

Topics in Cylindrical Algebraic Decomposition

Corin Lee

Department of Mathematical Sciences
University of Bath

17th Conference on Intelligent Computer Mathematics
Montréal, Québec, Canada
August 6, 2024

Overview

- An introduction to Cylindrical Algebraic Decomposition:
 - Definitions, applications and construction algorithms.
- Highlight of current work enhancing CAD in computer algebra systems:
 - Comparing and analysing construction methods, and tailoring algorithms for specific cases.
 - Implementations and tools to spread and enhance understanding.

Cylindrical Algebraic Decomposition

First presented in [Col75] as an algorithm to tackle real QE problems (taking a formula with quantifiers \forall, \exists and obtaining an equivalent one without quantifiers).

Applications in various fields: robotics [Man+12], economics [MDE18] and biology [RS21].

Cylindrical Algebraic Decomposition

First presented in [Col75] as an algorithm to tackle real QE problems (taking a formula with quantifiers \forall, \exists and obtaining an equivalent one without quantifiers).

Applications in various fields: robotics [Man+12], economics [MDE18] and biology [RS21].

Input: Set of polynomials $\mathcal{F}_n = \{f_1, \dots, f_r\}$ in variables x_1, \dots, x_n .

Cylindrical Algebraic Decomposition

First presented in [Col75] as an algorithm to tackle real QE problems (taking a formula with quantifiers \forall, \exists and obtaining an equivalent one without quantifiers).

Applications in various fields: robotics [Man+12], economics [MDE18] and biology [RS21].

Input: Set of polynomials $\mathcal{F}_n = \{f_1, \dots, f_r\}$ in variables x_1, \dots, x_n .

Output: $\text{CAD}(\mathcal{F}_n)$, a decomposition of \mathbb{R}^n into *cells* which have some sort of invariance with respect to \mathcal{F}_n , are described by polynomial constraints and are cylindrically arranged.

Cylindrical Algebraic Decomposition

First presented in [Col75] as an algorithm to tackle real QE problems (taking a formula with quantifiers \forall, \exists and obtaining an equivalent one without quantifiers).

Applications in various fields: robotics [Man+12], economics [MDE18] and biology [RS21].

Input: Set of polynomials $\mathcal{F}_n = \{f_1, \dots, f_r\}$ in variables x_1, \dots, x_n .

Output: $\text{CAD}(\mathcal{F}_n)$, a decomposition of \mathbb{R}^n into *cells* which have some sort of invariance with respect to \mathcal{F}_n , are described by polynomial constraints and are cylindrically arranged.

Examples of invariance include:

- *sign-invariant*: input polynomials have constant sign in each cell i.e. $+$, $-$ or 0 ,
- *truth-invariant*: sign-invariant cells are marked true/false according to *Tarski formula* \mathcal{T}_n : boolean combinations of these f_i with sign conditions ($<$, $>$, $=$) and quantifiers (\forall, \exists), e.g. $\exists x (f_1 > 0) \wedge (f_2 > 0)$.

Cylindrical Algebraic Decomposition

- **Decomposition** of \mathbb{R}^n into finitely many *cells* C_i where $\bigcup C_i = \mathbb{R}^n$, $C_i \cap C_j = \emptyset$ if $i \neq j$,

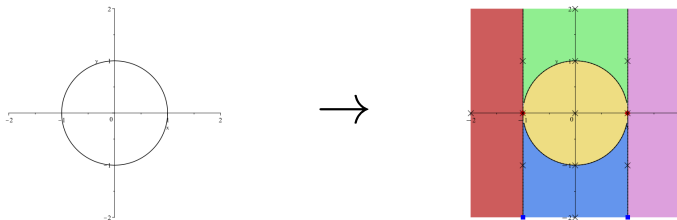


Figure: The graph of $f = x^2 + y^2 - 1$ and an associated sign-invariant CAD of \mathbb{R}^2 .

Cylindrical Algebraic Decomposition

- **Decomposition** of \mathbb{R}^n into finitely many *cells* C_i where $\bigcup C_i = \mathbb{R}^n$, $C_i \cap C_j = \emptyset$ if $i \neq j$,
 - and for $0 \leq j \leq n$, a *j-cell* in \mathbb{R}^n is a subset of \mathbb{R}^n that is homeomorphic to \mathbb{R}^j .

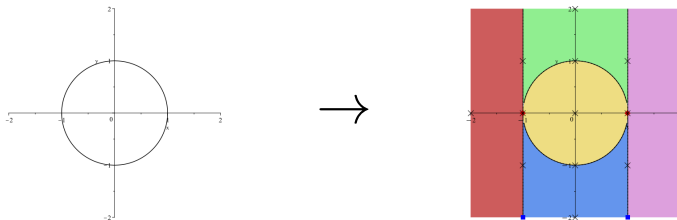


Figure: This CAD is made up of 13 cells: five 2-cells, six 1-cells and two 0-cells. The black crosses represent the sample points for each cell.

Cylindrical Algebraic Decomposition

- **Decomposition** of \mathbb{R}^n into finitely many *cells* C_i where $\bigcup C_i = \mathbb{R}^n$, $C_i \cap C_j = \emptyset$ if $i \neq j$,
 - and for $0 \leq j \leq n$, a *j-cell* in \mathbb{R}^n is a subset of \mathbb{R}^n that is homeomorphic to \mathbb{R}^j .
- **Cylindrical:** The *projections* of any two cells into lower-dimensional space are either equal or disjoint (they 'stack up' over lower-dimensional cells).

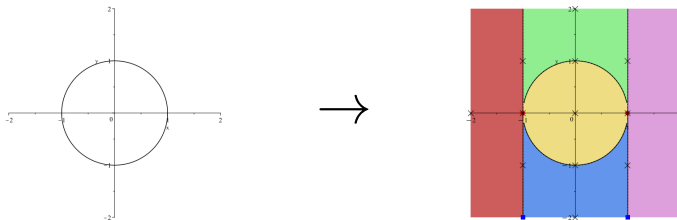


Figure: These cells stack in *cylinders* over the 5 cells of the CAD of \mathbb{R}^1 (the two points at the bottom and the regions between them).

Cylindrical Algebraic Decomposition

- **Decomposition** of \mathbb{R}^n into finitely many *cells* C_i where $\bigcup C_i = \mathbb{R}^n$, $C_i \cap C_j = \emptyset$ if $i \neq j$,
 - and for $0 \leq j \leq n$, a *j-cell* in \mathbb{R}^n is a subset of \mathbb{R}^n that is homeomorphic to \mathbb{R}^j .
- **Cylindrical:** The *projections* of any two cells into lower-dimensional space are either equal or disjoint (they 'stack up' over lower-dimensional cells).
- **Algebraic:** Each cell can be described by a finite set of polynomial equations and inequalities.

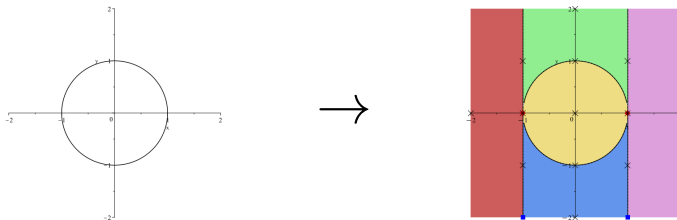


Figure: Each cell is described by constraints on x and y .

Projection and Lifting

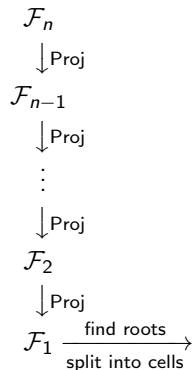
Projection Phase: Apply a *projection operator* Proj to \mathcal{F}_n to obtain a new set \mathcal{F}_{n-1} in $n - 1$ variables. Repeat this until one obtains \mathcal{F}_1 .

$$\begin{array}{c} \mathcal{F}_n \\ \downarrow \text{Proj} \\ \mathcal{F}_{n-1} \\ \downarrow \text{Proj} \\ \vdots \\ \downarrow \text{Proj} \\ \mathcal{F}_2 \\ \downarrow \text{Proj} \\ \mathcal{F}_1 \end{array}$$

Projection and Lifting

Projection Phase: Apply a *projection operator* Proj to \mathcal{F}_n to obtain a new set \mathcal{F}_{n-1} in $n - 1$ variables. Repeat this until one obtains \mathcal{F}_1 .

Base Phase: Decompose \mathbb{R}^1 into the roots of each polynomial in \mathcal{F}_1 and the open intervals either side of them to obtain $\text{CAD}(\mathcal{F}_1)$.

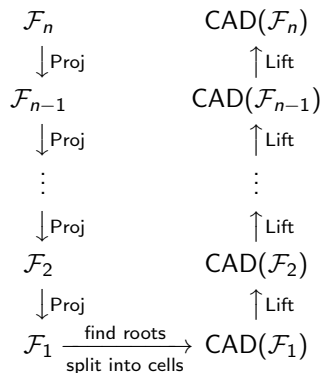


Projection and Lifting

Projection Phase: Apply a *projection operator* Proj to \mathcal{F}_n to obtain a new set \mathcal{F}_{n-1} in $n - 1$ variables. Repeat this until one obtains \mathcal{F}_1 .

Base Phase: Decompose \mathbb{R}^1 into the roots of each polynomial in \mathcal{F}_1 and the open intervals either side of them to obtain $\text{CAD}(\mathcal{F}_1)$.

Lifting Phase: For each cell C of $\text{CAD}(\mathcal{F}_1)$, substitute a *sample point* into \mathcal{F}_2 to obtain a new set of univariate polynomials. Decompose these to obtain a set of cells that lie above C , and the full collection of these cells for every C forms $\text{CAD}(\mathcal{F}_2)$. Repeat this process until one obtains $\text{CAD}(\mathcal{F}_n)$.



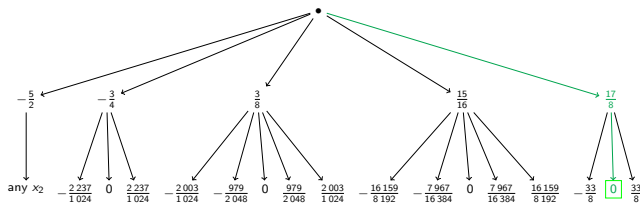
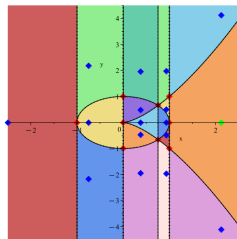
Current work

- Tools and implementations:
 - Open CAD in *Macaulay2*
 - CAD graphing tool in *Maple*
- Focus on a specific type of input polynomials (multilinearity)
 - Definition
 - Customised construction algorithm
- Competing construction algorithms (PL-CAD vs RC-CAD)
 - Timings of implementations
 - Complexity analysis of RC-CAD

The CADecomposition Package for *Macaulay2*

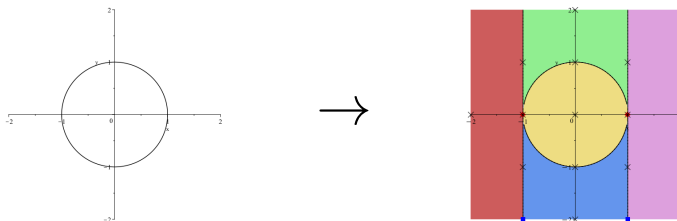
- First implementation of CAD in *M2*.
- Produces tree of sample points in \mathbb{Q}^n representing the open cells of a CAD of \mathbb{R}^n for input polynomials in $\mathbb{Q}[x_1, \dots, x_n]$.
- Able to: return full tree or branches, check these for points where all polynomials are positive, step through the algorithm.
- Improves/fixes RealRoots' real root isolation.

Output: \mathcal{S}_n , a tree of sample points in n variables which represent each *open cell* of $\text{CAD}(\mathcal{F}_n)$.



CAD Graphing in *Maple*

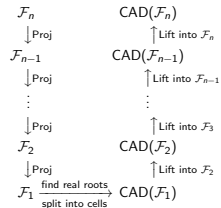
- Production of commands that graph a CAD of \mathbb{R}^2 .
- Currently takes output of *QuantifierElimination's CylindricalAlgebraicDecompose*, returning 2-cells as coloured regions, 1-cells as black lines and 0-cells as red points.
- Also returns sample points and 1d projection.
- Future aims are to produce a 1d/2d/3d projection of any higher-dimension CAD, take output from *RegularChains* package, and offer truth-invariant CAD colouring.



Multilinearity

- Property where for $\mathcal{F}_n = \{f_1, \dots, f_r\}$ in variables x_1, \dots, x_n , we have some $1 \leq r \leq n$ variables of maximum degree 1 in any f_i .
- Such variables will be simpler to project, not requiring certain parts of the projection operator and producing a smaller number of cells - so 'projecting away' these variables first should produce a 'nicer' CAD,
- Such polynomials often arise in practical applications, so optimising for these is worthwhile.

CAD Construction Algorithms

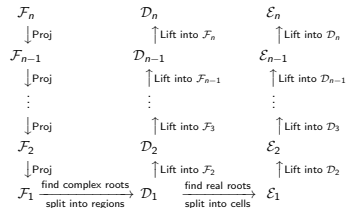


PL-CAD

Projection: Apply a *projection operator* Proj to \mathcal{F}_n to obtain a new set \mathcal{F}_{n-1} in $n - 1$ variables. Repeat this until one obtains \mathcal{F}_1 .

Construct CD \mathcal{D}_n : Decompose \mathbb{C}^1 into $r + 1$ regions, the sets of roots of the r polynomials of \mathcal{F}_1 and the set of all other points. Substitute sample points of each region into \mathcal{F} to obtain a new set of univariate polynomials and decompose again. The full collection is \mathcal{D}_2 . Repeat to \mathcal{D}_n .

Construct CAD \mathcal{E}_n : Decompose \mathbb{R}^1 into the roots of each polynomial in \mathcal{D}_1 and the open intervals either side of them to obtain \mathcal{E}_1 . Substitute sample points of each cell of \mathcal{E}_1 into every region of \mathcal{D}_2 above it to obtain a new set of univ. polynomials, decompose again. Full collection is \mathcal{E}_2 . Repeat to \mathcal{E}_n .



RC-CAD

PL-CAD vs RC-CAD

- RC-CAD has been implemented in *Maple* via the *RegularChains* package.
- Papers on RC-CAD are promising, showing RC-CAD often performing better than implementations of PL-CAD, and producing fewer cells.
- Despite this, it has not been extensively compared, or run against the newest implementation of RC-CAD in *Maple*.
- Preliminary timings have been run, but aim is to find a fair way to compare both algorithms and attempt to understand when one algorithm may work better than another.
- Future aim is also to perform complexity analysis on RC-CAD in order to identify conditions on which it will run faster.

References I

- [Col75] George E. Collins. “Quantifier elimination for real closed fields by cylindrical algebraic decomposition”. In: *Lecture Notes in Computer Science* (1975).
- [Man+12] Montserrat Manubens et al. “Cusp Points in the Parameter Space of Degenerate 3-RPR Planar Parallel Manipulators”. In: *Journal of Mechanisms and Robotics* 4.4 (2012).
- [MDE18] Casey B. Mulligan, James H. Davenport, and Matthew England. “TheoryGuru: A Mathematica Package to Apply Quantifier Elimination Technology to Economics”. In: *Mathematical Software – ICMS 2018*. Ed. by James H. Davenport et al. Springer International Publishing, 2018.
- [RS21] Gergely Röst and AmirHosein Sadeghimanesh. “Exotic Bifurcations in Three Connected Populations with Allee Effect”. In: *International Journal of Bifurcation and Chaos* 31.13 (2021).

The End



Figure: https://github.com/cel34-bath/CAD_M2_ICMS_2024