing $\mathcal{L}_{\mathrm{RCF}}$ formulae expressing problems with as few variables as possible. A lot of this seems to be a variation on "without loss of generality", for example a naïve algebraicisation of

   > Given a triangle ABC

   might be

   > Given a triangle ABC where A is at (a_x, a_y), B is at (b_x, b_y) and C is at (c_x, c_y)

   but a mathematician using coordinates is more likely to write

   > Given a triangle ABC where, without loss of generality, A is at $(0, 0)$, B is at $(0, 1)$ and C is at (c_x, c_y)

   thus reducing the number of free variables by four, and more if we then need to choose a point on AB.

3. Extend the machine learning approach of [25] to a wider class of problems, and also a wider set of choices, not just variable ordering but also preconditioning [36] and formulation [19].

4. The method of [37], which seems to be the most accurate predictor of the "best formulation", is only applicable to Projection/Lifting methods. Can the fundamental idea be applied to Regular Chains methods as well?

5. The lower bounds for quantifier elimination are based on a construction which alternates quantifiers, e.g. [4] uses $\underbrace{\exists\forall\forall}_{\text{block}} \ldots \underbrace{\exists\forall\forall}_{\text{block}}$. There are theoretical methods [23] which are doubly-exponential only in the number a of alternations of quantifiers: $(md)^{n^{O(a)}}$. To the best of the author's knowledge, these have never been implemented. Note, however, that since all the Projection/Lifting algorithms we have shown are doubly-exponential in n (just in the Projection phases), this means that cylindrical algebraic decomposition is theoretically not the best tool.

6. This point is borne out by [27, 28], who solve the *purely existential* version of quantifier elimination, more precisely

$$\exists x_1 \exists x_2 \ldots \exists x_n \phi(x_1, \ldots, x_n), \tag{6.1}$$

by a method which can be thought of as a blend of the algebra underpinning cylindrical algebraic decomposition with the methodology of modern SAT-solvers. It currently seems to be impossible to generalise beyond (6.1), even to a single alternation, but again this is a topic crying out for progress.

References

[1] N.H. Arai, T. Matsuzaki, H. Iwane, and H. Anai. Mathematics by Machine. In K. Nabeshima, editor, *Proceedings ISSAC 2014*, pages 1–8, 2014.

[2] R.J. Bradford, C. Chen, J.H. Davenport, M. England, M. Moreno Maza, and D.J. Wilson. Truth Table Invariant Cylindrical Algebraic Decomposition by Regular Chains. In *Proceedings CASC 2014*, pages 44–58, 2014.

[3] R.J. Bradford and J.H. Davenport. Towards Better Simplification of Elementary Functions. In T. Mora, editor, *Proceedings ISSAC 2002*, pages 15–22, 2002.

[4] C.W. Brown and J.H. Davenport. The Complexity of Quantifier Elimination and Cylindrical Algebraic Decomposition. In C.W. Brown, editor, *Proceedings ISSAC 2007*, pages 54–60, 2007.

[5] R.J. Bradford, J.H. Davenport, M. England, S. McCallum, and D.J. Wilson. Cylindrical Algebraic Decompositions for Boolean Combinations. In *Proceedings ISSAC 2013*, pages 125–132, 2013.

[6] R.J. Bradford, J.H. Davenport, M. England, S. McCallum, and D.J. Wilson. Truth Table Invariant Cylindrical Algebraic Decomposition. `http://arxiv.org/abs/1401.0645`, 2014.

[7] R.J. Bradford, J.H. Davenport, M. England, and D.J. Wilson. Optimising Problem Formulation for Cylindrical Algebraic Decomposition. In J. Carette *et al.*, editor, *Proceedings CICM 2013*, pages 19–34, 2013.

[8] C.W. Brown. Guaranteed Solution Formula Construction. In S. Dooley, editor, *Proceedings ISSAC '99*, pages 137–144, 1999.

[9] C. Chen and M. Moreno Maza. An Incremental Algorithm for Computing Cylindrical Algebraic Decompositions. `http://arxiv.org/abs/1210.5543`, 2012.

[10] C. Chen and M. Moreno Maza. Quantifier Elimination by Cylindrical Algebraic Decomposition Based on Regular Chains. In K. Nabeshima, editor, *Proceedings ISSAC 2014*, pages 91–98, 2014.

[11] C. Chen, M. Moreno Maza, B. Xia, and L. Yang. Computing Cylindrical Algebraic Decomposition via Triangular Decomposition. In J. May, editor, *Proceedings ISSAC 2009*, pages 95–102, 2009.

[12] G.E. Collins. Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition. In *Proceedings 2nd. GI Conference Automata Theory & Formal Languages*, pages 134–183, 1975.

[13] G.E. Collins. Quantifier elimination by cylindrical algebraic decomposition — twenty years of progess. In B.F. Caviness and J.R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pages 8–23. Springer Verlag, Wien, 1998.

[14] M. Coste and M.-F. Roy. Thom's Lemma, the Coding of Real Algebraic Numbers and the Computation of the Topology of Semi-Algebraic Sets. *J. Symbolic Comp.*, 5:121–129, 1988.

[15] J.H. Davenport. Computer Algebra for Cylindrical Algebraic Decomposition. Technical Report TRITA-NA-8511 NADA KTH Stockholm (Reissued as Bath Computer Science Technical report 88-10), 1985.

[16] J.H. Davenport. The geometry of $\mathbf{C}^n$ is important for the algebra of elementary functions. In M. Jowsig and N. Takayama, editors, *Algebra Geometry and software systems*, pages 207–224. Springer, 2003.

[17] J.H. Davenport and J. Heintz. Real Quantifier Elimination is Doubly Exponential. *J. Symbolic Comp.*, 5:29–35, 1988.

[18] A. Dolzmann, A. Seidl, and Th. Sturm. Efficient Projection Orders for CAD. In J. Gutierrez, editor, *Proceedings ISSAC 2004*, pages 111–118, 2004.

[19] M. England, R. Bradford, C. Chen, J.H. Davenport, M.M. Maza, and D.J. Wilson. Problem formulation for truth-table invariant cylindrical algebraic decomposition by incremental triangular decomposition. In S.M. Watt *et al.*, editor, *Proceedings CICM 2014*, pages 45–60, 2014.

[20] M. England, R. Bradford, and J.H. Davenport. Improving the use of equational constraints in cylindrical algebraic decomposition. `http://arxiv.org/abs/1501.04466`, 2015.

[21] M. England, R. Bradford, J.H. Davenport, and D.J. Wilson. Choosing a Variable Ordering for Truth-Table Invariant Cylindrical Algebraic Decomposition by Incremental Triangular Decomposition. In *Proceedings ICMS 2014*, pages 450–457, 2014.

[22] M. England, D.J. Wilson, R. Bradford, and J.H. Davenport. Using the Regular Chains Library to build cylindrical algebraic decompositions by projecting and lifting. In *Proceedings ICMS 2014*, pages 458–465, 2014.

[23] N. Fitchas, A. Galligo, and J. Morgenstern. Precise sequential and parallel complexity bounds for the quantifier elimination over algebraic closed fields. *J. Pure and Applied Algebra*, 67:1–14, 1990.

[24] A. Gabrielov and N. Vorobjov. Complexity of computations with Pfaffian and Noetherian functions. In *Normal Forms, Bifurcations and Finiteness Problems in Differential Equations*, pages 211–250. Kluwer, 2004.

[25] Z. Huang, M. England, D. Wilson, J.H. Davenport, L.C. Paulson, and J. Bridge. Applying machine learning to the problem of choosing a heuristic to select the variable ordering for cylindrical algebraic decomposition. In S.M.Watt *et al.*, editor, *Proceedings CICM 2014*, pages 92–107, 2014.

[26] H. Hong. An Improvement of the Projection Operator in Cylindrical Algebraic Decomposition. In S. Watanabe and M. Nagata, editors, *Proceedings ISSAC '90*, pages 261–264, 1990.

[27] D. Jovanović and L. de Moura. Solving Non-Linear Arithmetic. In *Proceedings IJCAR 2012*, pages 339–354, 2012.

[28] D. Jovanović and L. de Moura. Solving non-linear arithmetic. *ACM Communications in Computer Algebra*, 46(3/4):104–105, 2013.

[29] Yu.V. Matiyasevich. Enumerable sets are Diophantine. *Soviet Math. Doklady 2*, 11:354–358, 1970.

[30] S. McCallum. An Improved Projection Operation for Cylindrical Algebraic Decomposition. Technical Report 548 Computer Science University Wisconsin at Madison, 1985.

[31] S. McCallum. On Projection in CAD-Based Quantifier Elimination with Equational Constraints. In S. Dooley, editor, *Proceedings ISSAC '99*, pages 145–149, 1999.

[32] N. Phisanbut, R.J. Bradford, and J.H. Davenport. Geometry of Branch Cuts. *Communications in Computer Algebra*, 44:132–135, 2010.

[33] N. Phisanbut. *Practical Simplification of Elementary Functions using Cylindrical Algebraic Decomposition.* PhD thesis, University of Bath, 2011.

[34] J.T. Schwartz and M. Sharir. On the "Piano-Movers" Problem: II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds. *Adv. Appl. Math.*, 4:298–351, 1983.

[35] A. Tarski. *A Decision Method for Elementary Algebra and Geometry.* 2nd ed., Univ. Cal. Press. Reprinted in *Quantifier Elimination and Cylindrical Algebraic Decomposition* (ed. B.F. Caviness & J.R. Johnson), Springer-Verlag, Wein-New York, 1998, pp. 24–84., 1951.

[36] D.J. Wilson, R.J. Bradford, and J.H. Davenport. Speeding up Cylindrical Algebraic Decomposition by Gröbner Bases. In J. Jeuring *et al.*, editor, *Proceedings CICM 2012*, pages 279–293, 2012.

[37] D.J. Wilson, R.J. Bradford, J.H. Davenport, and M. England. Cylindrical Algebraic Sub-Decompositions. *Mathematics in Computer Science*, 8:263–288, 2014.

[38] D.J. Wilson, J.H. Davenport, M. England, and R.J. Bradford. A "Piano Movers" Problem Reformulated. In *Proceedings SYNASC 2013*, pages 53–60, 2013.