High-dimensional polytopes defined by oracles: algorithms, computations and applications

Vissarion Fisikopoulos

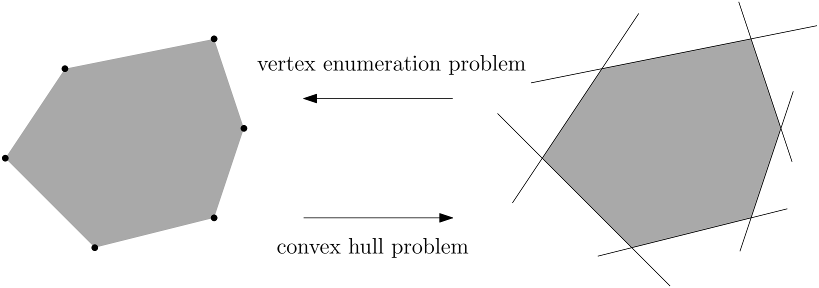
Dept. of Informatics & Telecommunications, University of Athens

PhD defence, Athens, 24.Apr.2014

Classical Polytope Representations

A convex polytope $P \subseteq \mathbb{R}^d$ can be represented as the

1. convex hull of a pointset $\{p_1, \ldots, p_n\}$ (V-representation)
2. intersection of halfspaces $\{h_1, \ldots, h_m\}$ (H-representation)



- These problems are equivalent by polytope duality.

Algorithmic Issues

- For general dimension d there is no polynomial algorithm for the convex hull (or vertex enumeration) problem since m can be $O\left(n^{\lfloor d/2 \rfloor}\right)$ [McMullen'70].

- It is open whether there exist a total poly-time algorithm for the convex hull (or vertex enumeration) problem, *i.e. runs in poly-time in* n, m, d.

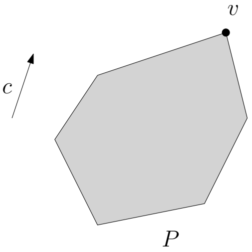
What is an Oracle?

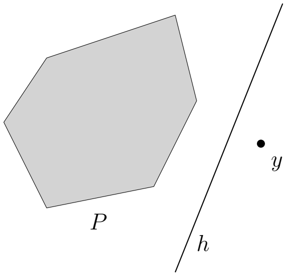
Polytope Oracles

Implicit representation for a polytope $P \subseteq \mathbb{R}^d$.

OPT_P: Given direction $c \in \mathbb{R}^d$ return the vertex $v \in P$ that maximizes $c^T v$.

SEP_P: Given point $y \in \mathbb{R}^d$, return yes if $y \in P$ otherwise a hyperplane h that separates y from P.





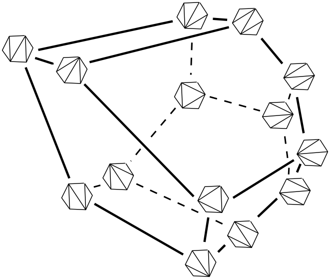
Why polytope Oracles?

- Polynomial time algorithms for combinatorial optimization problems using the ellipsoid method
  [Grötschel-Lovász-Schrijver'93]

- Randomized polynomial-time algorithms for approximating the volume of convex bodies
  [Dyer-Frieze-Kannan '90],. . . ,[Lovász-Vempala '04]

Our view of the Oracles

Resultant, Discriminant, Secondary polytopes



- Vertices $\rightarrow$ subdivisions of a pointset's convex hull
- OPT_P is available via a subdivision computation

- Applications in Computational Algebraic Geometry, Geometric Modelling, Optimization, Combinatorics

Outline

Introduction

An algorithm for computing projections of resultant polytopes

Edge-skeleton computation for polytopes defined by oracles

A practical volume algorithm for high dimensional polytopes

Combinatorics of 4-d resultant polytopes

High-dimensional predicates: algorithms and software

Outline

Introduction

An algorithm for computing projections of resultant polytopes

Edge-skeleton computation for polytopes defined by oracles

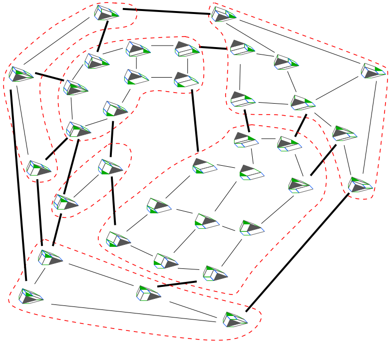
A practical volume algorithm for high dimensional polytopes

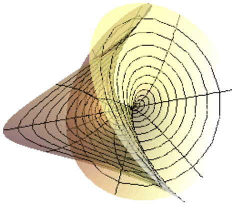
Combinatorics of 4-d resultant polytopes

High-dimensional predicates: algorithms and software

Main actor: resultant polytope

- Geometry: Minkowski summands of secondary polytopes, generalize Birkhoff polytopes
- Algebra: resultant expresses the solvability of polynomial systems
- Applications: resultant computation, implicitization of parametric hypersurfaces [Emiris, Kalinka, Konaxis, LuuBa '12]





Enneper's Minimal Surface

Polytopes and Algebra

- Given $n+1$ polynomials on n variables.

$f_0(x) = ax^2 + b$

$f_1(x) = cx^2 + dx + e$

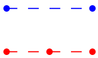
Polytopes and Algebra

- Given $n+1$ polynomials on n variables.
- Supports (set of exponents of monomials with non-zero coefficient) $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$.

$$f_0(x) = ax^2 + b$$
$$f_1(x) = cx^2 + dx + e$$

A_0

A_1



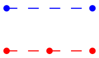
Polytopes and Algebra

- Given $n+1$ polynomials on n variables.
- Supports (set of exponents of monomials with non-zero coefficient) $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$.
- The resultant R is the polynomial in the coefficients of a system of polynomials which vanishes if there exists a common root in the torus of the given polynomials.

$$f_0(x) = ax^2 + b$$
$$f_1(x) = cx^2 + dx + e$$

A_0

A_1



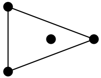
$$R(a,b,c,d,e) = ad^2b + c^2b^2 - 2caeb + a^2e^2$$

Polytopes and Algebra

- Given $n+1$ polynomials on n variables.
- Supports (set of exponents of monomials with non-zero coefficient) $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$.
- The resultant R is the polynomial in the coefficients of a system of polynomials which vanishes if there exists a common root in the torus of the given polynomials.
- The resultant polytope $N(R)$, is the convex hull of the support of R, i.e. the Newton polytope of the resultant.

$f_0(x) = ax^2 + b$ A_0 

$f_1(x) = cx^2 + dx + e$ A_1 

$R(a,b,c,d,e) = ad^2b + c^2b^2 - 2caeb + a^2e^2$ $N(R)$ 

Mixed subdivisions

A subdivision S of Minkowski sum $A_0 + A_1 + \cdots + A_n$ is

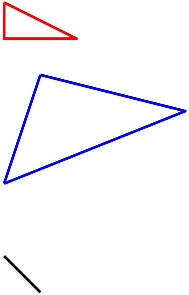
- mixed: cells are Minkowski sums of subsets of A_i's,
- fine: for each cell $\sigma = \sigma_0 + \cdots + \sigma_n$, $\dim(\sigma) = \sum_{i=0}^{n} \dim(\sigma_i)$

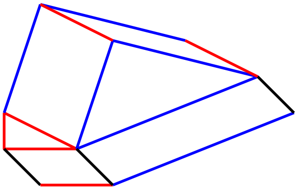
Example

A_0

A_1

A_2



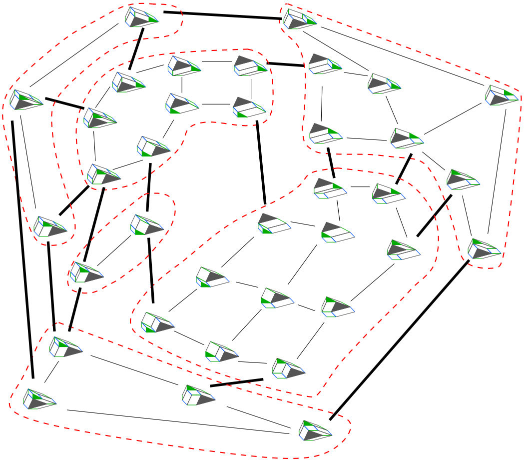


fine mixed subdivision S of $A_0 + A_1 + A_2$

Resultant polytope vertices

Theorem [GKZ'94, Sturmfels'94]

- many-to-one relation from regular fine mixed subdivisions of $A_0 + \cdots + A_n$ to $N(R)$ vertices
- one-to-one relation between regular fine mixed subdivisions and secondary polytope Σ vertices

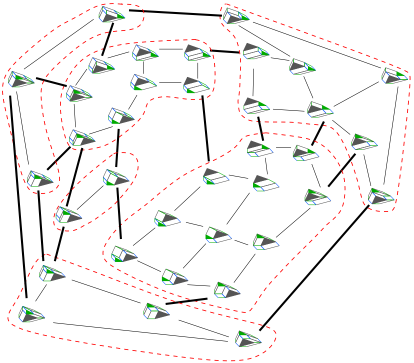


The idea of the algorithm

Input: $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$

Simplistic method:

- compute the vertices of secondary polytope Σ [Rambau '02]
- many-to-one relation between vertices of Σ and $N(R)$

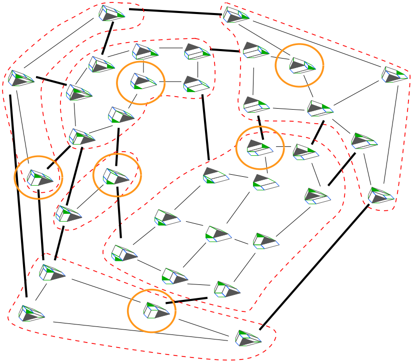


The idea of the algorithm

Input: $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$

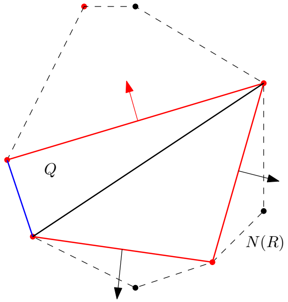
New Algorithm:

- Optimization oracle for $N(R)$ by subdivision computation
- Output sensitive: 1 subd. per $N(R)$ vertex + 1 per $N(R)$ facet
- Computes projections of $N(R)$, Σ



The Algorithm

- incremental
- first: compute conv.hull of $d+1$ aff. indep. vertices of $N(R)$
- step: call the oracle with outer normal vector of a halfspace
  $\rightarrow$ either validate this halfspace
  $\rightarrow$ or add a new vertex to the convex hull



The Algorithm

- incremental
- first: compute conv.hull of $d+1$ aff. indep. vertices of $N(R)$
- step: call the oracle with outer normal vector of a halfspace
  $\rightarrow$ either validate this halfspace
  $\rightarrow$ or add a new vertex to the convex hull

Theorem

H-, V-repr. & triang. T *of* $N(R)$ *can be computed in*

$$O(d^5 n s^2) \text{ arithmetic operations } + O(n+m) \text{ oracle calls}$$

n, m, s *are the number of vertices, facets of* $N(R)$, *cells of* T *resp.*

The Algorithm

- incremental
- first: compute conv.hull of $d+1$ aff. indep. vertices of $N(R)$
- step: call the oracle with outer normal vector of a halfspace
  $\rightarrow$ either validate this halfspace
  $\rightarrow$ or add a new vertex to the convex hull

Theorem

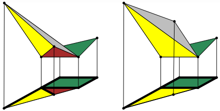
H-, V-repr. & triang. T *of* $N(R)$ *can be computed in*

$$O(d^5 n s^2) \text{ arithmetic operations } + O(n+m) \text{ oracle calls}$$

n, m, s *are the number of vertices, facets of* $N(R)$, *cells of* T *resp.*

BUT: s can be $O\left(n^{\lfloor d/2 \rfloor}\right)$

ResPol package



- C++
- Towards high-dimensional CGAL
- Propose *hashing of determinantal predicates* scheme: optimizing sequences of similar determinants (x100 speed-up)
- Computes 5-, 6- and 7-dimensional polytopes with 35K, 23K and 500 vertices, respectively, within 2hrs
- Computes polytopes of many important surface equations encountered in geometric modeling in $<$ 1sec, whereas the corresponding secondary polytopes are intractable

References

- Emiris, F, Konaxis, Peñaranda An output-sensitive algorithm for computing projections of resultant polytopes. Proc. of 28th ACM Annual Symposium on Computational Geometry, 2012, Chapel Hill, NC, USA.

- Emiris, F, Konaxis, Peñaranda An oracle-based, output sensitive algorithm for projections of resultant polytopes. International Journal of Computational Geometry and Applications (Special issue) World Scientific.

Outline

Vertex enumeration with edge-directions

Given $\mathrm{OPT_P}$ and a superset D of the edge directions $\mathrm{D(P)}$ of $\mathrm{P} \subseteq \mathbb{R}^{\mathrm{d}}$, compute the vertices P.

Proposition (Rothblum, Onn '07)

Let $\mathrm{P} \subseteq \mathbb{R}^{\mathrm{d}}$ *given by* $\mathrm{OPT_P}$, *and* $\mathrm{D} \supseteq \mathrm{D(P)}$. *All vertices of* P *can be computed in*

$$\mathrm{O}(|\mathrm{D}|^{\mathrm{d}-1}) \text{ calls to } \mathrm{OPT_P} + \mathrm{O}(|\mathrm{D}|^{\mathrm{d}-1}) \text{ arithmetic operations.}$$

Well-described polytopes and oracles

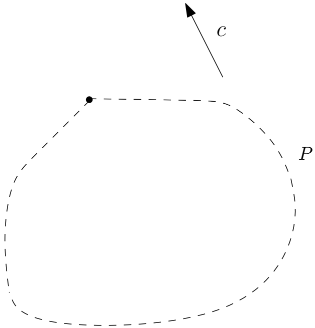
Definition

A polytope $P \subseteq \mathbb{R}^d$ is well-described (with a parameter φ) if there exists an H-representation for P in which every inequality has encoding length at most φ. The encoding length of P is $\langle P \rangle = d + \varphi$.

Proposition (Grötschel et al.'93)

For a well-described polytope, we can compute OPT_P from SEP_P (and vice versa) in oracle polynomial-time. The runtime (polynomially) depends on d and φ.

The edge-skeleton algorithm



Input:

- OPT_P
- Edge vec. P (dir. & len.): D

Output:

- Edge-skeleton of P

Sketch of **Algorithm**:

- Compute a vertex of P ($x = \mathrm{OPT}_P(c)$ for arbitrary $c^T \in \mathbb{R}^d$)

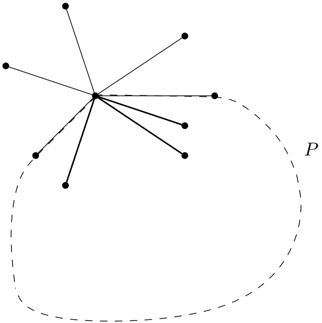
The edge-skeleton algorithm

Input:

- OPT_P
- Edge vec. P (dir. & len.): D

Output:

- Edge-skeleton of P



Sketch of **Algorithm**:

- Compute a vertex of P ($x = \mathrm{OPT}_P(c)$ for arbitrary $c^T \in \mathbb{R}^d$)
- Compute segments $S = \{(x, x+d)$, for all $d \in D\}$

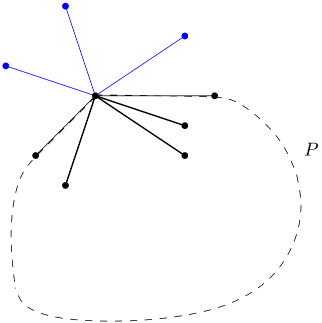
The edge-skeleton algorithm

Input:

- OPT_P
- Edge vec. P (dir. & len.): D

Output:

- Edge-skeleton of P



Sketch of **Algorithm**:

- Compute a vertex of P ($x = \mathrm{OPT}_P(c)$ for arbitrary $c^T \in \mathbb{R}^d$)
- Compute segments $S = \{(x, x + d)$, for all $d \in D\}$
- Remove from S all segments (x, y) s.t. $y \notin P$ ($\mathrm{OPT}_P \rightarrow \mathrm{SEP}_P$)

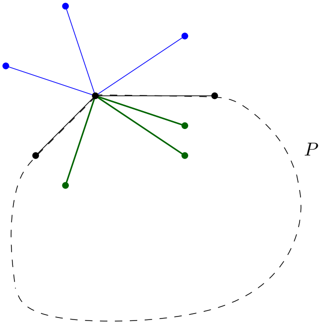
The edge-skeleton algorithm

Input:

- OPT_P
- Edge vec. P (dir. & len.): D

Output:

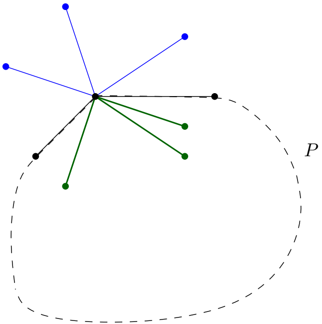
- Edge-skeleton of P



Sketch of **Algorithm**:

- Compute a vertex of P ($x = \mathrm{OPT}_P(c)$ for arbitrary $c^T \in \mathbb{R}^d$)
- Compute segments $S = \{(x, x + d)$, for all $d \in D\}$
- Remove from S all segments (x, y) s.t. $y \notin P$ ($\mathrm{OPT}_P \rightarrow \mathrm{SEP}_P$)
- Remove from S the segments that are not extreme

The edge-skeleton algorithm



Input:

- OPT_P
- Edge vec. P (dir. & len.): D

Output:

- Edge-skeleton of P

Sketch of **Algorithm**:

- Compute a vertex of P ($x = \mathrm{OPT}_P(c)$ for arbitrary $c^T \in \mathbb{R}^d$)
- Compute segments $S = \{(x, x + d)$, for all $d \in D\}$
- Remove from S all segments (x, y) s.t. $y \notin P$ ($\mathrm{OPT}_P \to \mathrm{SEP}_P$)
- Remove from S the segments that are not extreme

Can be altered to work with edge directions only

Complexity

Theorem

Given OPT_P *and a superset of edge directions* D *of a well-described polytope* $P \subseteq \mathbb{R}^d$, *the edge skeleton of* P *can be computed in oracle total polynomial-time*

$$O\left(n|D|\left(T + \mathbb{LP}(d^3|D|\langle B\rangle) + d\log n\right)\right),$$

- n *the number of vertices of* P,
- T : *runtime of oracle conversion algorithm for* P *and* D,
- $\langle B\rangle$ *is the binary encoding length of the vector set* P *and* D,
- $\mathbb{LP}(\langle A\rangle + \langle b\rangle + \langle c\rangle)$ *runtime of max* $c^T x$ *over* $\{x : Ax \le b\}$.

Applications

Corollary

The edge skeleton of resultant, secondary polytopes can be computed in oracle total polynomial-time.

Corollary

The edge skeletons of polytopes appearing in convex combinatorial optimization [Rothblum-Onn '04] and convex integer programming [De Loera et al. '09] problems can be computed in oracle total polynomial-time.

References

- Emiris, F, Gärtner Efficient Volume and Edge-Skeleton Computation for Polytopes Given by Oracles. Proc. of 29th European Workshop on Computational Geometry, Braunschweig, Germany 2013.

- Emiris, F, Gärtner Efficient edge skeleton computation for polytopes defined by oracles. Submitted to Computational Geometry - Theory and Applications.

Outline

The volume computation problem

Input: Polytope $P := \{x \in \mathbb{R}^d \mid Ax \leq b\}$ $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$

Output: Volume of P

- #-P hard for vertex and for halfspace repres. [DyerFrieze'88]

The volume computation problem

Input: Polytope $P := \{x \in \mathbb{R}^d \mid Ax \leq b\}$ $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$

Output: Volume of P

- #-P hard for vertex and for halfspace repres. [DyerFrieze'88]
- randomized poly-time algorithm approximates the volume of a convex body with high probability and arbitrarily small relative error [DyerFriezeKannan'91] $O^*(d^{23}) \rightarrow O^*(d^4)$ [LovVemp'04]

The volume computation problem

Input: Polytope $P := \{x \in \mathbb{R}^d \mid Ax \leq b\}$ $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$

Output: Volume of P

- #-P hard for vertex and for halfspace repres. [DyerFrieze'88]
- randomized poly-time algorithm approximates the volume of a convex body with high probability and arbitrarily small relative error [DyerFriezeKannan'91] $O^*(d^{23}) \to O^*(d^4)$ [LovVemp'04]

Implementations

- Exact (`VINCI`, `Qhull`, etc.) cannot compute in high dimensions (e.g. > 20)
- Randomized ([CousinsVempala'14], [EmirisF'14]) compute in high dimensions (e.g. 100)

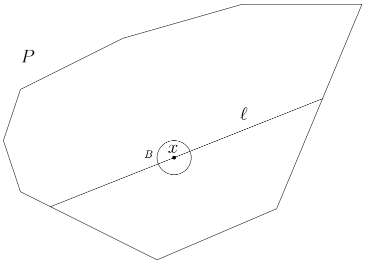
How do we compute a random point in a polytope P?

- easy for simple shapes like simplex or cube

How do we compute a random point in a polytope P?

- easy for simple shapes like simplex or cube
- BUT for arbitrary polytopes we need *random walks*

Random Directions Hit-and-Run (RDHR)



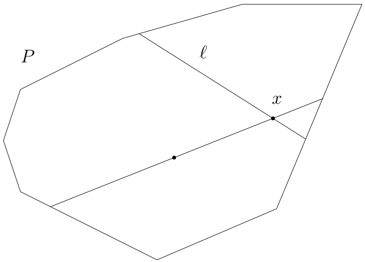
Input: point $x \in P$

Output: new point $x' \in P$

1. line ℓ through x, uniform on $B(x, 1)$
2. set x' to be a uniform disrtibuted point on $P \cap \ell$

Iterate this for W steps.

Random Directions Hit-and-Run (RDHR)



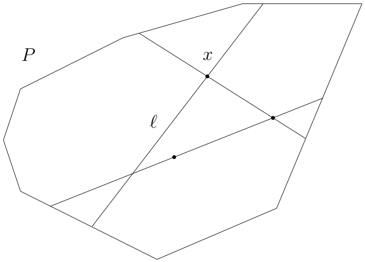
Input: point $x \in P$

Output: new point $x' \in P$

1. line ℓ through x, uniform on $B(x, 1)$
2. set x' to be a uniform disrtibuted point on $P \cap \ell$

Iterate this for W steps.

Random Directions Hit-and-Run (RDHR)



Input: point $x \in P$

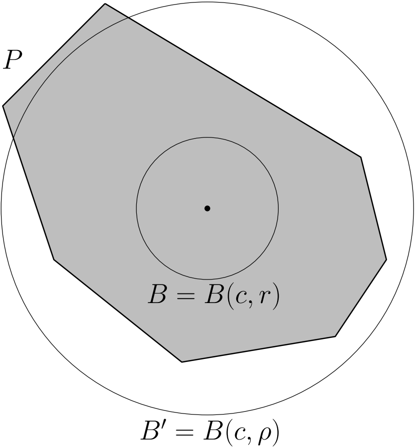
Output: new point $x' \in P$

1. line ℓ through x, uniform on $B(x, 1)$
2. set x' to be a uniform disrtibuted point on $P \cap \ell$

Iterate this for W steps.

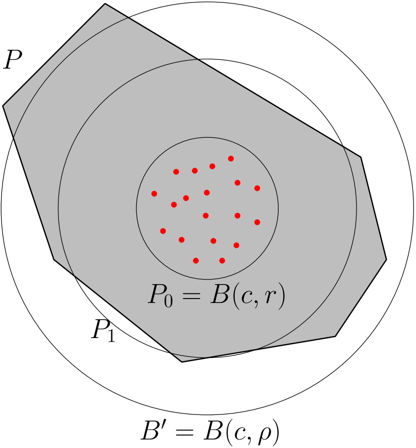
- x' is unif. random distrib. in P after $W = O^*(d^3)$ steps, where $O^*(\cdot)$ hides log factors [LovaszVempala'06]
- to generate many random points iterate this procedure

Multiphase Monte Carlo (Sequence of balls)



- $B(c, 2^{i/d}),\ i = \alpha, \alpha+1, \ldots, \beta,$
  $\alpha = \lfloor d \log r \rfloor,\ \beta = \lceil d \log \rho \rceil$
- $P_i := P \cap B(c, 2^{i/d}),\ i = \alpha, \alpha+1, \ldots, \beta,$
  $P_\alpha = B(c, 2^{\alpha/d}) \subseteq B(c, r)$

Multiphase Monte Carlo (Generate/count random points)



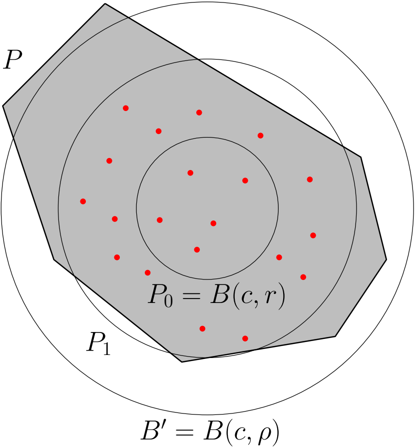
- $B(c, 2^{i/d})$, $i = \alpha, \alpha+1, \ldots, \beta$,
  $\alpha = \lfloor d \log r \rfloor$, $\beta = \lceil d \log \rho \rceil$

- $P_i := P \cap B(c, 2^{i/d})$, $i = \alpha, \alpha+1, \ldots, \beta$,
  $P_\alpha = B(c, 2^{\alpha/d}) \subseteq B(c, r)$

1. Generate rand. points in P_i
2. Count how many rand. points in P_i fall in P_{i-1}

$$\mathrm{vol}(P) = \mathrm{vol}(P_\alpha) \prod_{i=\alpha+1}^{\beta} \frac{\mathrm{vol}(P_i)}{\mathrm{vol}(P_{i-1})}$$

Multiphase Monte Carlo (Generate/count random points)



- $B(c, 2^{i/d})$, $i = \alpha, \alpha + 1, \ldots, \beta$,
  $\alpha = \lfloor d \log r \rfloor$, $\beta = \lceil d \log \rho \rceil$
- $P_i := P \cap B(c, 2^{i/d})$, $i = \alpha, \alpha + 1, \ldots, \beta$,
  $P_\alpha = B(c, 2^{\alpha/d}) \subseteq B(c, r)$

1. Generate rand. points in P_i
2. Count how many rand. points in P_i fall in P_{i-1}

$$\mathrm{vol}(P) = \mathrm{vol}(P_\alpha) \prod_{i=\alpha+1}^{\beta} \frac{\mathrm{vol}(P_i)}{\mathrm{vol}(P_{i-1})}$$

Contributions

Some modifications towards practicality

- $W = \lfloor 10 + d/10 \rfloor$ random walk steps (vs. $O^*(d^3)$ which hides constant 10^{11}) achieve $< 1\%$ error in up to 100 dim.
- implement boundary oracles with $O(m)$ runtime in coordinate (vs. random) directions hit-and-run

Contributions

Some modifications towards practicality

- $W = \lfloor 10 + d/10 \rfloor$ random walk steps (vs. $O^*(d^3)$ which hides constant 10^{11}) achieve $< 1\%$ error in up to 100 dim.
- implement boundary oracles with $O(m)$ runtime in coordinate (vs. random) directions hit-and-run

Highlights of experimental results

- approximate the volume of a series of polytopes (cubes, random, cross, Birkhoff) for $d < 100$ in <2 hrs with mean approximation error <1%
- Compute the volume of Birkhoff polytopes $B_{11}, \ldots, B_{15}$ in few hrs whereas exact methods have only computed that of B_{10} by specialized software in ~ 1 year of parallel computation

References

- Emiris, F. Efficient random-walk methods for approximating polytope volume. Proc. of 30th ACM Annual Symposium on Computational Geometry, 2014, Kyoto, Japan.

Outline

Introduction

An algorithm for computing projections of resultant polytopes

Edge-skeleton computation for polytopes defined by oracles

A practical volume algorithm for high dimensional polytopes

Combinatorics of 4-d resultant polytopes

High-dimensional predicates: algorithms and software

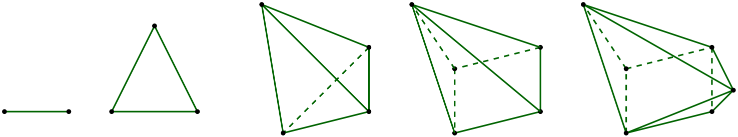
Existing work

- [GKZ'90] Univariate case / general dimensional $N(R)$

Existing work

- [GKZ'90] Univariate case / general dimensional $N(R)$

- [Sturmfels'94] Multivariate case / up to 3 dimensional $N(R)$



One step beyond... 4-dimensional N(R)

- Polytope $P \subseteq \mathbb{R}^4$; f-vector is the vector of its face cardinalities.

One step beyond... 4-dimensional N(R)

- Polytope $P \subseteq \mathbb{R}^4$; f-vector is the vector of its face cardinalities.
- Call vertices, edges, ridges, facets, the $0,1,2,3$-d, faces of P.

One step beyond... 4-dimensional N(R)

- Polytope $P \subseteq \mathbb{R}^4$; f-vector is the vector of its face cardinalities.
- Call vertices, edges, ridges, facets, the $0,1,2,3\text{-}d$, faces of P.
- f-vectors of 4-dimensional N(R) (computed with `ResPol`)

(5, 10, 10, 5)
(6, 15, 18, 9)
(8, 20, 21, 9)
(9, 22, 21, 8)
.
.
.
(17, 49, 48, 16)
(17, 49, 49, 17)
(17, 50, 50, 17)
(18, 51, 48, 15)
(18, 51, 49, 16)
(18, 52, 50, 16)
(18, 52, 51, 17)
(18, 53, 51, 16)
(18, 53, 53, 18)
(18, 54, 54, 18)
(19, 54, 52, 17)
(19, 55, 51, 15)
(19, 55, 52, 16)
(19, 55, 54, 18)
(19, 56, 54, 17)
(19, 56, 56, 19)
(19, 57, 57, 19)
(20, 58, 54, 16)
(20, 59, 57, 18)
(20, 60, 60, 20)
(21, 62, 60, 19)
(21, 63, 63, 21)
(22, 66, 66, 22)

Main result

Theorem

Given $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$ with $N(R)$ of dimension 4. Then $N(R)$ are degenerations of the polytopes in following cases.

- Degenarations can only decrease the number of faces.

Main result

Theorem

Given $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$ with $N(R)$ of dimension 4. Then $N(R)$ are degenerations of the polytopes in following cases.

(i) *All $|A_i| = 2$, except for one with cardinality 5, is a 4-simplex with f-vector $(5, 10, 10, 5)$.*

- Degenarations can only decrease the number of faces.

Main result

Theorem

Given $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$ *with* $N(R)$ *of dimension* 4*. Then* $N(R)$ *are degenerations of the polytopes in following cases.*

(i) *All* $|A_i| = 2$*, except for one with cardinality* 5*, is a* 4*-simplex with* f*-vector* $(5, 10, 10, 5)$*.*

(ii) *All* $|A_i| = 2$*, except for two with cardinalities* 3 *and* 4*, has* f*-vector* $(10, 26, 25, 9)$*.*

- Degenarations can only decrease the number of faces.

Main result

Theorem

Given $A_0, A_1, \ldots, A_n \subset \mathbb{Z}^n$ *with* $N(R)$ *of dimension* 4. *Then* $N(R)$ *are degenerations of the polytopes in following cases.*

(i) *All* $|A_i| = 2$, *except for one with cardinality* 5, *is a* 4*-simplex with* f*-vector* $(5, 10, 10, 5)$.

(ii) *All* $|A_i| = 2$, *except for two with cardinalities* 3 *and* 4, *has* f*-vector* $(10, 26, 25, 9)$.

(iii) *All* $|A_i| = 2$, *except for three with cardinality* 3, *maximal number of ridges is* $\tilde{f}_2 = 66$ *and of facets* $\tilde{f}_3 = 22$. *Moreover,* $22 \leq \tilde{f}_0 \leq 28$, *and* $66 \leq \tilde{f}_1 \leq 72$. *The lower bounds are tight.*

- Degenarations can only decrease the number of faces.
- Focus on new case (iii), which reduces to $n = 2$ and each $|A_i| = 3$.

Main result

Theorem

Given $A_0, A_1, \dots, A_n \subset \mathbb{Z}^n$ with $N(R)$ of dimension 4. Then $N(R)$ are degenerations of the polytopes in following cases.

(i) *All $|A_i| = 2$, except for one with cardinality 5, is a 4-simplex with f-vector $(5, 10, 10, 5)$.*

(ii) *All $|A_i| = 2$, except for two with cardinalities 3 and 4, has f-vector $(10, 26, 25, 9)$.*

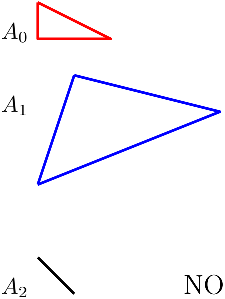
(iii) *All $|A_i| = 2$, except for three with cardinality 3, maximal number of ridges is $\tilde{f}_2 = 66$ and of facets $\tilde{f}_3 = 22$. Moreover, $22 \leq \tilde{f}_0 \leq 28$, and $66 \leq \tilde{f}_1 \leq 72$. The lower bounds are tight.*

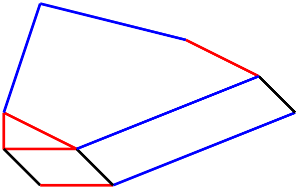
- Degenarations can only decrease the number of faces.
- Focus on new case (iii), which reduces to $n = 2$ and each $|A_i| = 3$.
- Generic upper bound for vertices yields 6608 [Sturmfels'94].

Tool (1): N(R) faces and subdivisions

A subdivision S of $A_0 + A_1 + \cdots + A_n$ is mixed when its cells are Minkowski sums of A_i's subsets.

Example





NOT fine mixed subdivision S of $A_0 + A_1 + A_2$

Tool (1): $N(R)$ faces and subdivisions

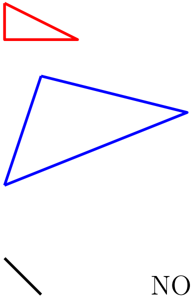
A subdivision S of $A_0 + A_1 + \cdots + A_n$ is mixed when its cells are Minkowski sums of A_i's subsets.

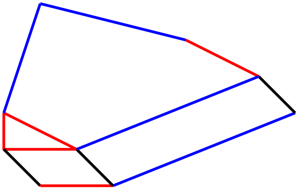
Example

A_0

A_1

A_2





NOT fine mixed subdivision S of $A_0 + A_1 + A_2$

Proposition (Sturmfels'94)

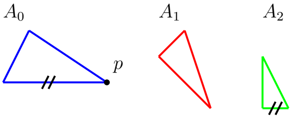
A regular mixed subdivision S *of* $A_0 + A_1 + \cdots + A_n$ *corresponds to a face of* $N(R)$.

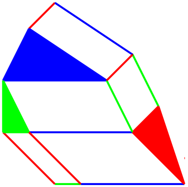
Tool (2): Input genericity

Proposition

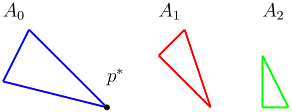
Input genericity maximizes the number of resultant polytope faces.

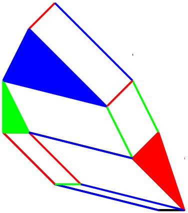
Proof idea





$N(R)$ f-vector: $(14, 38, 36, 12)$





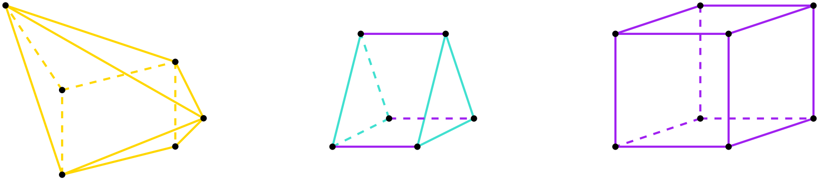
$N(R^*)$ f-vector: $(18, 52, 50, 16)$

Facets of 4-d resultant polytopes

Lemma

A 4-dimensioanl $\mathrm{N}(\mathrm{R})$ *have at most*

- *9 resultant facets: 3-d N(R)*
- *9 prism facets: 2-d N(R) (triangle)* $+$ *1-d N(R)*
- *4 zonotope facets: Mink. sum of 1-d N(R)s*



References

- Dickenstein, Emiris, F. Combinatorics of 4-dimensional Resultant Polytopes. Proc. of the 38th ACM Symposium on Symbolic and Algebraic Computation, 2013, Boston, MA, USA.

Outline

Geometric algorithms and predicates

Setting

- geometric algorithms $\rightarrow$ sequence of geometric predicates
- Hi-dim: as dimension grows predicates become more expensive

Geometric algorithms and predicates

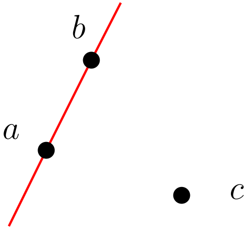
Setting

- geometric algorithms $\longrightarrow$ sequence of geometric predicates
- Hi-dim: as dimension grows predicates become more expensive

Examples

- Orientation: Does c lie on, left or right of ab?

$$\begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} \gtreqless 0$$



Determinant computation

Given matrix $A \subseteq \mathbb{R}^{d \times d}$

- Theory: State-of-the-art $O(d^{\omega})$, $\omega \sim 2.3727$ [Williams'12]
- Practice: Gaussian elimination, $O(d^3)$

Dynamic Determinant Computations

One-column update problem

Given matrix $A \subseteq \mathbb{R}^{d \times d}$, answer queries for $det(A)$ when i-th column of A, $(A)_i$, is replaced by $u \subseteq \mathbb{R}^d$.

Dynamic Determinant Computations

One-column update problem

Given matrix $A \subseteq \mathbb{R}^{d\times d}$, answer queries for $\det(A)$ when i-th column of A, $(A)_i$, is replaced by $u \subseteq \mathbb{R}^d$.

Solution: Sherman-Morrison formula (1950)

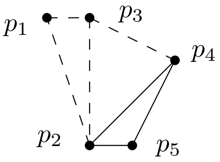
$$A'^{-1} = A^{-1} - \frac{(A^{-1}(u-(A)_i))\ (e_i^T A^{-1})}{1 + e_i^T A^{-1}(u-(A)_i)}$$

$$\det(A') = (1 + e_i^T A^{-1}(u-(A)_i)\det(A)$$

- Only vector×vector, vector×matrix → Complexity: $O(d^2)$

Incremental convex hull: REVISITED

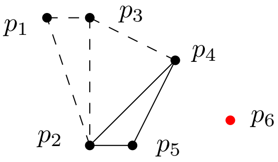
$$A = \begin{array}{|c|c|c|} \hline p_2 & p_4 & p_5 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



- $\mathrm{Orientation}(p_2, p_4, p_5) = \mathrm{sgn}(\det(A))$

Incremental convex hull: REVISITED

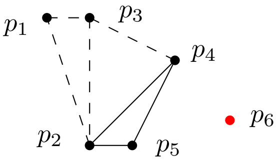
$$A' = \begin{array}{|c|c|c|} \hline p_6 & p_4 & p_5 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



- Orientation$(p_6, p_4, p_5) = \mathrm{sgn}(\det(A'))$ in $O(d^2)$

Incremental convex hull: REVISITED

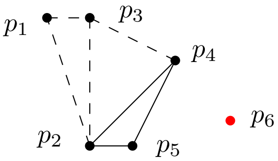
$$A' = \begin{array}{|c|c|c|}\hline p_6 & p_4 & p_5 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$



- Orientation$(p_6, p_4, p_5) = sgn(\det(A'))$ in $O(d^2)$
- Store $\det(A)$, A^{-1} in a hash table

Incremental convex hull: REVISITED

$$A' = \begin{array}{|c|c|c|} \hline p_6 & p_4 & p_5 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



- Orientation$(p_6, p_4, p_5) = sgn(\det(A'))$ in $O(d^2)$
- Store $\det(A)$, A^{-1} in a hash table
- Update $\det(A')$, A'^{-1} (Sherman-Morrison)

Experiments

Determinants (1-column updates)

- 2 and 7 times faster than state-of-the-art software (Eigen, Linbox, Maple) in rational and integer arithmetic resp.

Convex hull

- Plug into `triangulation/CGAL` improving performance
- Outperforms `polymake`, `lrs`, `cdd` in most cases with generic input in $d \leq 7$

Point location

- Improves up to 78 times in `triangulation/CGAL`, using up to 50 times more memory, $d \leq 11$

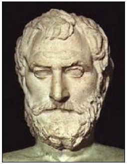
References

- F, Peñaranda. Faster Geometric Algorithms via Dynamic Determinant Computation. Proc. of European Symposium on Algorithms, LNCS, 2012, Ljubljana, Slovenia.

Acknowledgements

Funding

CGL



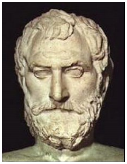
Co-authors





Acknowledgements

Funding



Co-authors





THANK YOU !!!