Mixed Integer Conic Optimization using Julia and JuMP

Juan Pablo Vielma

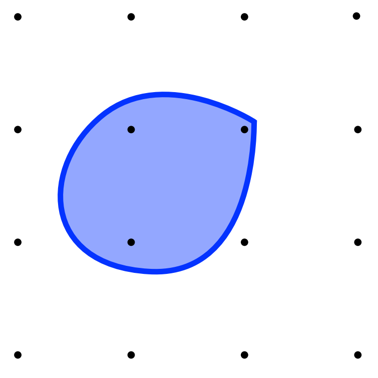
Massachusetts Institute of Technology

3rd Los Alamos National Laboratory Grid Science Winter School & Conference,

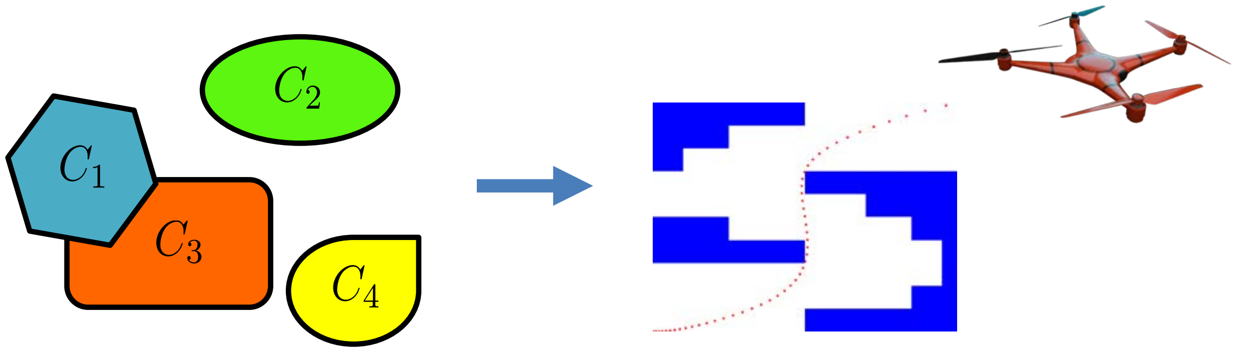
Santa Fe, NM, January, 2019.

Mixed Integer Convex Optimization (MICONV)

$$
\begin{aligned}
\min \quad & f(x) \\
s.t. \quad & \\
& x \in C \\
& x_i \in \mathbb{Z} \quad i \in I
\end{aligned}
$$



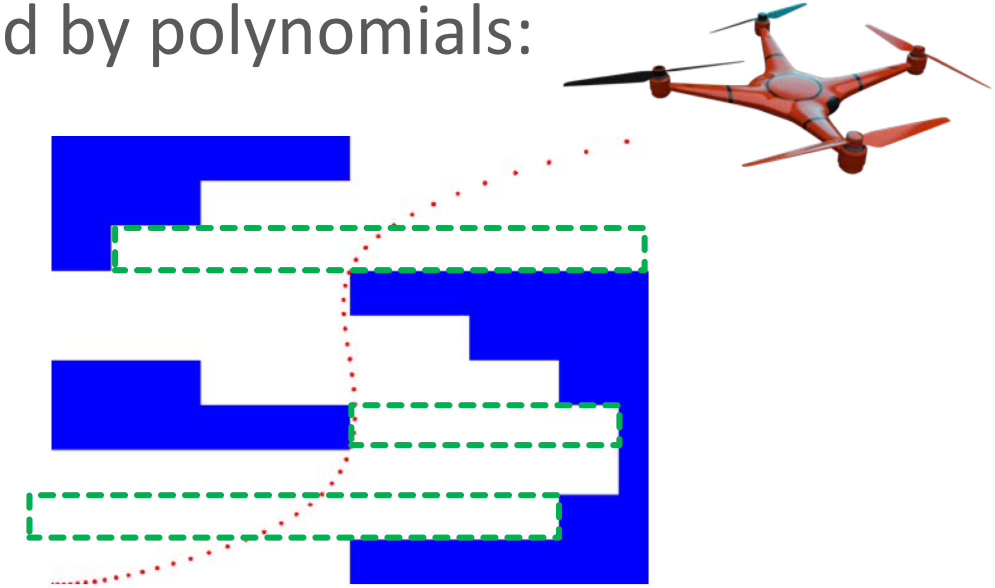
convex f and C.



http://www.gurobi.com/company/example-customers

An MICONV Example

- Position described by polynomials:



$(x(t), y(t))_{t\in[0,1]}$

- Step 1: discretize time into intervals $0 = T_1 < T_2 < \ldots < T_N = 1$

$$\{p_i : [T_i, T_{i+1}] \to \mathbb{R}^2\}_{i=1}^N \qquad (x(t), y(t)) = p_i(t) \quad t \in [T_i, T_{i+1}]$$

- Step 2: split domain into "safe polyhedrons" $P^r = \{x \in \mathbb{R}^2 \colon A^r x \le b^r\}$

$$\forall\, i \quad \exists\, r \quad s.t. \qquad p_i(t) \in P^r \qquad \forall t \in [T_i, T_{i+1}]$$

Disjunctive *Polynomial* Optimization

Variables = Polynomials : $\{p_i : [T_i, T_{i+1}] \to \mathbb{R}^2\}_{i=1}^{N}$

$$\min_{\boxed{p}} \quad \sum_{i=1}^{N} ||p_i'''(t)||^2$$

$$\text{s.t.} \quad p_1(0) = X_0,\ p'(0) = X_0',\ p''(0) = X_0''$$

$$p_N(1) = X_f,\ p_N'(1) = X_f',\ p_N''(1) = X_f''$$

Initial/Terminal Conditions

$$p_i(T_{i+1}) = p_{i+1}(T_{i+1}) \quad \forall i \in \{1, \ldots, N-1\}$$

$$p_i'(T_{i+1}) = p_{i+1}'(T_{i+1}) \quad \forall i \in \{1, \ldots, N-1\}$$

$$p_i''(T_{i+1}) = p_{i+1}''(T_{i+1}) \quad \forall i \in \{1, \ldots, N-1\}$$

Interstitial Smoothing Conditions

$$\bigvee_{r=1}^{R} [A^r p_i(t) \leq b^r] \ \text{ for } t \in [T_i, T_{i+1}] \quad \forall i \in \{1, \ldots, N-1\}$$

Remain in Safe Regions

Mixed-Integer ~~Disjunctive~~ *Polynomial* Optimization

Variables = Polynomials : $\{p_i : [T_i, T_{i+1}] \rightarrow \mathbb{R}^2\}_{i=1}^N$

$$\min_{\boxed{p}} \quad \sum_{i=1}^N ||p_i'''(t)||^2$$

s.t.

$$p_1(0) = X_0,\ p'(0) = X_0',\ p''(0) = X_0''$$
$$p_N(1) = X_f,\ p_N'(1) = X_f',\ p_N''(1) = X_f''$$

Initial/Terminal Conditions

$$p_i(T_{i+1}) = p_{i+1}(T_{i+1}) \quad \forall i \in \{1, \ldots, N-1\}$$
$$p_i'(T_{i+1}) = p_{i+1}'(T_{i+1}) \quad \forall i \in \{1, \ldots, N-1\}$$
$$p_i''(T_{i+1}) = p_{i+1}''(T_{i+1}) \quad \forall i \in \{1, \ldots, N-1\}$$

Interstitial Smoothing Conditions

$$b_j^r + M_j^r(1 - z_{i,r}) - A_j^r p_i(t) \geq 0 \qquad \text{for } t \in [T_i, T_{i+1}]$$

$$\sum_{r=1}^R z_{i,r} = 1 \qquad z \in \{0,1\}^{N \times R}$$

$\forall i \in\{1, \ldots,N\}$, $r\in\{1, \ldots,R\}$, $j\in\{1, 2\}$

Mixed-Integer ~~Disjunctive~~ ~~*Polynomial*~~ *Conic (SDP)* Optimization

Variables = Polynomials : $\{p_i : [T_i, T_{i+1}] \to \mathbb{R}^2\}_{i=1}^N$

$$\min_{\boxed{p}} \quad \sum_{i=1}^{N} ||p_i'''(t)||^2$$

s.t.

$$p_1(0) = X_0,\ p'(0) = X_0',\ p''(0) = X_0''$$
$$p_N(1) = X_f,\ p_N'(1) = X_f',\ p_N''(1) = X_f''$$

Initial/Terminal Conditions

$$p_i(T_{i+1}) = p_{i+1}(T_{i+1}) \quad \forall i \in \{1, \dots, N-1\}$$
$$p_i'(T_{i+1}) = p_{i+1}'(T_{i+1}) \quad \forall i \in \{1, \dots, N-1\}$$
$$p_i''(T_{i+1}) = p_{i+1}''(T_{i+1}) \quad \forall i \in \{1, \dots, N-1\}$$

Interstitial Smoothing Conditions

$$b_j^r + M_j^r(1 - z_{i,r}) - A_j^r p_i(t) \ \text{is SOS for } t \in [T_i, T_{i+1}]$$

$$\sum_{r=1}^{R} z_{i,r} = 1 \qquad z \in \{0,1\}^{N \times R}$$

$\forall i \in \{1, \dots, N\}$, $r \in \{1, \dots, R\}$, $j \in \{1, 2\}$

Sum of Squares (SOS)

- $p(t) = \sum_k g_k^2(t)$
- $(d-1) \times (d-1)$ SDP for degree $\leq d$ polynomials

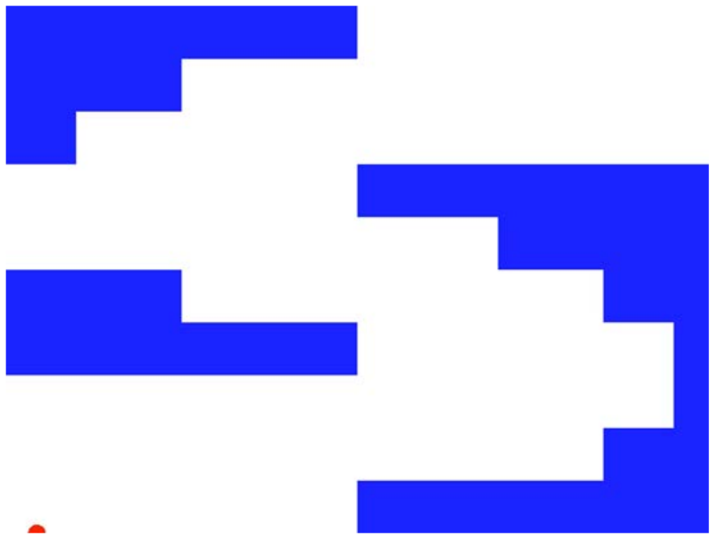
JUMP

+

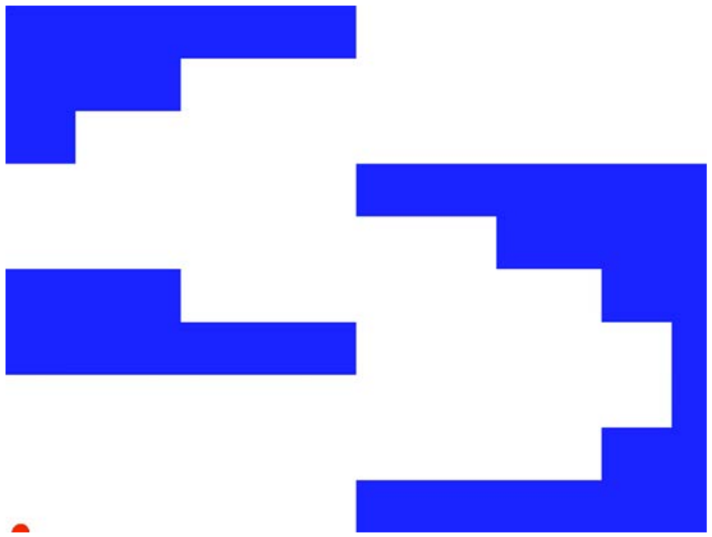

```julia
model = SOSModel(solver=PajaritoSolver())

@polyvar(t)
Z = monomials([t], 0:r)

@variable(model, H[1:N,boxes], Bin)

p = Dict()
for j in 1:N
   @constraint(model, sum(H[j,box] for box in boxes) == 1)
   p[(:x,j)] = @polyvariable(model, _, Z)
   p[(:y,j)] = @polyvariable(model, _, Z)
   for box in boxes
      xl, xu, yl, yu = box.xl, box.xu, box.yl, box.yu
      @polyconstraint(model, p[(:x,j)] >= Mxl + (xl-Mxl)*H[j,box], domain = (t >= T[j] && t <= T[j+1]))
      @polyconstraint(model, p[(:x,j)] <= Mxu + (xu-Mxu)*H[j,box], domain = (t >= T[j] && t <= T[j+1]))
      @polyconstraint(model, p[(:y,j)] >= Myl + (yl-Myl)*H[j,box], domain = (t >= T[j] && t <= T[j+1]))
      @polyconstraint(model, p[(:y,j)] <= Myu + (yu-Myu)*H[j,box], domain = (t >= T[j] && t <= T[j+1]))
   end
end

for ax in (:x,:y)
   @constraint(model,               p[(ax,1)      ]([0], [t]) == X₀[ax])
   @constraint(model, differentiate(p[(ax,1)], t   )([0], [t]) == X₀′[ax])
   @constraint(model, differentiate(p[(ax,1)], t, 2)([0], [t]) == X₀′′[ax])
   for j in 1:N-1
      @constraint(model,               p[(ax,j)    ]([T[j+1]],[t]) ==               p[(ax,j+1)    ]([T[j+1]],[t]))
      @constraint(model, differentiate(p[(ax,j)],t  )([T[j+1]],[t]) == differentiate(p[(ax,j+1)],t  )([T[j+1]],[t]))
      @constraint(model, differentiate(p[(ax,j)],t,2)([T[j+1]],[t]) == differentiate(p[(ax,j+1)],t,2)([T[j+1]],[t]))
   end
   @constraint(model,               p[(ax,N)      ]([1], [t]) == X₁[ax])
   @constraint(model, differentiate(p[(ax,N)], t   )([1], [t]) == X₁′[ax])
   @constraint(model, differentiate(p[(ax,N)], t, 2)([1], [t]) == X₁′′[ax])
end

@variable(model, γ[keys(p)] ≥ 0)
for (key,val) in p
   @constraint(model, γ[key] ≥ norm(differentiate(val, t, 3)))
end
@objective(model, Min, sum(γ))
```

```julia
function eval_poly(r)
    for i in 1:N
        if T[i] <= r <= T[i+1]
            return PP[(:x,i)]([r], [t]), PP[(:y,i)]([r], [t])
            break
        end
    end
end
```

Results for 9 Regions and 8 time steps

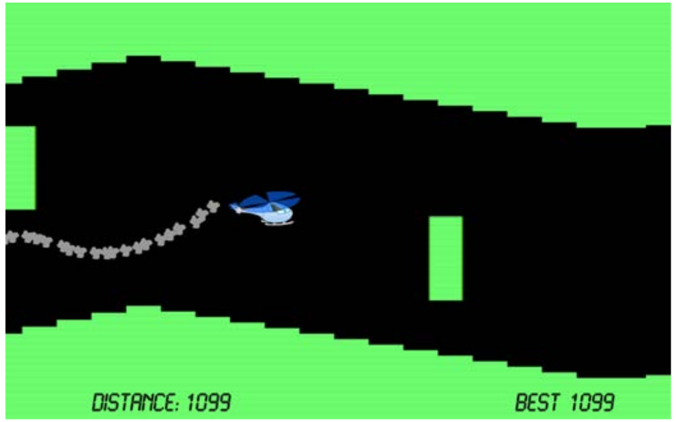


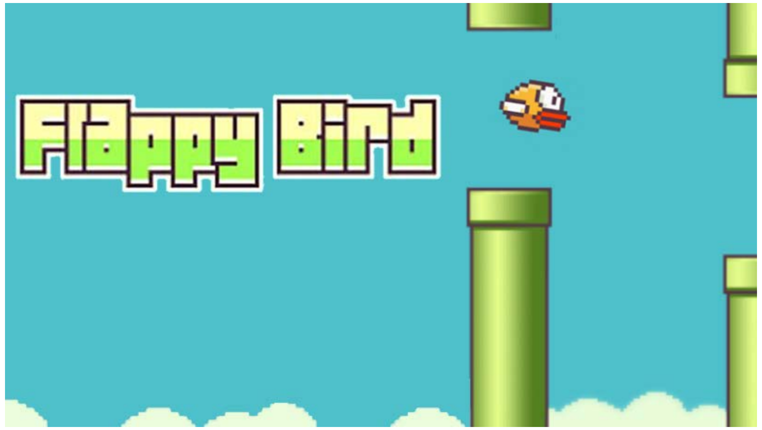
First Feasible Solution:
58 seconds



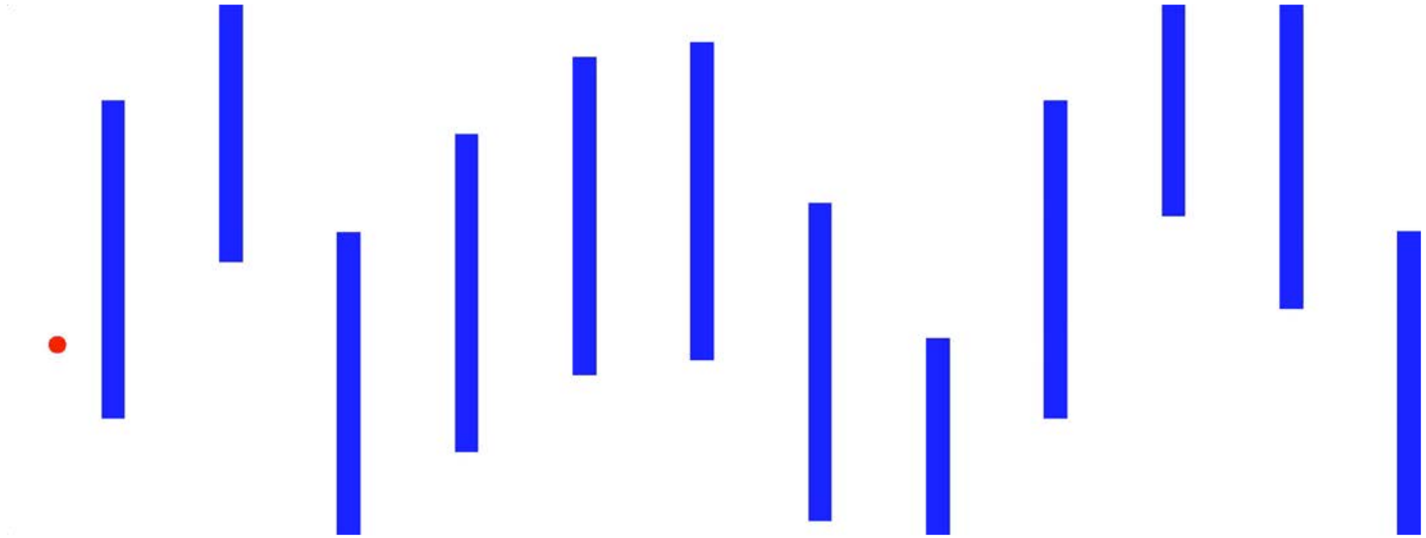
Optimal Solution:
651 seconds

Helicopter Game / Flappy Bird



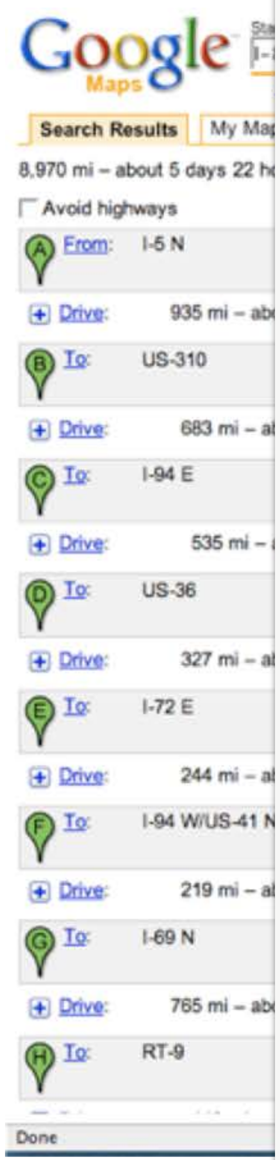


- 60 horizontal segments, obstacle every 5 = 80 sec. to opt.



Solving Mixed Integer Convex/Conic Optimization Problems

How hard is MICONV: Traveling Salesman Problem ?



The Washington Post

Quantum computers may be more of an imminent threat than AI,

Vivek Wadhwa, February 5, 2018

“As the number of cities increases, the problem becomes exponentially complex. It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example.”

MIP = Avoid Enumeration

- Number of tours for **49 cities** $48!/2 \approx \mathbf{10^{60}}$
- Fastest supercomputer $\approx$ $\mathbf{10^{17}}$ **flops**
- Assuming one floating point operation per tour:
  - more than $\mathbf{10^{25}}$ **times the age of the universe**!
- How long does it take on an **iphone**?
  - **Less than a second!**
  - 4 iterations of cutting plane method!
  - Dantzig, Fulkerson and Johnson 1954 did it by hand!
  - Cutting planes are the key for effectively solving (even NP-hard) MIP problems in practice.

50+ Years of MIP = Significant Solver Speedups

- Algorithmic Improvements (**Machine Independent**):
  - CPLEX → ILOG → IBM
    - v1.2 (1991) – v11 (2007): **29,000 x** speedup
  - GUROBI OPTIMIZATION
    - v1 (2009) – v6.5 (2015): **48.7 x** speedup

≈ **1.9 x / year**

- Also convex nonlinear:
  - GUROBI OPTIMIZATION
    - v6.0 (2014) – v6.5 (2015) quadratic: **4.43 x**

(V., Dunning, Huchette, Lubin, 2015)

State of MIP Solvers

- Mature: Linear and Quadratic (Conic Quadratic/SOCP)
  - Commercial:

    GUROBI OPTIMIZATION IBM CPLEX optimized by Xpress MP FICO

  - “Open Source”

    SCIP COIN|OR CBC GNU GLPK

- Emerging: Convex Nonlinear

  KNITRO COIN|OR Bonmin

MICONV B&B Algorithms

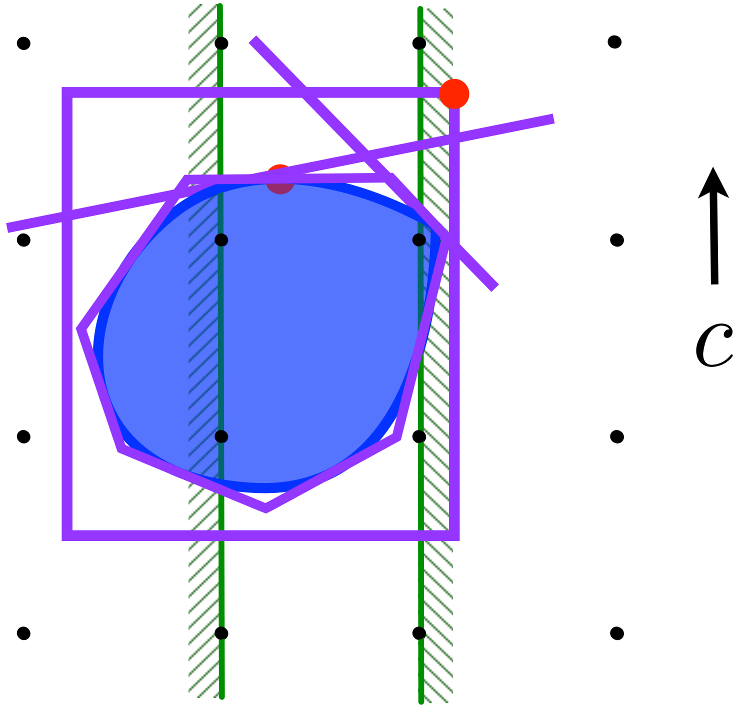
- NLP (QCP) Based B&B
- (Dynamic) LP Based B&B
  - Few cuts = high speed.
  - Possible slow convergence.
- Lifted LP B&B
  - Extended or Lifted relaxation.
  - Static relaxation
    - Mimic NLP B&B.
  - Dynamic relaxation
    - Standard LP B&B

$$\max \quad \sum_{i=1}^{n} c_i x_i$$

$$s.t. \quad Ax + Dz \leq b,$$

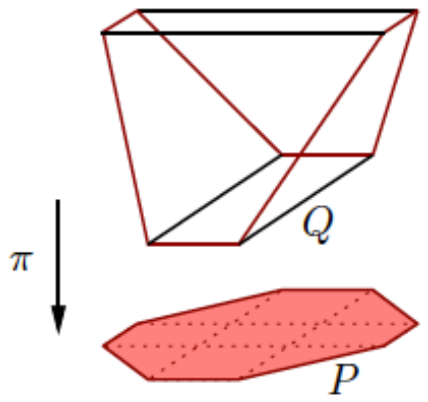
$$g_i(x) \leq 0,\ i \in I, \quad x \in \mathbb{Z}^n$$

$$x \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}$$



Lifted or Extended Approximations

- Projection = multiply constraints.
- V., Ahmed. and Nemhauser 2008:
  – Extremely accurate, but static and complex approximation by Ben-Tal and Nemirovski
- V., Dunning, Huchette and Lubin 2015: Simple, dynamic and good approximation:
  – First talks: May ‘14 (SIOPT), Dec ‘14 IBM
  – Paper in arxive, May ’15
  – Adopted in CPLEX v12.6.2, Jun 15’
  – Gurobi (Oct ‘15), Xpress (May ‘16), SCIP (Mar’ 17)



$$y_i^2 \leq z_i \cdot y_0 \quad \forall i \in [n]$$

$$\sum_{i=1}^n z_i \leq y_0$$

$$\|y\|_2 \leq y_0$$

Image from Lipton and Regan, https://rjlipton.wordpress.com

Not MICONV but, Mixed Integer Conic Programming (MICP)

$$\min_{\boldsymbol{x} \in \mathbb{R}^N} \quad \langle \boldsymbol{c}, \boldsymbol{x} \rangle :$$

$$\boldsymbol{b}_k - \boldsymbol{A}_k \boldsymbol{x} \in \mathcal{C}_k \quad \forall k \in [M]$$

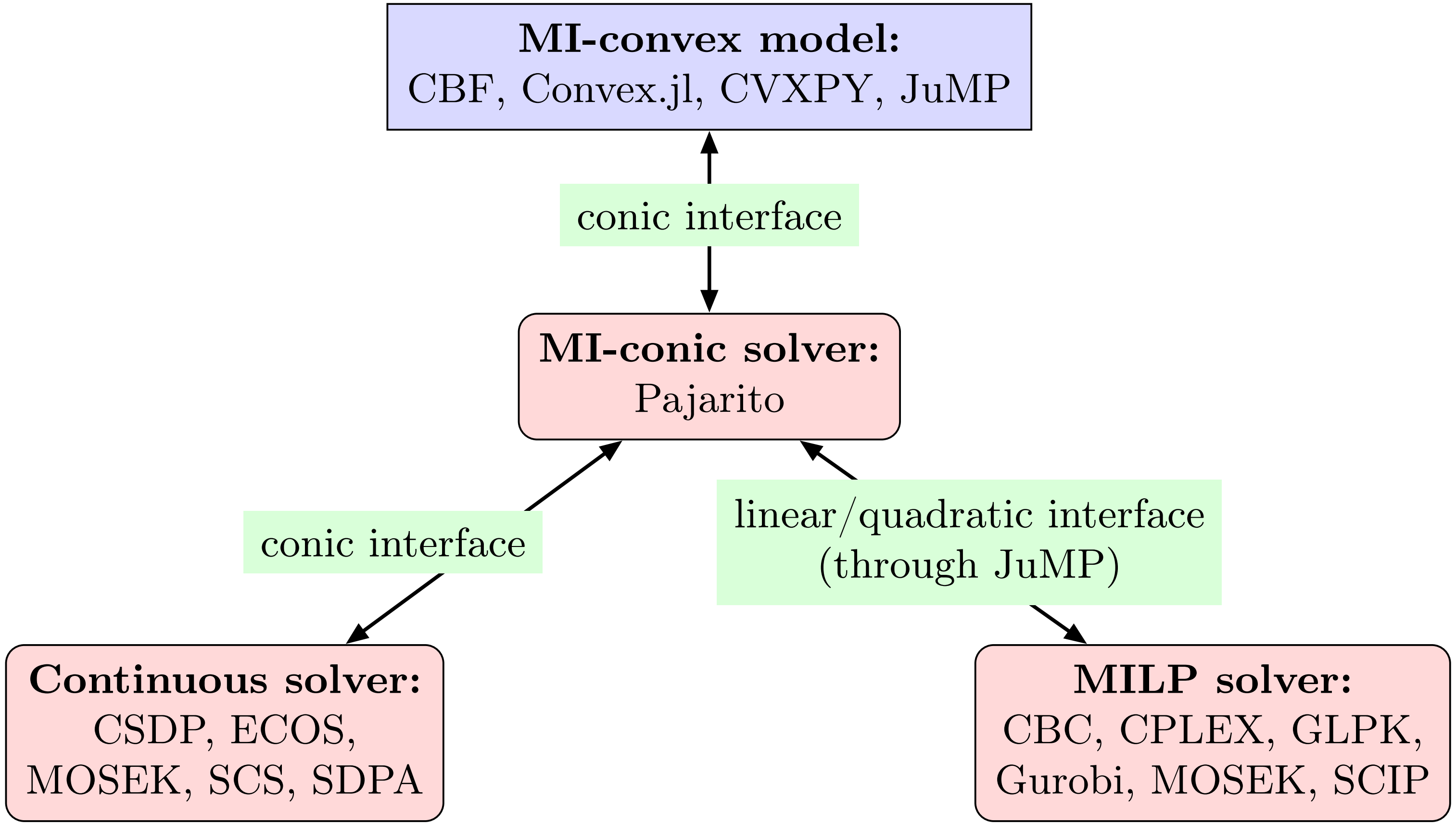
$$x_i \in \mathbb{Z} \qquad \forall i \in [I]$$

- $\mathcal{C}_k$ closed convex cones
  - Linear, SOCP, rotated SOCP, SDP
  - Exponential cone, power cone, …
  - Spectral norm, relative entropy, sum-of-squares, …

- Fast and stable interior point algorithms for continuous relaxation
- Geometrically intuitive conic duality guides linear inequality selection
- Conic formulation techniques usually lead to extended formulations
  - MINLPLIB2 instances unsolved since 2001 solved by re-write to MISOCP

Pajarito: A Julia-based MICP Solver

Los Alamos
NATIONAL LABORATORY



- Early version solved gams01, tls5 and tls6 (MINLPLIB2)

Performance for MISOCP Instances (120 from CBLIB)

	solver	statuses				
		ok	limit	error	wrong	time (s)
open source	Bonmin-BB	34	44	11	31	463
	Bonmin-OA	25	53	29	13	726
	Bonmin-OA-D	30	48	29	13	610
	Pajarito-GLPK-ECOS	56	60	3	1	377
	Pajarito-CBC-ECOS	78	30	3	9	163
restricted	SCIP (4.0.0)	74	35	8	3	160
	CPLEX (12.7.0)	90	16	5	9	50
	Pajarito-CPLEX-MOSEK (9.0.0.29-alpha)	97	20	2	1	56

Stability of CONIC Interior Point Algorithms is KEY!

- Why? Avoid non-differentiability issues? Stronger theory?
- Industry change in 2018:
  - KNITRO® version 11.0 adds support for SOCP constraints
  - mosek version 9.0 deprecates nonlinear formulations

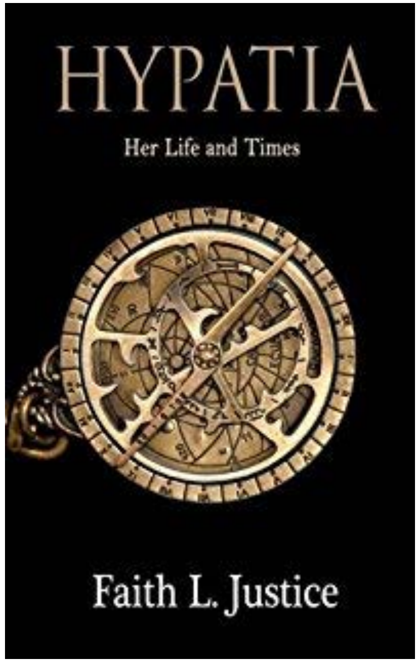
$$\begin{array}{lll} \min & f(x) & \\ \text{subject to} & g(x) & \leq \quad 0, \end{array}$$

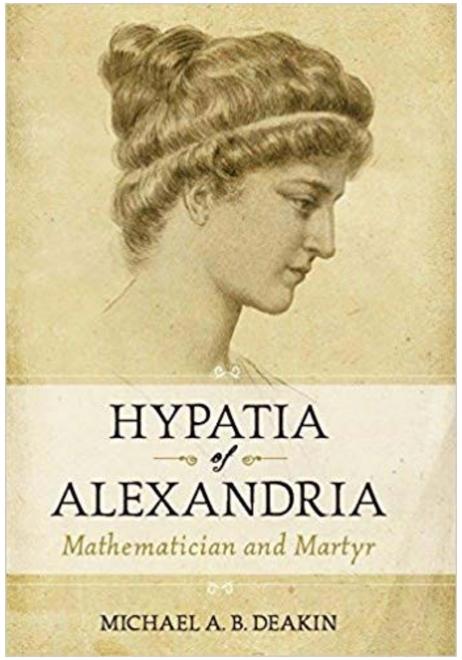
and focuses on pure conic (linear, SOCP, rotated SOCP, SDP, exp & power)

Hypatia: **Pure Julia-based** IPM Beyond "Standard" Cones

- Extension of methods in CVXOPT and Alfonso
  - A customizable homogeneous interior-point solver for nonsymmetric convex
  - Skajaa and Ye '15, Papp and Yıldız '17, Andersen, Dahl, and Vandenberghe '04-18
- Cones: LP, dual Sum-of-Squares, SOCP, RSOCP, 3-dim exponential cone, PSD, L_∞, n-dim power cone (using AD), spectral norm, …
- Potential:
  - flexible number types and linear algebra
  - BOB: bring your own barrier (in ~50 lines of code)
  - Alternative prediction steps (Runge–Kutta)

Chris Coey





Early Comparison with Alfonso for LP and SOS

First Hypatia commit : Jul 15

			Aug 5	Aug 19	Aug 23
test	iters	Matlab	75cba5f	c9f1eb5	133b422
dense lp	65	5.8	4.1	2.03	1.25
envelope	30	0.085	0.043	0.020	x
butcher	32/30	0.63	0.41	0.357	0.136
caprasse	31/30	1.38	1.87	1.80	0.530
lotka-volt	31/30	0.47	0.38	0.37	0.104
motzkin	41/42	0.35	0.24	x	0.054
reac-diff	29/30	0.32	0.23	0.19	0.075
robinson	29	0.34	0.23	0.17	0.034

Linear Optimization (dense lp)

Polynomial Envelope (envelope)

Polynomial Minimization (butcher – robinson)

- First Batch of Tests on CBLIB Instances (SDP/SOCP): Only 2 – 10K times slower than Mosek 8!

Modeling with Conic Optimization

How to get conic representation?

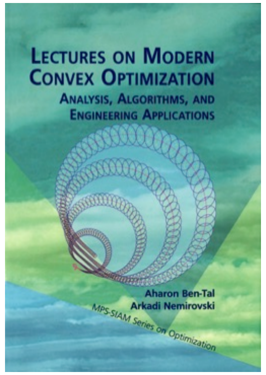
$$C_i = \left\{ x \in \mathbb{R}^d \ : \ \begin{array}{c} \exists u \in \mathbb{R}^{p_i} \text{ s.t.} \\ A^i x + D^i u - b \in K^i \end{array} \right\}$$

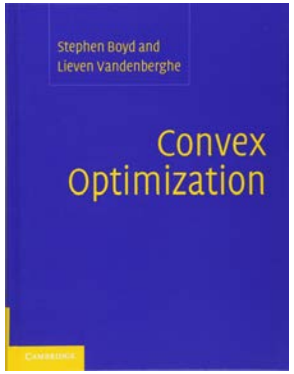
- Using modeling tools like Disciplined Convex Programming (DCP)

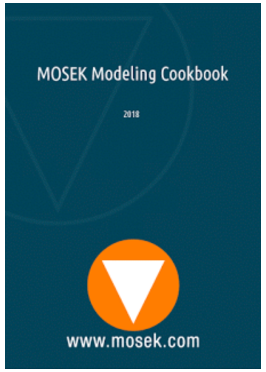
CVX RESEARCH

Convex.jl in julia

- Using standard constructions for standard cones







← **MMC**

- Relatively mechanical, but understanding details can help performance

MICP Example 1: MI - Second Order Cone (SOCP)

$$\min_{\boldsymbol{x}\in\mathbb{R}^N} \quad \langle \boldsymbol{c}, \boldsymbol{x} \rangle :$$

$$\boldsymbol{b}_k - \boldsymbol{A}_k \boldsymbol{x} \in \mathcal{C}_k \quad \forall k \in [M]$$

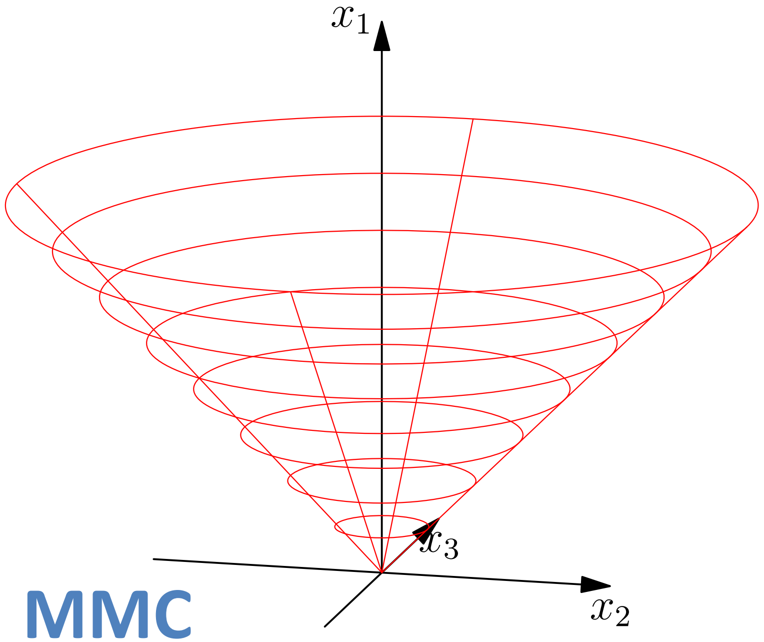
$\mathcal{C}_k$ is a *closed convex cone*

$$\alpha_1 \boldsymbol{y}_1 + \alpha_2 \boldsymbol{y}_2 \in \mathcal{C}_k \qquad \forall \alpha_1, \alpha_2 \geq 0 \qquad \forall \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{C}_k$$

$$\left(\sigma, Q^{1/2}x\right) \in \mathcal{L}^{1+n} \qquad \mathcal{L}^{1+n} = \{(r, \boldsymbol{t}) \in \mathbb{R}^{1+n} : r \geq \|\boldsymbol{t}\|_2\}.$$

Lorentz cone or Second Order Cone

$$\mathcal{Q}^n = \left\{ x \in \mathbb{R}^n \mid x_1 \geq \sqrt{x_2^2 + x_3^2 + \cdots + x_n^2} \right\}$$



MMC

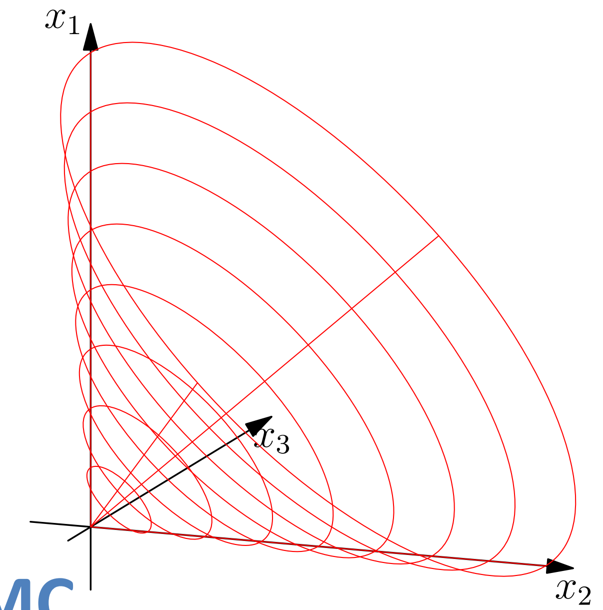
Portfolio Optimization Problems:

$$\begin{aligned} \max \quad & \bar{a}x \\ s.t. \quad & \\ & \left\|Q^{1/2}x\right\|_2 \leq \sigma \\ & \sum_{j=1}^{n} x_j = 1, \quad x \in \mathbb{R}^n_+ \\ & x_j \leq z_j \quad \forall j \in [n] \\ & \sum_{j=1}^{n} z_j \leq K, \quad z \in \{0,1\}^n \end{aligned}$$

- $\bar{a}$ expected returns.
- $Q^{1/2}$ square root of covariance matrix.
- K maximum number of assets.
- σ maximum risk.

MICP Example 1a: Rotated Second Order Cone (RSOCP)

- RSOCP : $\mathcal{Q}_r^n = \{x \in \mathbb{R}^n \mid 2x_1x_2 \geq x_3^2 + \cdots + x_n^2,\ x_1, x_2 \geq 0\}$
- Can be used to model geometric mean



MMC

assume $n = 2^k$

$$t \leq \left(\prod_{i=1}^{n} \delta_i\right)^{1/n}$$

$$\Updownarrow$$

$$\begin{array}{llll}
u_{1,1}^2 \leq \delta_1 \cdot \delta_2 & u_{2,1}^2 \leq u_{1,1} \cdot u_{1,2} & & \\
u_{1,2}^2 \leq \delta_3 \cdot \delta_4 & u_{2,2}^2 \leq u_{1,3} \cdot u_{1,4} & \cdots & t^2 \leq u_{k-1,1} \cdot u_{k-1,2} \\
\vdots & \vdots & & \\
u_{1,n/2}^2 \leq \delta_{n-1} \cdot \delta_n & u_{2,n/4}^2 \leq u_{1,n/2-1} \cdot u_{1,n/2} & &
\end{array}$$

MICP Example 1a: Rotated Second Order Cone (RSOCP)

- Exp-Cone :

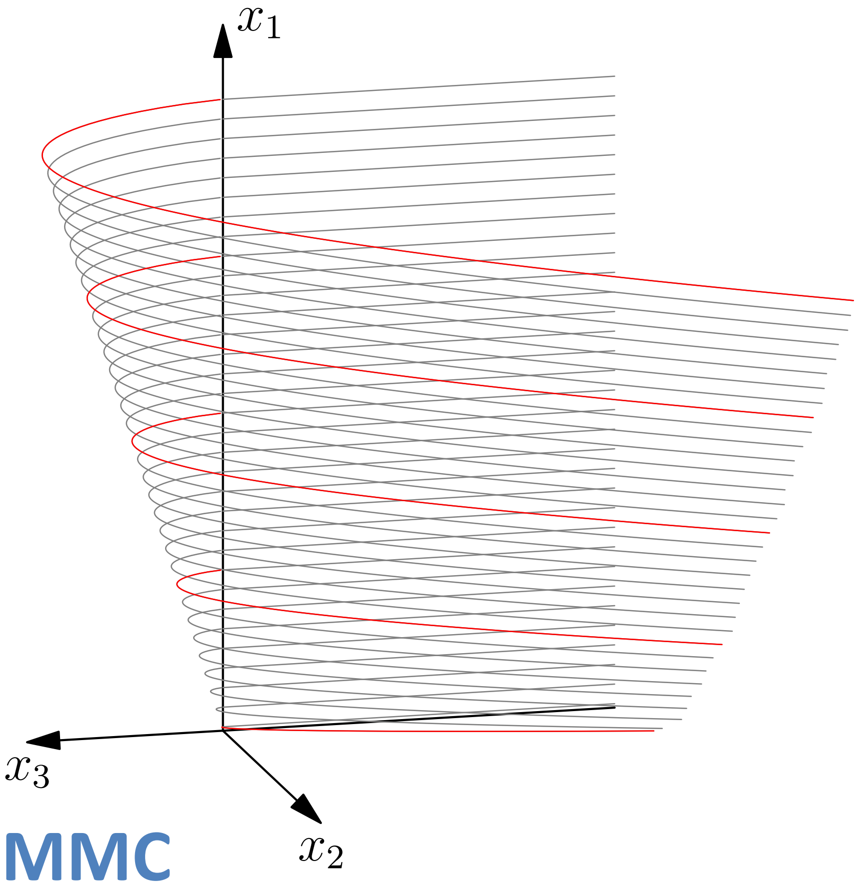
$$K_{\exp} \;=\; \{(x_1, x_2, x_3) \;:\; x_1 \geq x_2 e^{x_3/x_2},\; x_2 > 0\} \cup \{(x_1, 0, x_3) \;:\; x_1 \geq 0,\; x_3 \leq 0\}$$

- Can be used to model log-sum-exp

$$t \geq \log(e^{x_1} + \cdots + e^{x_n})$$

$$\Updownarrow$$

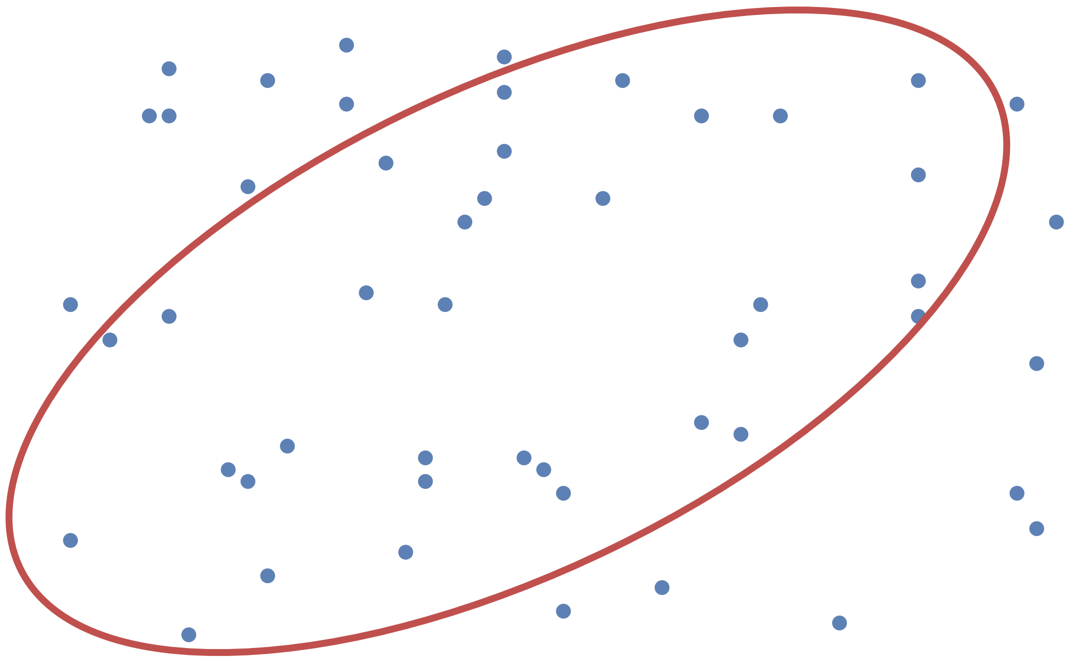
$$\begin{aligned} \sum u_i &\leq 1, \\ (u_i, 1, x_i - t) &\in K_{\exp},\; i = 1, \ldots, n. \end{aligned}$$



MMC

MICP Example 2: MI – Semidefinite Programming (+ Exp Cone)

- Find minimum volume ellipsoid that contains 90% of data points



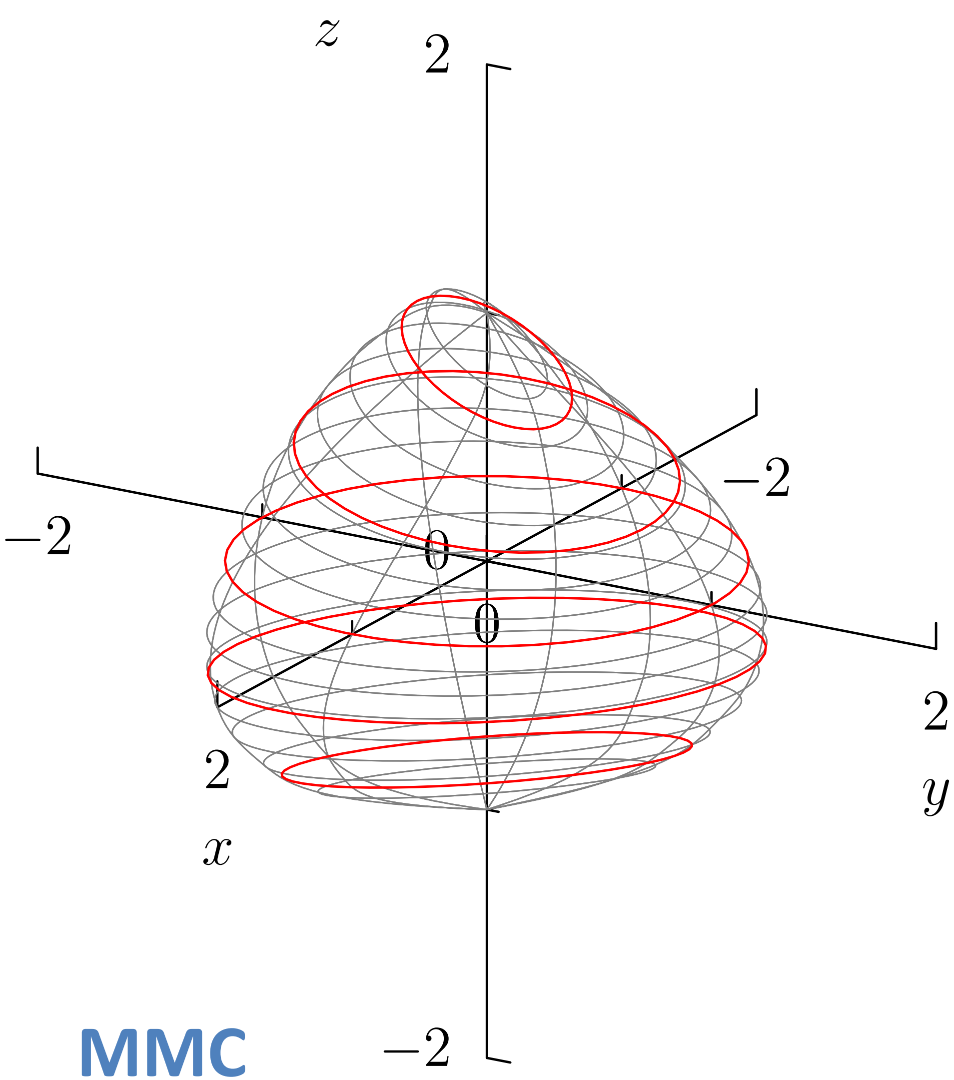
MICP Example 2: MI – Semidefinite Programming (+ Exp Cone)

- Semidefinite constraints:
  - $\mathcal{S}^n$: set of symmetric matrices
  - Set of positive semidefinite symmetric matrices:

$$\mathcal{S}^n_+ = \{X \in \mathcal{S}^n \mid z^T X z \geq 0,\ \forall z \in \mathbb{R}^n\}$$

$$X \succeq Y \quad \Leftrightarrow \quad (X - Y) \in \mathcal{S}^n_+$$

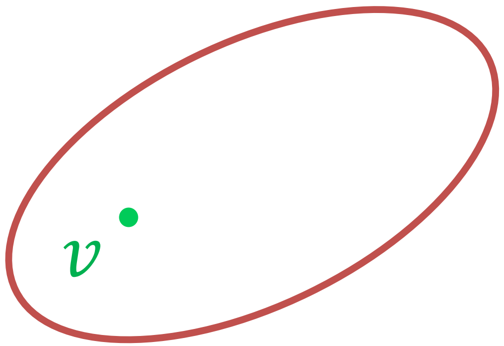
$$A(x,y,z) = \begin{bmatrix} 1 & x & y \\ x & 1 & z \\ y & z & 1 \end{bmatrix} \succeq 0$$



MMC

MICP Example 2: SDP representation of ellipsoid take 1

$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \leq 1\}, \qquad A \in \boldsymbol{S}_+^n$



$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2}$

$$(v-c)^T A (v-c) \leq 1$$

$$\Leftrightarrow v^T A v - 2v^T A c + c^T A c \leq 1$$

$$\Leftrightarrow v^T A v - 2v^T A c + s \leq 1$$

$$c^T A c \leq s$$

$$\Leftrightarrow v^T A v - 2v^T \boxed{Ac} + s \leq 1$$

$$\begin{pmatrix} s & c^T \\ c & A^{-1} \end{pmatrix} \succcurlyeq 0$$

Schur Complement Lemma:

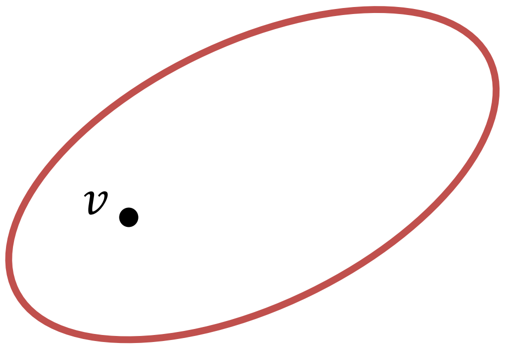
$$A = \begin{pmatrix} B & C^T \\ C & D \end{pmatrix} \succeq 0$$

$$\Updownarrow$$

$$D - CB^{-1}C^T \succeq 0$$

MICP Example 2: SDP representation of ellipsoid take 2

$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \le 1\}$, $A \in \boldsymbol{S}_+^n$



$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2}$

$$
\begin{aligned}
& (v-c)^T A (v-c) \le 1 \\
\Leftrightarrow\ & v^T A v - 2v^T A c + c^T A c \le 1 \\
\Leftrightarrow\ & v^T A v - 2v^T A c + s \le 1 \\
& \qquad c^T A c \le s \qquad \longleftarrow \quad z = Ac \\
\Leftrightarrow\ & v^T A v - 2v^T z + s \le 1 \\
& \qquad z^T A^{-1} z \le s \\
\Leftrightarrow\ & \boxed{\begin{array}{c} v^T A v - 2v^T z + s \le 1 \\ \begin{pmatrix} s & z^T \\ z & A \end{pmatrix} \succcurlyeq 0 \quad (🐶) \end{array}}
\end{aligned}
$$

Schur Complement Lemma:

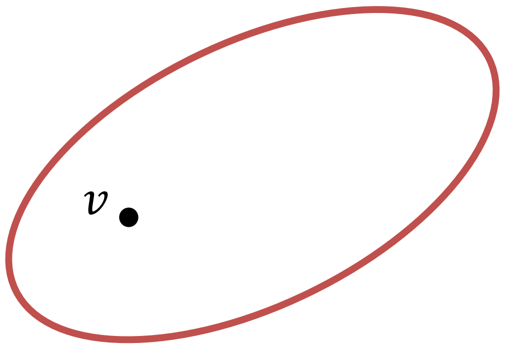
$$A = \begin{pmatrix} B & C^T \\ C & D \end{pmatrix} \succeq 0$$

$$\Updownarrow$$

$$D - CB^{-1}C^T \succeq 0$$

MICP Example 2: SDP representation of determinant part 1

$E = \{x \ \in \mathbb{R}^n \colon (x-c)^T A(x-c) \leq 1\},$ $\qquad A \in \boldsymbol{S}_+^n$



$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2}$

$$\prod_{i=1}^{n} D_{i,i} \leq \mathrm{Det}(A)$$

$$\Updownarrow$$

$$\begin{pmatrix} D & U \\ U^T & A \end{pmatrix} \succcurlyeq 0,$$

$$\mathrm{diag}(U) = \mathrm{diag}(D)$$

U is upper triangular

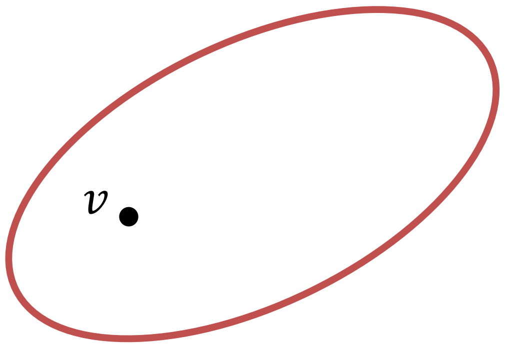
D is diagonal

(🐱)

MICP Example 2: SDP representation of determinant part 2a

$$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \le 1\}, \qquad A \in \boldsymbol{S}_+^n$$

$$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2} \qquad \prod_{i=1}^n D_{i,i} \le \mathrm{Det}(A)$$

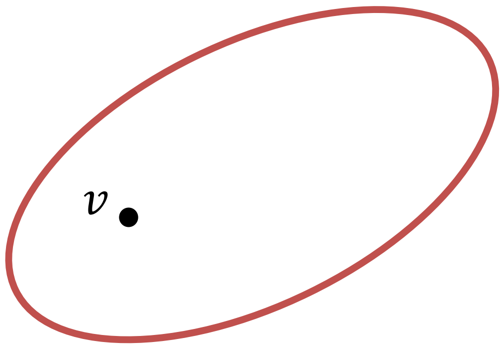


$$t \le \left(\prod_{i=1}^n D_{i,i}\right)^{1/n} \Leftrightarrow \boxed{\begin{array}{l} u_{1,1}^2 \le D_{1,1} \cdot D_{2,2} \\ u_{1,2}^2 \le D_{3,3} \cdot D_{4,4} \\ \quad\vdots \\ u_{1,n/2}^2 \le D_{(n-1,n-1)} \cdot D_{n,n} \end{array} \quad \cdots \quad t^2 \le u_{k-1,1} \cdot u_{k-1,2}}$$

()

MICP Example 2: SDP representation of determinant part 2a

$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \leq 1\}$, $A \in \boldsymbol{S}_+^n$



$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2}$

$$\prod_{i=1}^{n} D_{i,i} \leq \mathrm{Det}(A)$$

min $\mathrm{Vol}(E)$

$\Updownarrow$

max $(\mathrm{Det}(A))^{1/n}$

$\Leftrightarrow$

max t

$s.t.$

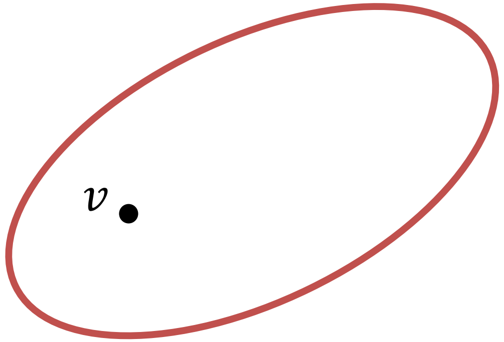
(🐶) SDP

(🐱) SDP SDP

(🐭) RSOCP

MICP Example 2: SDP representation of determinant part 2b

$$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \leq 1\}, \qquad A \in \boldsymbol{S}_+^n$$



$$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2} \qquad \prod_{i=1}^{n} \delta_{i,i} \leq \mathrm{Det}(A)$$

$$t \leq \sum_{i=1}^{n} \mathrm{Log}\,(D_{i,i}) \leq \mathrm{Log}(\mathrm{Det}(A))$$

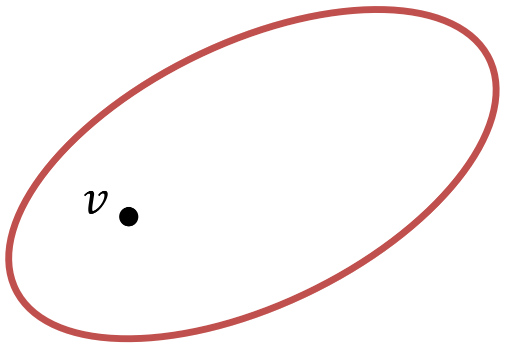
$$t \leq \sum_{i=1}^{n} \mathrm{Log}\,(D_{i,i}) \quad \Leftrightarrow \quad \boxed{\begin{array}{l} t \leq \sum_{i=1}^{n} t_i \\ \left(D_{i,i}, 1, t_i\right) \in K_{\exp} \end{array}}$$

(🐹)

MICP Example 2: SDP representation of determinant part 2b

$$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \leq 1\}, \qquad A \in \boldsymbol{S}_+^n$$

$$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2} \qquad \prod_{i=1}^{n} D_{i,i} \leq \mathrm{Det}(A)$$



$$\min \ \mathrm{Vol}(E)$$

$$\Updownarrow$$

$$\max \ \mathrm{Log}(\mathrm{Det}(A))$$

$$\Leftrightarrow$$

$$\max \ t$$
$$s.t.$$

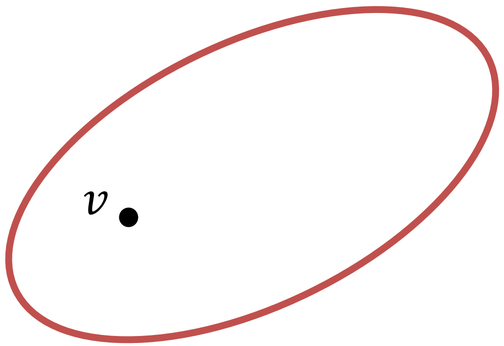
(🐶) SDP

(🐱) SDP SDP + EXP

(🐹) EXP-Cone

MICP Example 2: MI part = choose points

$$E = \{x \in \mathbb{R}^n : (x-c)^T A (x-c) \le 1\}, \qquad A \in \boldsymbol{S}_+^n$$



$$\mathrm{Vol}(E) \propto \mathrm{Det}(A)^{-1/2}$$

$$v^T A v - 2v^T z + s \le 1 \quad \forall\, v \in V$$

$$\begin{pmatrix} s & z^T \\ z & A \end{pmatrix} \succcurlyeq 0$$

(🐶)

$$v^T A v - 2v^T z + s \le 1 + M(1 - y_v) \quad \forall\, v \in V$$

$$\textstyle\sum_{v \in V} y_v \ge 0.9|V|$$

$$y_v \in \{0,1\} \quad \forall\, v \in V$$

$$\begin{pmatrix} s & z^T \\ z & A \end{pmatrix} \succcurlyeq 0$$

(🦊)

References:

- Conic Optimization:
  - http://www2.isye.gatech.edu/~nemirovs/LMCO_LN.pdf
  - https://web.stanford.edu/~boyd/cvxbook/
  - https://themosekblog.blogspot.com/2018/05/new-modeling-cookbook.html
- MI-Conic Optimization:
  - https://arxiv.org/abs/1808.05290