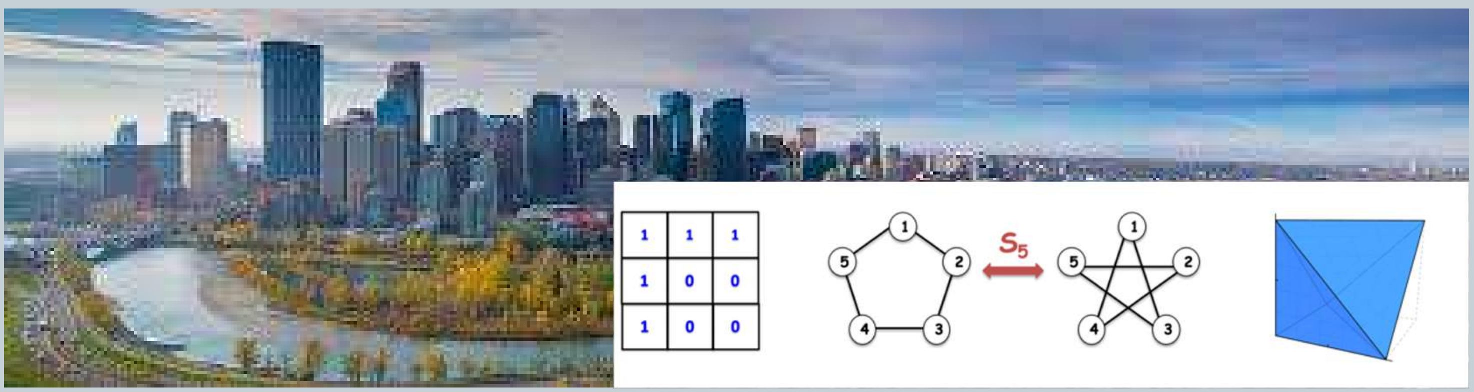
Panorama of scaling problems and algorithms

Ankit Garg
Microsoft Research India

FOCS 2018, October 6, 2018



Overview

- Sinkhorn initiated study of *matrix scaling* in 1964.

- *Numerous applications* in statistics, numerical computing, theoretical computer science and even Sudoku!

A RELATIONSHIP BETWEEN ARBITRARY POSITIVE MATRICES AND DOUBLY STOCHASTIC MATRICES

By Richard Sinkhorn

University of Houston

1. Introduction. Suppose one observes n transitions of a Markov chain with N states and stochastic matrix $P = (p_{ij})$. The usual estimate of p_{ij} is $t_{ij} = a_{ij}/\lambda_i$ where a_{ij} is the number of transitions from i to j which are observed, and $\lambda_i = \sum_j a_{ij}$. (Cf. [1].) This amounts to a normalization of the rows of $A = (a_{ij})$, and can be expressed as a matrix equation $T = D_1 A$ where $T = (t_{ij})$ and $D_1 = \text{diag}[\lambda_1^{-1}, \cdots, \lambda_N^{-1}]$.

If it is known that the stochastic matrix P is in fact doubly stochastic, (i.e., $\sum_i p_{ij} = 1$), what then is a good estimate of T? The maximum likelihood equations are difficult to solve. One estimate which has been used (for example, by Welch [4]) is to alternately normalize the rows and columns of A, in the belief that this iterative process converges to a doubly stochastic matrix, T, which might be, in some sense, a good estimate.

Sinkhorn Solves Sudoku

Todd K. Moon, *Senior Member, IEEE*, Jacob H. Gunther, *Member, IEEE*, and Joseph J. Kupin

***Abstract*—The Sudoku puzzle is a discrete constraint satisfaction problem, as is the error correction decoding problem. We propose here an algorithm for solution to the Sinkhorn puzzle based on Sinkhorn balancing. Sinkhorn balancing is an algorithm for projecting a matrix onto the space of doubly stochastic matrices. The Sinkhorn balancing solver is capable of solving all but the most difficult puzzles. A proof of convergence is presented, with some information theoretic connections. A random generalization of the Sudoku puzzle is presented, for which the Sinkhorn-based solver is also very effective.**

***Index Terms*—Belief propagation (BP), constraint satisfaction, low-density parity-check (LDPC) decoding, Sinkhorn, Sudoku.**

(sometimes called Sinkhorn scaling) has been widely studiec and makes its appearance in a variety of applications. (See, fo example [6].) The Sinkhorn balancing approach to solution i successful at solving all but the most difficult Sudoku puzzles Sinkhorn balancing furthermore generalizes well to situation in which clues are presented as random elements in a set.

As there are other methods of solving Sudoku puzzles, th method presented here needs some justification. Our exploratio was motivated by a desire to develop decoding algorithms fo linear codes having many cycles in their Tanner graphs. Whil the BP algorithm fares poorly for such codes (and Sudoku puz

Overview

- *Generalized* in several unexpected directions with multiple themes.

1. *Analytic* approaches for *algebraic* problems.

- Special cases of polynomial identity testing (*PIT*).
- *Isomorphism* related problems: Null cone, orbit intersection, orbit-closure intersection.

2. Provable fast convergence of *alternating minimization* algorithms in problems with *symmetries*.
3. *Tractable polytopes* with exponentially many vertices and facets. *Brascamp-Lieb* polytopes, *moment* polytopes etc.

Outline

- Matrix scaling
- Operator scaling
- *Unified source* of scaling problems
- *Even more* scaling problems

Matrix scaling: Sinkhorn's algorithm, analysis and an application

Matrix Scaling

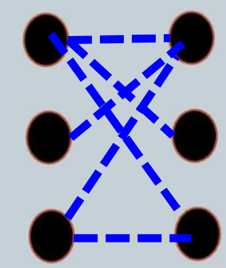
- Non-negative $n \times n$ matrix A.
- Scaling: B is a scaling of A if $B = RAC$. R and C are positive diagonal matrices.

$$B_{i,j} = R_{i,i} \cdot C_{j,j} \cdot A_{i,j}$$

- Doubly stochastic: B is doubly stochastic if all row and column sums are 1.
- [Sinkhorn 64]: If $A_{i,j} > 0$ for all i, j, then a doubly stochastic scaling of A exists.
- Proved that a natural iterative algorithm converges.
- [Sinkhorn, Knopp 67]: Iterative algorithm converges iff $\mathrm{supp}(A)$ admits a *perfect matching*.

Matrix scaling: Example 1

- [Sinkhorn 64]: Alternately normalize rows and columns.

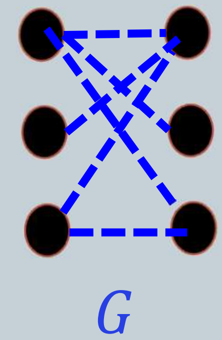


G

1	**1**	**1**
1	**0**	**0**
1	**0**	**1**

Matrix scaling: Example 1

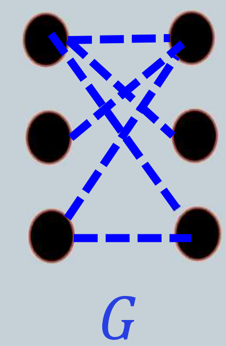
- [Sinkhorn 64]: Alternately normalize rows and columns.



1/3	1/3	1/3
1	0	0
1/2	0	1/2

Matrix scaling: Example 1

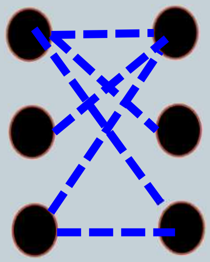
- [Sinkhorn 64]: Alternately normalize rows and columns.



2/11	**1**	**2/5**
6/11	**0**	**0**
3/11	**0**	**3/5**

Matrix scaling: Example 1

- [Sinkhorn 64]: Alternately normalize rows and columns.

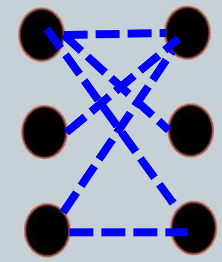


G

10/87	**55/87**	**22/87**
1	**0**	**0**
5/16	**0**	**11/16**

Matrix scaling: Example 1

- [Sinkhorn 64]: Alternately normalize rows and columns.

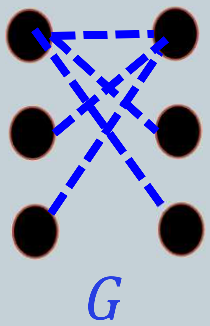


G

0	1	0
1	0	0
0	0	1

Matrix scaling: Example 2

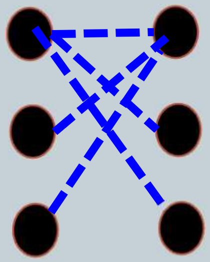
- [Sinkhorn 64]: Alternately normalize rows and columns.



1	**1**	**1**
1	**0**	**0**
1	**0**	**0**

Matrix scaling: Example 2

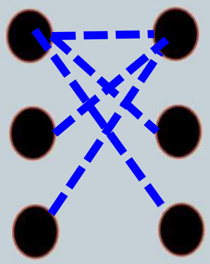
- [Sinkhorn 64]: Alternately normalize rows and columns.



1/3	1/3	1/3
1	0	0
1	0	0

Matrix scaling: Example 2

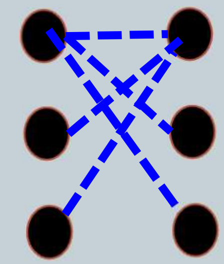
- [Sinkhorn 64]: Alternately normalize rows and columns.



1/7	**1**	**1**
3/7	**0**	**0**
3/7	**0**	**0**

Matrix scaling: Example 2

- [Sinkhorn 64]: Alternately normalize rows and columns.

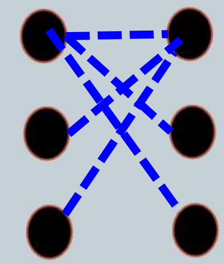


G

1/15	**7/15**	**7/15**
1	**0**	**0**
1	**0**	**0**

Matrix scaling: Example 2

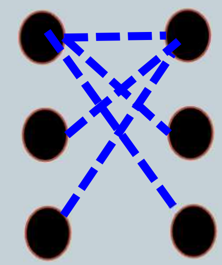
- [Sinkhorn 64]: Alternately normalize rows and columns.



0	1	1
1/2	0	0
1/2	0	0

Matrix scaling: Example 2

- [Sinkhorn 64]: Alternately normalize rows and columns.

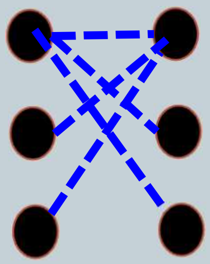


G

0	1/2	1/2
1	0	0
1	0	0

Matrix scaling: Example 2

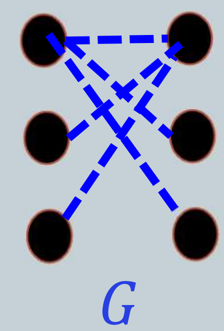
- [Sinkhorn 64]: Alternately normalize rows and columns.



0	1	1
1/2	0	0
1/2	0	0

Matrix scaling: Example 2

- [Sinkhorn 64]: Alternately normalize rows and columns.



0	1/2	1/2
1	0	0
1	0	0

Analysis

Algorithm S

- Input: A
- Repeat for N steps:
1. Normalize rows;
2. Normalize columns;
- Output: $\hat{A}$

- Theorem [Linial, Samorodnitsky, Wigderson 00]: With $N = O\left(\frac{n(b+\log(n))}{\epsilon}\right)$, $\hat{A}$ "ϵ-close to being DS" (if scalable).
- Initial A integer entries with bit complexity b.
- $r_1, \dots, r_n, c_1, \dots, c_n$ row and column sums of $\hat{A}$.
- $ds(\hat{A}) = \sum_i (r_i - 1)^2 + \sum_j (c_j - 1)^2 \leq \epsilon$

Analysis

- Need a potential function.
- [Sinkhorn, Knopp 67]: A scalable iff $\mathrm{supp}(A)$ admits a *perfect matching*.
- Potential function: $\mathrm{perm}(A) = \sum_{\sigma \in S_n} \prod_i A_{i,\sigma(i)}$.
- A scalable and integer entries $\Rightarrow$ $\mathrm{perm}(A) \geq 1$.
- After first normalization $A \to A'$, $\mathrm{perm}(A') \geq 2^{-n(b+\log(n))}$.

3-step analysis

Analysis

- [Lower bound]: Initially $\text{perm} \geq 2^{-n(b+\log(n))}$.
- [Progress per step]: If ϵ-far from DS, normalization increases perm by a factor of $\exp(\epsilon/6)$. Consequence of a robust *AM-GM* inequality.
- [Upper bound]: If row or column normalized, $\text{perm} \leq 1$.

- Therefore get ϵ-close to DS in $O\left(\frac{n(b+\log(n))}{\epsilon}\right)$ steps.
- Crucial property of permanent:

$$\text{perm}(R\ A\ C) = \prod_i R_{i,i} \prod_j C_{j,j}\ \text{perm}(A)$$

- (R, C diagonal). Permanent *invariant* under action of diagonal matrices (with determinant 1).

Another potential function: capacity

- [Gurvits, Yianilos 98] provided an alternate analysis of Sinkhorn's algorithm using the notion of capacity.

$$\mathrm{cap}(A) = \inf\left\{\prod_i (Ax)_i : \prod_i x_i = 1, x > 0\right\}$$

- Matrix scaling is equivalent to solving this optimization problem.

Application: Bipartite matching

- [Sinkhorn, Knopp 67]: Iterative algorithm converges iff $\mathrm{supp}(A)$ admits a *perfect matching*.
- [Linial, Samorodnitsky, Wigderson 00]: Only need to check $1/n$ close to DS.

Algorithm

- Input A_G
- Repeat for $O(n^2 \log(n))$ steps:

1. Normalize rows;
2. Normalize columns;

- Output $\hat{A}$
- Test if $ds(\hat{A}) < 1/n$,

  Yes: PM in G.

  No: No PM in G.

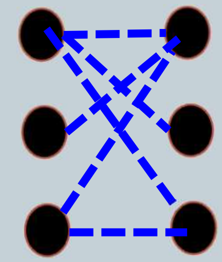
Another algorithm: Matching

1	1	1
1	0	0
1	0	1

A_G

x_{11}	x_{12}	x_{13}
x_{21}	0	0
x_{31}	0	x_{33}

$A_G(X)$



G

- G has a perfect matching iff $\mathrm{Det}(A_G(X)) \neq 0$.
- Plug in random values and check non-zeroness.
- Fast parallel algorithm.
- The algorithm generalizes to a "much harder" problem.

Edmonds' problem [1967]

- $L(X)$: entries linear forms in $X = \{x_1, \ldots, x_m\}$.
- Edmonds' problem: Test if $\mathrm{Det}(L(X)) \neq 0$.
- [Valiant 79]: Captures *PIT*.
- Easy randomized algorithm.
- Deterministic algorithm major open challenge.
- Is there a *scaling approach* to Edmonds' problem?
- Gurvits went on this quest.

L_{11}	L_{12}	L_{13}
L_{21}	L_{22}	L_{23}
L_{31}	L_{32}	L_{33}

$L(X)$

Operator scaling: Gurvits' algorithm and an application

Operator scaling

- Input: $A_1, \ldots, A_m$ $n \times n$ complex matrices.
- Same type as input for Edmonds' problem.
- $L(X)$: entries linear forms in $X = \{x_1, \ldots, x_m\}$. $L(X) = \sum_i x_i A_i$.
- Definition [Gurvits 04]: Call $A_1, \ldots, A_m$ *doubly stochastic* if

$$\sum_i A_i A_i^\dagger = I \text{ and } \sum_i A_i^\dagger A_i = I.$$

- A generalization of doubly stochastic matrices.
- $n \times n$ non-negative matrix $M \to n^2$ matrices, $A_{k,\ell} = \sqrt{M_{k,\ell}}\, E_{k,\ell}$.
- Natural from the point of quantum operators $T_A: P \to \sum_i A_i P A_i^\dagger$.
- Definition [Gurvits 04]: $A_1', \ldots, A_m'$ is a scaling of $A_1, \ldots, A_m$ if there exist invertible matrices B, C s.t. $A_1', \ldots, A_m' = BA_1C, \ldots, BA_mC$.
- Simultaneous basis change.

Operator

- Question [Gurvits 04]: When c
  doubly stochastic?
- Does it solve Edmonds' problem
- Gurvits designed a scaling algorithm.
- Proved it converges in poly time in special cases.
- Solves *special cases* of the Edmonds' problem, e.g. all A_i's rank 1.
- [G, Gurvits, Oliveira, Wigderson 16]: Proved Gurvits' algorithm converges in poly time, in general.
- Solves a *close cousin* of the Edmonds' problem (*non-commutative* version).

Gurvits' algorithm

- Goal: Transform $A_1, \ldots, A_m$ to satisfy

$$\sum_i A_i A_i^T = I \text{ and } \sum_i A_i^T A_i = I.$$

- Left normalize: $A_1, \ldots, A_m \rightarrow \left(\sum_i A_i A_i^T\right)^{-1/2} A_1, \ldots, \left(\sum_i A_i A_i^T\right)^{-1/2} A_m$.
- Ensures $\sum_i A_i A_i^T = I$.
- Right normalize: $A_1, \ldots, A_m \rightarrow A_1 \left(\sum_i A_i^T A_i\right)^{-1/2}, \ldots, A_m \left(\sum_i A_i^T A_i\right)^{-1/2}$.
- Ensures $\sum_i A_i^T A_i = I$.

Algorithm G

- Input: $A_1, \ldots, A_m$
- Repeat for N steps:

1. Left normalize;
2. Right normalize;

- Output: $A_1', \ldots, A_m'$

Gurvits' algorithm

- Theorem [G, Gurvits, Oliveira, Wigderson 16]: With $N = O\left(\frac{n(b+\log(n))}{\epsilon}\right)$, $A_1', \ldots, A_m'$ "ϵ-close to being DS" (if scalable).
- b: bit complexity of input.
- Analysis in *Rafael's next talk*.

Non-commutative singularity

- Symbolic matrices: $L = \sum_{i=1}^{m} x_i A_i$.
- $A_1, \dots, A_m$ are $n \times n$ complex matrices.
- Edmonds' problem: Test if $\mathrm{Det}\big(L(X)\big) \neq 0$.
- Or is $L(X)$ non-singular?
- Implicitly assume x_i's *commute*.
- NC-SING: $L(X)$ non-singular when x_i's non-commuting?
- Highly non-trivial to define.
- Work by Cohn and others in 70's.

$$\begin{bmatrix} \boldsymbol{L_{11}} & \boldsymbol{L_{12}} & \boldsymbol{L_{13}} \\ \boldsymbol{L_{21}} & \boldsymbol{L_{22}} & \boldsymbol{L_{23}} \\ \boldsymbol{L_{31}} & \boldsymbol{L_{32}} & \boldsymbol{L_{33}} \end{bmatrix}$$

$\boldsymbol{L}(\boldsymbol{X})$

Non-commutative singularity

- Easiest definition: $L = \sum_{i=1}^{m} x_i A_i$ NC-SING if

$$\mathrm{Det}(\sum_{i=1}^{m} X_i \otimes A_i) = 0,$$

for all d, X_i are $d \times d$ generic matrices (entries distinct formal commutative variables).

- Theorem [G, Gurvits, Oliveira, Wigderson 16]: *Deterministic poly* time algorithm for NC-SING.
- [Ivanyos, Qiao, Subrahmanyam 16; Derksen, Makam 16]: Algebraic algorithms. Work over other fields.
- Strongest PIT result in non-commutative algebraic complexity.

Analysis for algebra: source of scaling

Linear actions of groups

- Group G acts *linearly* on vector space V.
- $\pi: G \to GL(V)$ group homomorphism.
- $\pi(g): V \to V$ invertible linear map $\forall\, g \in G$.
- $\pi(g_1 g_2) = \pi(g_1) \circ \pi(g_2)$ and $\pi(id) = id$.

Example 1

- $G = S_n$ acts on $V = C^n$ by *permuting coordinates*.

$$\sigma \cdot (x_1, \dots, x_n) \to \left(x_{\sigma(1)}, \dots, x_{\sigma(n)}\right).$$

Example 2

- $G = GL_n(C)$ acts on $V = M_n(C)$ by *conjugation*.

$$A \cdot X = AXA^{-1}.$$

Orbits and orbit-closures

- Group G acts *linearly* on vector space V.

Objects of study

- Orbits: Orbit of vector v, $O_v = \{g \cdot v : g \in G\}$.
- Orbit-closures: Orbits may not be closed. Take their closures. Orbit-closure of vector v, $\overline{O_v} = \mathrm{cl}\,\{g \cdot v : g \in G\}$.

Example 1

- $G = S_n$ acts on $V = C^n$ by permuting coordinates.

$$\sigma \cdot (x_1, \dots, x_n) \to \left(x_{\sigma(1)}, \dots, x_{\sigma(n)}\right).$$

- x, y in same orbit iff they are of *same type*. $\forall c \in C$, $|\{i: x_i = c\}| = |\{i: y_i = c\}|$.
- Orbit-closures same as orbits.

Orbits and orbit-closures

Example 2

- $G = GL_n(C)$ acts on $V = M_n(C)$ by conjugation.

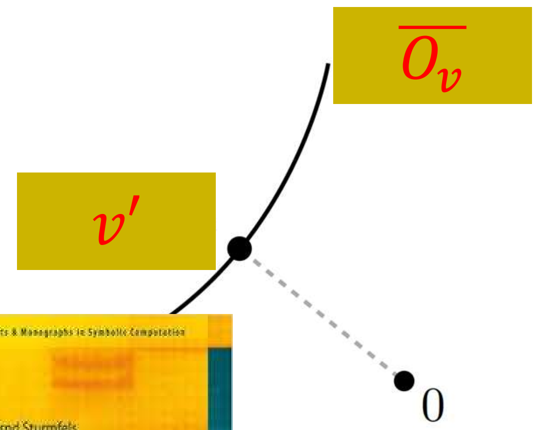
$$A \cdot X = AXA^{-1}.$$

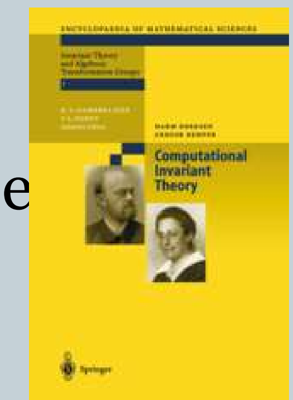
- Orbit of X: Y with same *Jordan normal form* as X.
- If X *not diagonalizable*, orbit and orbit-closure differ.
- Orbit-closures of X and Y intersect iff *same eigenvalues*.

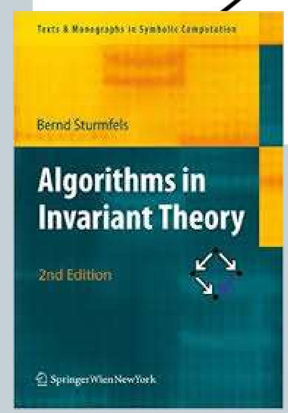
- Capture several interesting problems in theoretical computer science.
- *Graph isomorphism*: Whether orbits of two graphs the same. Group action: permuting the vertices.
- *Arithmetic circuits*: The VP vs VNP question. Whether permanent lies in the orbit-closure of the determinant. Group action: Action of $GL_{n^2}(C)$ on polynomials induced by action on variables.
- *Tensor rank*: Whether a tensor lies in the orbit-closure of the diagonal unit tensor. Group action: Natural action of $GL_n(C) \times GL_n(C) \times GL_n(C)$.

Connection to scaling

- Scaling: finding *minimal norm* elements in orbit-closures!
- Group G acts *linearly* on vector space V.
- $NC(v) = \inf_{g \in G} \|g \cdot v\|_2^2$.
- Null cone: v s.t. $NC(v) = 0$, i.e
- Determines *scalability*.
- v scalable iff *not in null cone*.
- Null cone membership fundamental problem in *invariant theory*.
- Scaling: natural analytic approach.







Example 1: Matrix scaling

- Given non-negative $n \times n$ matrix A, find non-negative diagonal matrices R, C s.t. RAC *doubly stochastic*.
- What is the group action?
- Defined by the problem itself!

Vector space	$n \times n$ *complex* matrices. (Minor translation: $M \in V \to A : A_{i,j} = \left\|M_{i,j}\right\|^2$.)
Group action	Left-right multiplication by diagonal matrices.
Annoying technicality	Need determinant 1 constraint.
Why doubly stochastic?	Critical point (KKT) condition.
Optimization problem	Gurvits' *capacity* for matrices.
Null cone	Bipartite matching.

Example 2: Operator scaling

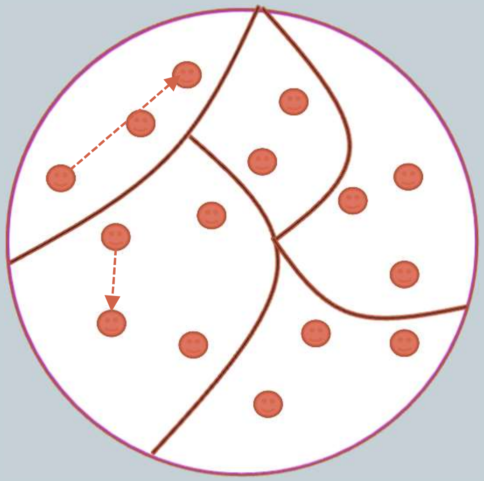
Vector space	Tuple of $n \times n$ complex matrices.
Group action	Simultaneous left-right multiplication.
Annoying technicality	Need determinant 1 constraint.
Why doubly stochastic?	Critical point (KKT) condition.
Optimization problem	Gurvits' *capacity* for operators.
Null cone	Non-commutative singularity.

Example 3: Geometric programming

Vector space	Polynomials in n variables $x_1, \dots, x_n$.
Group action	*Scaling* of variables. $x_i \to \alpha_i x_i$.
Annoying technicality	Need Laurent polynomials. Polynomials in $x_1, \dots, x_n$, $x_1^{-1}, \dots, x_n^{-1}$. Or determinant 1 constraint.
Optimization problem	Unconstrained Geometric programming. Or Gurvits' *capacity* for polynomials.
Null cone	Linear programming.

Significance for isomorphism problems

- Group G acts *linearly* on vector space V.
- $G = GL_n$ for simplicity.
- Natural *equivalence relation*: $v_1 \sim v_2$ if orbit-closures intersect.
- Strategy for testing equivalence: find *canonical* elements and test if equal.
- Fundamental theorems in invariant theory: *minimal norm* elements canonical (*up to unitary* action).

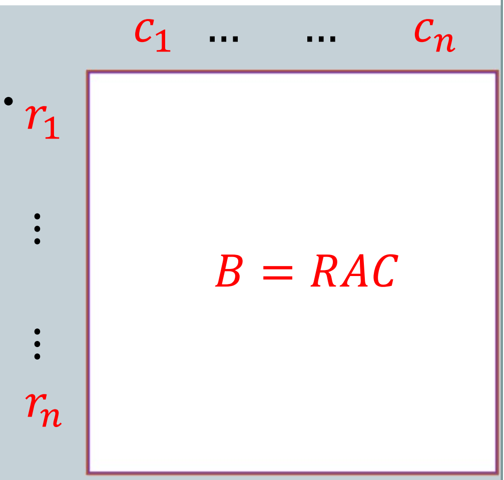


- Reduce problem to simpler unitary subgroup.
- Useful for orbit problems? When orbits closed – random orbits?

More scaling problems: interesting polytopes

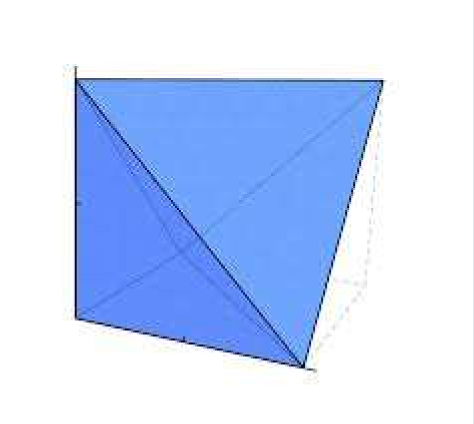
Non-uniform matrix scaling

- (r, c): probability distributions over $\{1, \ldots, n\}$.
- Non-negative $n \times n$ matrix A.
- Scaling of A with *row sums* $r_1, \ldots, r_n$ and *column sums* $c_1, \ldots, c_n$?
- $P_A = \{\text{such } (r, c)\}$.
- [...; Rothblum, Schneider 89]: P_A *convex polytope*!
- $P_A = \{(r, c) : \exists\, Q, \mathrm{supp}(Q) \subseteq \mathrm{supp}(A), Q \text{ marginals } (r, c)\}$.
- *Commutative group* actions: *classical marginal* problems.
- Computing *maximum entropy* distributions: *Nisheeth*'s talk.



Quantum marginals

- *Pure* quantum state $|\psi\rangle_{S_1,\ldots,S_d}$ (d quantum systems).
- Characterize marginals $\rho_{S_1},\ldots,\rho_{S_d}$ (marginal states on systems)?
- Only the spectra matter (local rotations for free).
- Collection of such spectra *convex polytope*!
- Follows from theory of *moment polytopes*.
- See Michael and Matthias' talks.
- Efficient algorithms via *non-uniform tensor scaling*. Cole's talk at FOCS 2018 (Tuesday 16:20).
- Underlying group action: Products of GL's on *tensors*.
- Other interesting moment polytopes: Schur-Horn, Horn, Brascamp-Lieb polytopes.



Conclusion and open problems

- Scaling problems: *natural optimization* problems with *symmetries*.
- *Analytic* tools for *algebraic* problems.
- Waiting for killer apps.

- Polynomial time algorithms for

1. *Null cone* membership?
2. *Moment polytope* membership, separation and optimization?
3. *Orbit-closure* intersection?

Thank You