Quantifier Elimination for Cylindrical Algebraic Decomposition

Final presentation

Oriol Brufau, Aina Cuxart, Enric Cosp, Raquel Pérez, Ferran Pujol.
Teacher: Jordi Saludes

14.12.2016

Table of Contents

Reminder

- Goal
- What is Quantifier Elimination?
- What is Cylindrical Algebraic Decomposition?
- Weaponry: Python, Sympy, Github

Quantifier elimination

A quantifier-free formula is an expression consisting of

- Polynomial equations: $f(x) = 0$
- Inequalities: $f(x) \leq 0$
- "And" operator: $\wedge$
- "Or" operator: $\vee$
- "Implies" operator: $\implies$.

No variable is quantified by $\forall$ nor $\exists$.

Theorem (Tarski-Seidenberg)

For every first-order formula over the real field there exists an equivalent quantifier-free formula and an explicit algorithm to compute this quantifier-free formula.

Cylindrical Algebraic Decomposition

- A CAD is a partition of $\mathbb{R}^n$ into cylindrical cells, over which polynomials have constant signs.
- A cell is cylindrical if it has the form $S \times \mathbb{R}^k$
- A sample point per cell can be used to determine the sign of the polynomials in the cell.
- The CAD associated to a formula depends only on its quantifier-free part. We can use the CAD to evaluate its truth value, and to perform quantifier elimination.

1. **Projection phase**
   We compute successive sets of polynomials in $n-1$, $n-2, \ldots, 1$ variables. The main idea is, given an input set of polynomials, to compute at each step a new set of polynomials obtained by eliminating one variable at a time.
2. **Lifting phase**
   We construct a decomposition of $\mathbb{R}$, at the lowest level of projection, after all but one variable have been eliminated.
   This decomposition of $\mathbb{R}$ is extended to a decomposition of $\mathbb{R}^n$.
3. Optionally, **formula construction**

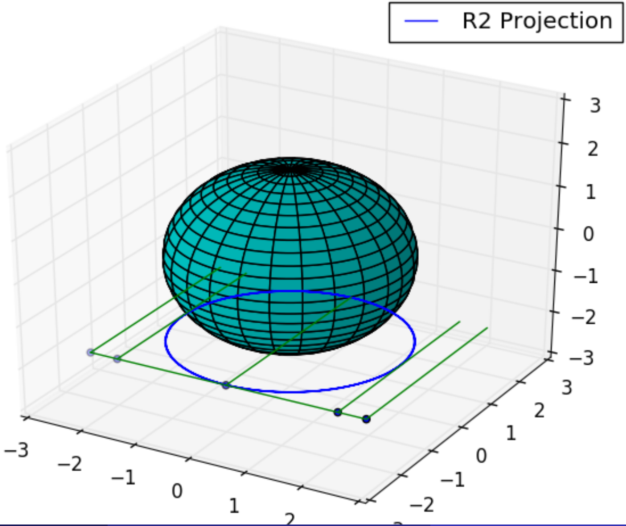
Table of Contents

What have we accomplished?

- We learned python and how to use github.
- We made a full implementation of cylindrical algebraic decomposition such that:
  - The input is a set of polynomials that represent a region of the space. This polynomials are operated by the *Projection* part.
  - *Projection* does the cylindrical projection of the polynomials and makes a projection factor set.
  - Now the *Lifting* part of the code operates the projection factor set and makes the final decomposition.
  - That's our output, a cad that contains all the cells of the region.

Table of Contents

Example 1: the sphere



- Input: $A = \{\ Poly(x^2 + y^2 + z^2 - 4)\ \}$
- Output:
  - $A = \{\ Poly(x^2 + y^2 + z^2 - 4)\ \}$
  - $PROJ(A) = \{\ Poly(x^2 + y^2 - 4)\ \}$
  - $PROJ^2(A) = \{\ Poly(x+2),\quad Poly(x-2)\ \}$

Example 1: the sphere

Extension

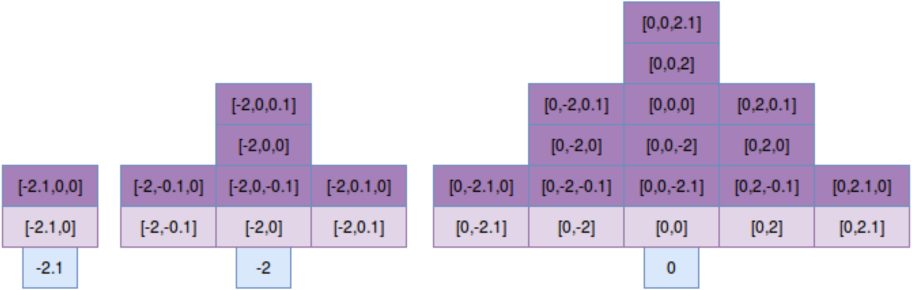


Table of Contents

Example 2: a paraboloid and a cylinder

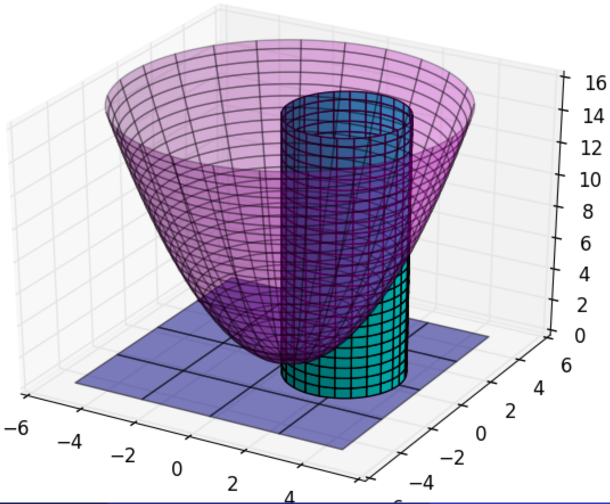


Table of Contents

- Different definitions of the same function
- Special cases
- Iterating the projection function
- Different process described in different references

- The issue with eval function.
- Compute with algebraic numbers
- Learn algebra.
- Making tests.

- The issue with eval function.
  - Symbolic method to find roots of polynomials: *only for rational coefficients*
- Computing with algebraic numbers
  - We represent an algebraic number with a polynomial and and interval
  - Operations on algebraic numbers become operations with polynomials

Algorithm

IN: Polynomial with *algebraic* coefficients with roots $\{\alpha_i\}$
OUT: A polynomial with *rational* coefficients with $\{\alpha_i\}$ among its roots

Table of Contents

Left to do

- Optimization of the algorithm
- Refine the code to handle every case correctly
- Adapt code format to Python and Sympy standards
- Implement the quantifier elimination using the cad.

Table of Contents

Behind the scenes

Theorem

Let $\mathcal{D}$ be a domain and $0 \leq \mathrm{j} \leq \min(\mathrm{p}, \mathrm{q})$ if $\mathrm{p} \neq \mathrm{q}$ (resp. $0 \leq \mathrm{j} \leq \mathrm{p}$ - 1 if $\mathrm{p} = \mathrm{q}$). Then $\deg(\gcd(\mathrm{P}, \mathrm{Q})) \geq \mathrm{j}$ if and only if

$$sRes_0(P, Q) = \cdots = sRes_{j-1}(P, Q) = 0$$

Table of Contents

QA

Questions?