An output-sensitive algorithm for computing projections of resultant polytopes

Ioannis Z. Emiris

Joint work with V. Fisikopoulos, C. Konaxis* and L. Peñaranda

Dept Informatics & Telecoms, University of Athens
∗ currently with Univeristy of Crete

CGL Meeting Zürich, 15.Dec.2011

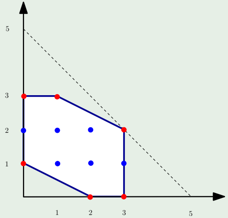
Sparse elimination

Definition

The Newton polytope $N(f)$ of polynomial f in $x_1, \ldots, x_n$ f is the convex hull of the set of exponent vectors appearing in f.

Example

$f(x,y) = 8y + xy - 24y^2 - 16x^2 + 220x^2y - 34xy^2 - 84x^3y + 6x^2y^2 - 8xy^3 + 8x^3y^2 + 8x^3 + 18y^3$



The Newton polytope is the equivalent notion of total degree in sparse elimination theory.

Sparse resultant

Definition

The sparse resultant $\mathcal{R}$ of polynomials $f_0, \ldots, f_n$, where

$$f_i = \sum_{a \in A_i} c_{ia} x^a, \quad x^a = (x_1^{a_1}, \ldots, x_n^{a_n}),$$

with symbolic c_{ia} and exponents $A_i \subset \mathbb{Z}^n$, is the unique irreducible integer polynomial in the c_{ij}, which vanishes iff the f_i have a common complex root.

Example

Let $A_0 = A_1 = \{0, 1\}$. Then $f_0 = c_{11}x + c_{12}$, $f_1 = c_{21}x + c_{22}$.

$$\mathcal{R} = c_{11}c_{22} - c_{12}c_{21} \in \mathbb{Z}[c_{11}, c_{12}, c_{21}, c_{22}]$$

Problem

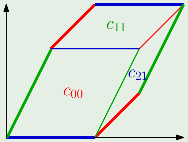
Compute the Newton polytope $N(\mathcal{R})$ or $\pi(N(\mathcal{R})) = \Pi$ for some orthogonal projection π; $\Pi = N(\pi(\mathcal{R}))$ for sufficiently generic π.

Mixed subdivisions

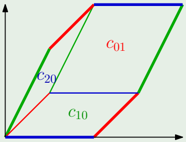
A fine mixed subdivision of $P = \mathrm{CH}(A_0 + \cdots + A_n) \subset \mathbb{R}^n$, is any family of cells partitioning P and intersecting properly as Minkowski sums of faces $F_i \subset \mathrm{CH}(A_i)$, where $\dim(\sum_i F_i) = \sum_i \dim F_i$.

Example

$$f_0 := c_{00} - c_{01} st, \;\; f_1 := c_{10} - c_{11} st^2, \;\; f_2 := c_{20} - c_{21} s^2$$



$c_{00}^4 c_{11}^2 c_{21}$



$4c_{01}^4 c_{10}^2 c_{20}$

$$\mathcal{R} = -c_{00}^4 c_{11}^2 c_{21} + c_{01}^4 c_{10}^2 c_{20}.$$

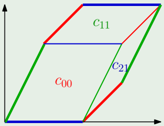
Resultant polytope

Theorem (Sturmfels'94)

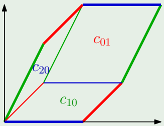
There exists a surjection ρ from the regular fine mixed subdivisions of P onto the vertices of $N(\mathcal{R})$. Subdivisions S, S' are equivalent iff $\rho_S = \rho_{S'}$.

Example

All cells mixed:



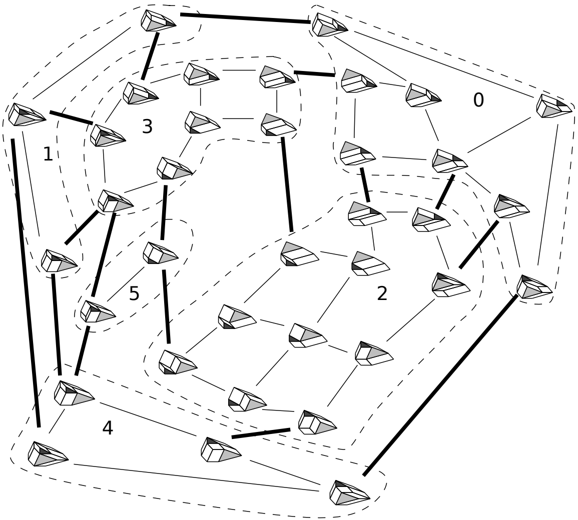
$c_{00}^4 c_{11}^2 c_{21}$

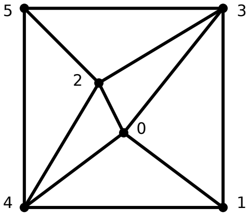


$4c_{01}^4 c_{10}^2 c_{20}$

$$\rho_S = (4, 0, 0, 2, 0, 1)$$
$$\rho_{S'} = (0, 4, 2, 0, 1, 0)$$

Example





(L) Graph of mixed subdivisions for $|\mathcal{A}| = 3 + 2 + 3$ points $\in \mathbb{Z}^2$: the 6 equivalence classes are dotted. (R) $N(\mathcal{R}) \subset \mathbb{R}^3$.

Existing work

Theory

- Cayley trick: Mixed subdivisions of P correspond bijectively to regular triangulations of $\mathcal{A} := \bigcup_{i=0}^{n}(A_i \times \{e_i\}) \subset \mathbb{Z}^{2n}$: e_i form an affine basis of $\mathbb{R}^n$.
- Michiels & Verschelde [DCG'99] define equivalence classes of mixed subdivisions mapping onto $N(\mathcal{R})$ vertices.
- Michiels & Cools [DCG'00] describe all Minkowski summands of $\Sigma(\mathcal{A})$ including $N(\mathcal{R})$.

Algorithms

- TOPCOM [Rambau '02] computes all mixed subdivisions. Inefficient for resultant polytope computation.
- Tropical geometry [Sturmfels-Yu '08] leads to algorithms for the polytope of (specialized) resultant based on the `GFan` library [Jensen-Yu '11].

Main Oracle

Vertex$\Pi(\mathcal{A}, w)$

Input: $\mathcal{A} \subset \mathbb{Z}^{2n}$, lifting $w \in (\mathbb{R}^m)^\times$, $\pi : \mathbb{R}^{|\mathcal{A}|} \to \mathbb{R}^m : N(\mathcal{R}) \mapsto \Pi$

Output: vertex $\in \Pi$, extremal wrt w

1. let $\widehat{w} = (w, 0) \in \mathbb{R}^{2n}$, use it as lifting for regular mixed subdivision S of P
2. refine S into fine regular mixed subdivision $\mathcal{T}$
3. compute $\rho_{\mathcal{T}} \in \mathbb{N}^{|\mathcal{A}|}$, return $\pi(\rho_{\mathcal{T}}) \in \mathbb{N}^m$

Incremental Algorithm

Definition

Supporting hyperplane H of polytope is *legal* if it supports facet $\subset \Pi$, else *illegal*

ComputeΠ

Input: A_i, π, H-rep. Q_H, V-rep. Q_V of $Q \subseteq \Pi$

Output: H-rep. Q_H, V-rep. Q_V of $Q = \Pi$

1. `while` $\exists$ illegal hyperplane $H \subset Q_H$ with (distinct) outer normal w `do`
2. compute $v = \text{Vertex}\Pi(\mathcal{A}, w)$, $Q_V \leftarrow Q_V \cup \{v\}$
3. `if` $v \in H$ `then` H is legal `else` $[v \notin Q_V \cap H]$ $Q_H \leftarrow \text{CH}(Q_H \cup \{v\})$

Inner/Outer approximation

At any step, Q is inner approximation; can obtain outer approximation.

Complexity

Theorem

We compute the vertex- and halfspace-representations of $\Pi = \pi(N(\mathcal{R}))$*, as well as a triangulation* S *of* Π*, in*

$$O^*(\, m^5 \, |vtx(\Pi)| \cdot |S|^2),$$

where $m = \dim \Pi$.

Elements of proof

- All computation in dimension $\leq m$.
- Compute $\leq \mathrm{vtx}(\Pi) + \mathrm{fct}(\Pi)$ mixed subdivisions of P.
- Beneath-Beyond algorithm for converting V-rep. to H-rep. (bottleneck)

ResPol Implementation

Tools

C++, CGAL, `triangulation` [Boissonnat,Devillers,Hornus], `extreme_points_d` [Gärtner]

- `extreme_points_d` preprocesses A_i.
- regular triangulations of $\mathcal{A}$ (Cayley) correspond bijectively to mixed subdivisions of P.
- compute "placing" triangulations of $\mathcal{A}$: points lifted to 0 are placed only at first iteration of ComputeΠ.
- `triangulation` also used to maintain V-rep. of Π.
- all cells of the triangulated facets of Π have same normal: `STL set` maintains them to ensure one normal per facet.

Hashing of determinantal predicates

Observation

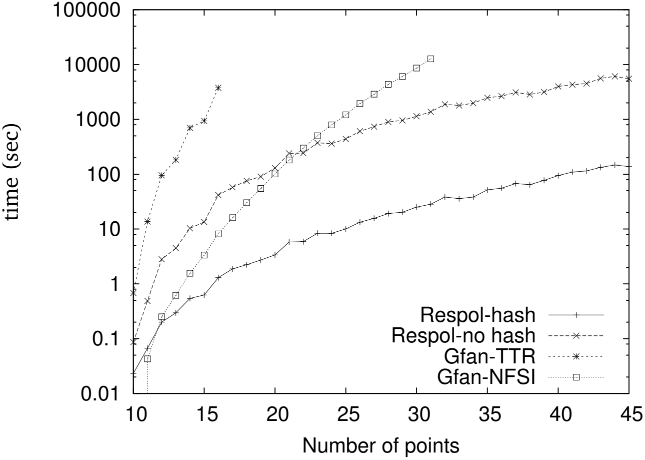
- algorithm computes many *similar* determinantal predicates (orientation, volume)
- use Laplace (cofactor) expansion to expand along last 2 rows
- set of possible minors is same in every call of a predicate by VertexΠ

Idea

- Hash a minor the first time you compute it, never compute it again.
- If all needed minors computed, orientation $= O(n^2)$, volume $= O(n)$.
- Storage issue: clean up hashtable from time to time.

Hashing, and `Gfan`

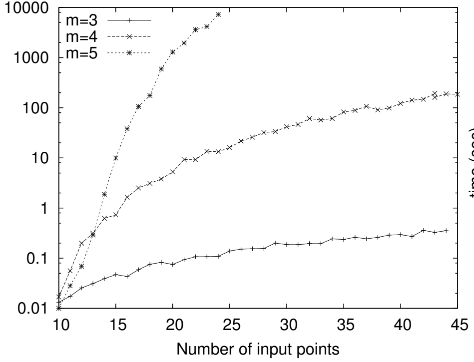
- *hashing determinants* speeds ≤ 10-100x when *dim*$(\Pi) = 3, 4$
- faster than `Gfan` [Yu-Jensen'11] for *dim*$\Pi \leq 6$, else slower ≤ 2x

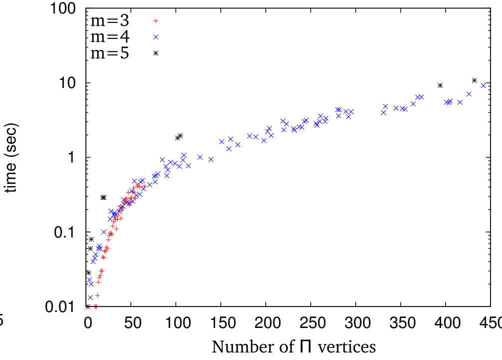


dim$(\Pi) = 4$:

Output-sensitivity

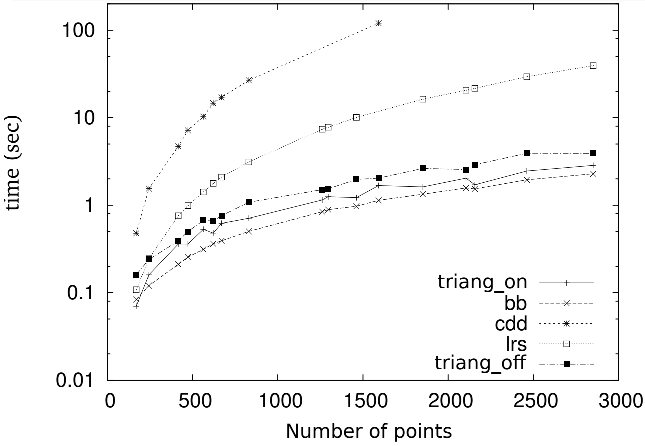
- computed points by VertexΠ are all vertices of Π: each either extends Π, or legalizes a hyperplane
- output vertices bound polynomially the output triangulation size
- subexponential runtime wrt to input points (left), output vertices (right)





Convex hull implementations

- From V- to H-rep. of Π.
- `triangulation (on/off-line)`, polymake `beneath-beyond`, `cdd`, `lrs`



$dim(\Pi) = 4$

Preliminary runtimes

input	m	3	3	4	4	5	5
	$\lvert\mathcal{A}\rvert$	200	490	20	30	17	20
exact	#vtx(Π)	98	133	416	1296	1674	5093
	time	2.03	5.87	3.72	25.97	51.54	239.96
approx. v(i)/v(o)	#vtx(Q_{in})	15	11	63	121	–	–
	vol(Q_{in})/vol(Π)	0.96	0.95	0.93	0.94	–	–
	vol(Q_{out})/vol(Π)	1.02	1.03	1.04	1.03	–	–
	time	0.15	0.22	0.37	1.42	> 10hr	> 10hr
rand.$\in$cone till no new	#vtx(Q_{in})	26	21	102	380	341	544
	vol(Q_{in})/vol(Π)	0.92	0.90	0.80	0.92	0.80	0.81
	vol(Q_{out})/vol(Π)	1.02	1.04	1.14	1.03	1.08	1.10
	time	0.16	0.31	1.54	23.66	59.87	211.50

References

The code

- http://respol.sourceforge.net

The paper

- http://arxiv.org/abs/1108.5985