

# Modeling Power of Mixed Integer Convex Optimization Problems And Their Effective Solution with Julia and JuMP

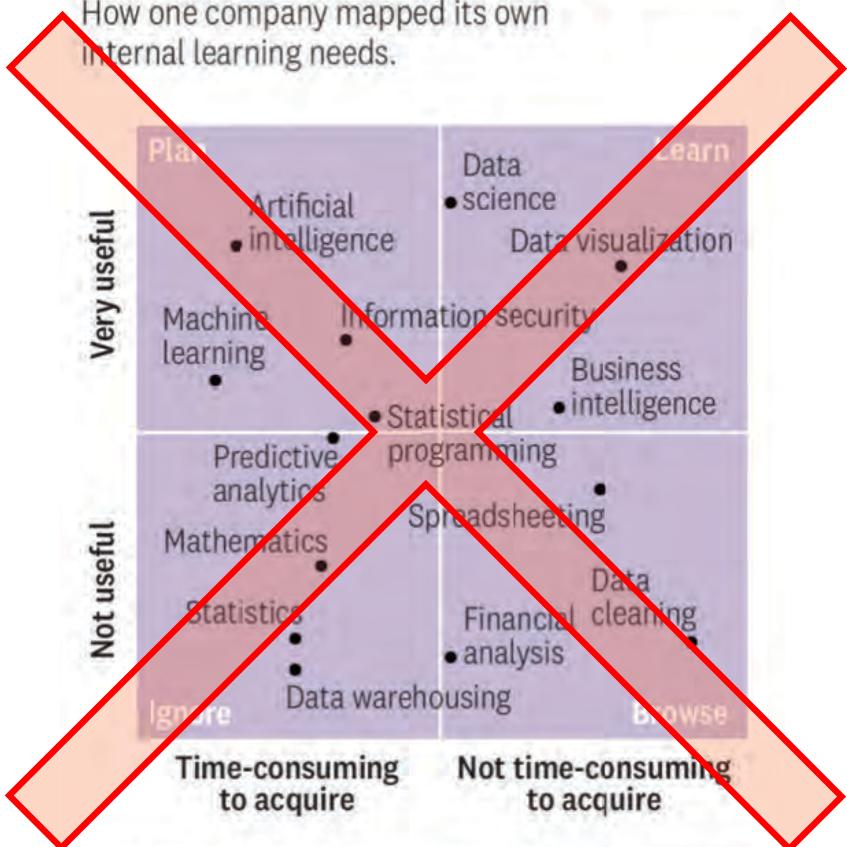
Juan Pablo Vielma

Massachusetts Institute of Technology

Michigan Institute for Computational Discovery and Engineering (MICDE) and  
Department of Industrial and Operations Engineering (IOE),  
University of Michigan at Ann Arbor,  
Ann Arbor, Michigan, October, 2018.

## An Example of How to Plot Data Skills on a 2x2 Learning Matrix

How one company mapped its own internal learning needs.

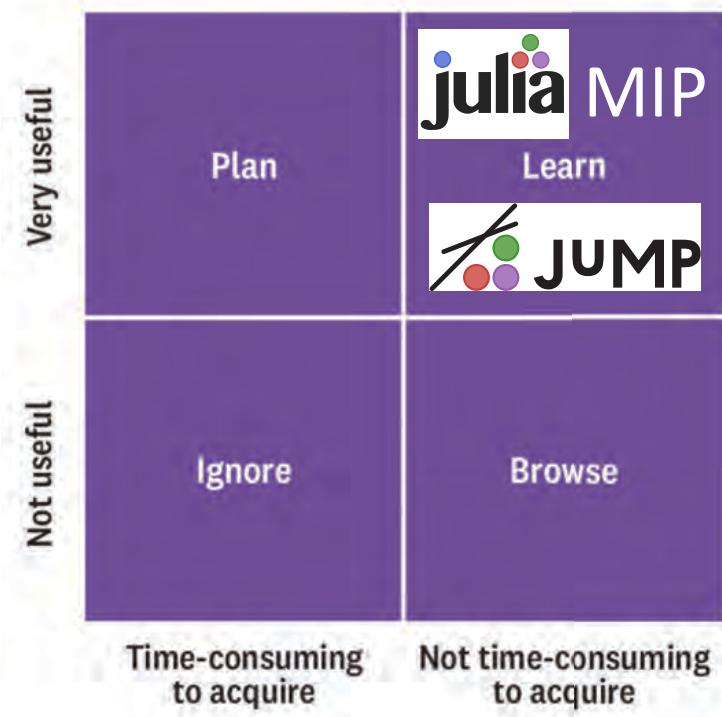


Source: Analysis of internal data learning needs by Filtered

© HBR

## Which Data Skills Should You Learn First?

Make the most of your limited learning time.



Source: Filtered

© HBR

<https://hbr.org/2018/10/which-data-skills-do-you-actually-need-this-2x2-matrix-will-tell-you>

# Mixed Integer Convex Optimization (MICONV)

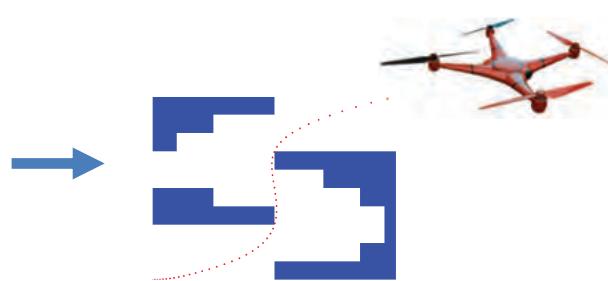
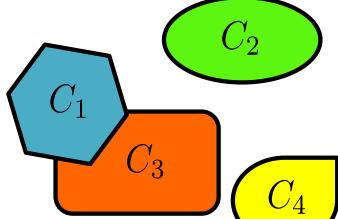
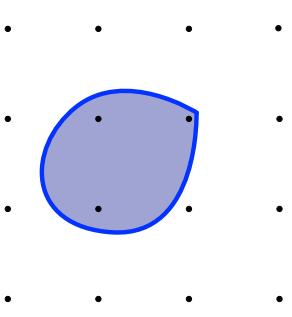
$$\min f(x)$$

s.t.

$$x \in C$$

$$x_i \in \mathbb{Z} \quad i \in I$$

convex  $f$  and  $C$ .

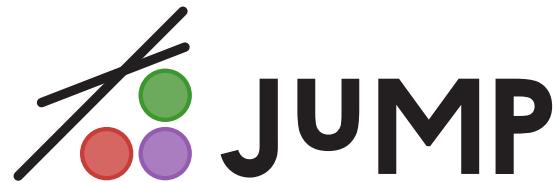


<http://www.gurobi.com/company/example-customers>

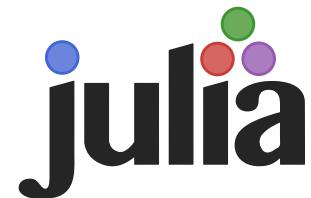
## Overview

---

- What can we model with MICONV
- How can we solve MICONVs
  - How can we access solvers



- How solvers work



# What can we model with MICONV?

Joint work with Miles Lubin and Ilias Zadik

## What Can MICONV Model?

---

- Optimal discrete experimental design
- Obstacle avoidance and trajectory planning in optimal control
- Portfolio optimization with nonlinear risk measures and combinatorial constraints
- ...

## No, Really. What Can MICONV Model?

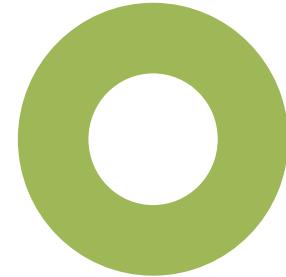
---

Two sheet hyperbola?



$$\{x \in \mathbb{R}^2 : 1 + x_1^2 \leq x_2^2\}$$

Spherical shell?



$$\{x \in \mathbb{R}^2 : 1 \leq \|x\| \leq 2\}$$

- Integer points in parabola  $\{(x, x^2) : x \in \mathbb{Z}\}$ ?
- The set of  $n \times n$  matrices with rank  $\leq k$ ?
- Set of prime numbers?

## MICONV Can Model Any Finite Union of (Closed) Convex Sets

---

- Let  $T_1, \dots, T_k$  be **closed convex** set. A MICONV formulation of  $x \in \bigcup_{i=1}^k T_i$ :

$$(x^i, z_i) \in \overline{\text{cone}}(T_i \times \{1\}) \quad \forall i \in \{1, \dots, k\}$$

$$\|x^i\|_2^2 \leq z_i t_i. \quad \forall i \in \{1, \dots, k\}$$

$$\sum_{i=1}^k x^i = x,$$

$$\sum_{i=1}^k z_i = 1,$$

$$z \in \{0,1\}^k,$$

$$t \in \mathbb{R}_+^k,$$

$$x^i \in \mathbb{R}^n \quad \forall i \in \{1, \dots, k\}$$

## No, Really. What Can MICONV Model?

---

Two sheet hyperbola?



$$\{x \in \mathbb{R}^2 : 1 + x_1^2 \leq x_2^2\}$$

Spherical shell?



$$\{x \in \mathbb{R}^2 : 1 \leq \|x\| \leq 2\}$$

- Integer points in parabola  $\{(x, x^2) : x \in \mathbb{Z}\}$ ?
- The set of  $n \times n$  matrices with rank  $\leq k$ ?
- Set of prime numbers?

## A Simple Obstruction for MICONV Formulations

---

- $S$  cannot have a MICONV formulation if there exists:
  - There exist infinite  $R \subseteq S$  s.t.

$$\frac{u + v}{2} \notin S \quad \forall u, v \in R, u \neq v$$

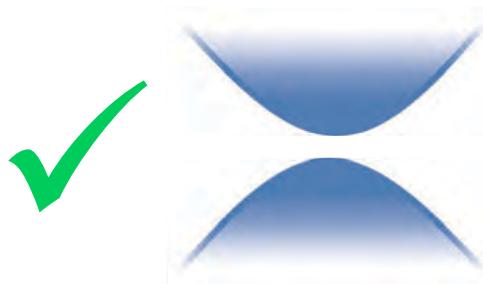
**X Spherical shell**  $\{x \in \mathbb{R}^2 : 1 \leq \|x\| \leq 2\}$



## No, Really. What Can MICONV Model?

---

Two sheet hyperbola?



$$\{x \in \mathbb{R}^2 : 1 + x_1^2 \leq x_2^2\}$$

Spherical shell?



$$\{x \in \mathbb{R}^2 : 1 \leq \|x\| \leq 2\}$$

✗ Integer points in parabola  $\{(x, x^2) : x \in \mathbb{Z}\}$ ?

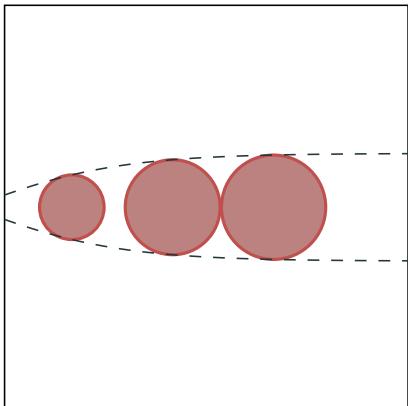
✗ The set of  $n \times n$  matrices with rank  $\leq k$ ?

✗ Set of prime numbers?

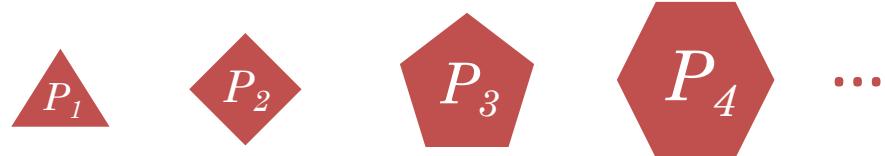
Does have non-convex polynomial MIP formulation

## MICONV = Structured Countably Infinite Unions of Convex Sets

---



- Can be “strange” unions, e.g. :
  - Infinite number of shapes



- There exist an increasing function  $h$  such that:
  - $P_z \subseteq \mathbb{R}^2$  regular  $h(z)$ -gon
  - $S = \bigcup_{z=1}^{\infty} P_z$  has MICONV formulation
- Equal volume  $\Rightarrow$  Finite # of Shapes

$$\sqrt{(x_1 - 2z)^2 + x_2^2} \leq 1 - 1/z, \\ z \geq 1, \quad z \in \mathbb{Z}$$

## Sets with MICONV Formulations can be REALLY “Strange”

---

- Dense discrete set  $\{\sqrt{2}x - \lfloor \sqrt{2}x \rfloor : x \in \mathbb{N}\} \subseteq [0, 1]$

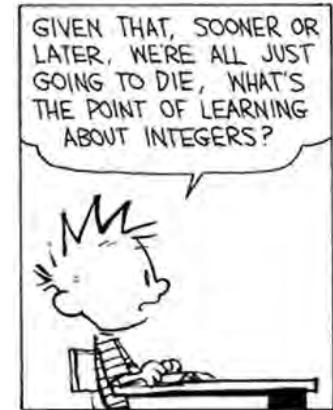
$$\begin{aligned}\|(z_1, z_1)\|_2 &\leq z_2 + 1, & \|(z_2, z_2)\|_2 &\leq 2z_1, & x_1 = y_1 - z_2, \\ \|(z_1, z_1)\|_2 &\leq y_1, & \|(y_1, y_1)\|_2 &\leq 2z_1, & z \in \mathbb{Z}^2\end{aligned}$$

- Non-Periodic Set of naturals  $\{x \in \mathbb{N} : \sqrt{2}x - \lfloor \sqrt{2}x \rfloor \notin (\varepsilon, 1 - \sqrt{2}\varepsilon)\}$

$$\begin{aligned}\|(x_1, x_1)\|_2 &\leq x_2 + \varepsilon, \\ \|(x_2, x_2)\|_2 &\leq 2x_1 + 2\varepsilon, & x &\in \mathbb{Z}_+^2\end{aligned}$$



"God made the integers, all else is the work of man"  
- Leopold Kronecker





MIP

## 50+ Years of MIP = Significant Solver Speedups

---

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



- “Open Source”



- Emerging: Convex Nonlinear (e.g. SDP)
  - Open-Source + Commercial linear MIP Solver > Commercial

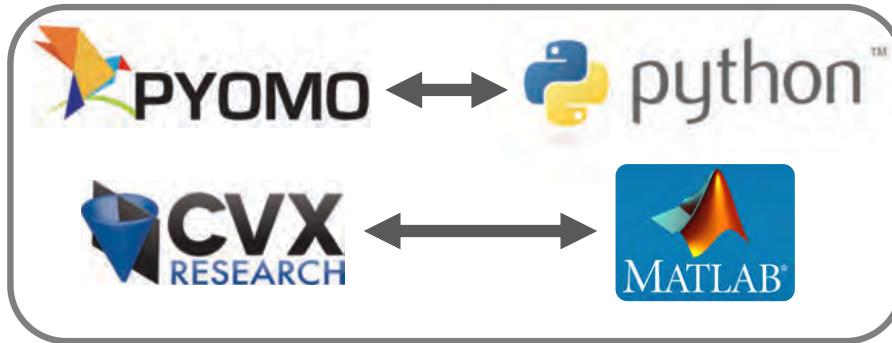
## Accessing MIP Solvers = Modelling Languages

---

- User-friendly algebraic modelling languages (AML):



Standalone and Fast



Based on General Language and Versatile

- Fast and Versatile, but complicated (and possibly proprietary)
  - Low-level C/C++ solver or Coin-OR interphases & frameworks
- 21st Century AMLs:



# 21st Century Programming/Modelling Languages

---



- Open-source and free!
- Developed at MIT
- “Floats like python/matlab, strings like C/Fortran”
- Easy to use and wide library ecosystem (specialized and frontend)
- Only language besides C/C++/Fortran to scale to 1 Petaflop!

- Open-source and free!
- Modelling language, interface and software ecosystem for optimization
- Easy to use and advanced
- Integrated into Julia
- Created at MIT and beyond...

# Large Software Stack and Vibrant Community



**JuliaCon is coming to London!**

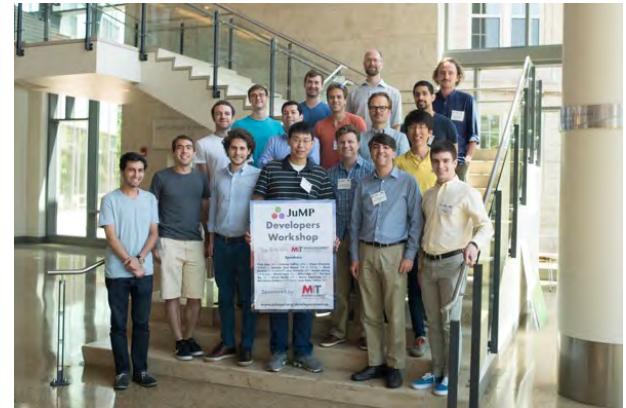
7th to 11th August, 2018 at University College London  
Roberts Engineering Building, Torrington Place, WC1E 7JE, London, UK



# Large Software Stack and Vibrant Community



2016



Iain Dunning, Miles Lubin  
and Joey Huchette



JU  
MP

# Not Just a Modeling Language / Interphase

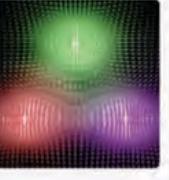
- JuMP domain specific language (DSL)
- Solve abstraction layers:
  - MathProgBase / MathOptInterface
- Solver interfaces
- Solvers: Pajarito.jl, Pavito.jl
- Extensions: SumOfSquares.jl, PolyJuMP.jl
- Now a NumFOCUS Sponsored project!



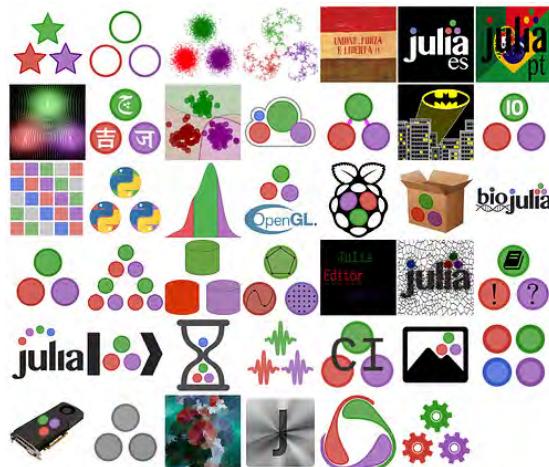
Modeling Tool	Linear / Quadratic	Convex		Nonconvex	Integer
		Conic	Smooth		
JuMP	✓	✓	✓	✓	✓
Convex.jl	✓	✓			✓
Solver					
CDD (.jl)	✓				
Cip (.jl)	✓				
OSQP (.jl)	✓				
Cbc (.jl)	✓				✓
GLPK (.jl)	✓				✓ <sup>cb</sup>
CSDP (.jl)	✓	✓			
ECOS (.jl)	✓	✓			
SCS (.jl)	✓	✓			
SDPA (.jl)	✓	✓			
CPLEX (.jl)	✓	✓			✓ <sup>cb</sup>
Gurobi (.jl)	✓	✓			✓ <sup>cb</sup>
FICO Xpress (.jl)	✓	✓			✓
Mosek (.jl)	✓	✓	✓		✓
Pajarito.jl	✓	✓	✓		✓
NLOpt (.jl)			✓	✓	
Ipopt (.jl)	✓		✓	✓	
Bonmin (via AmplNLWriter.jl)	✓		✓	✓	✓
Couenne (via AmplNLWriter.jl)	✓		✓	✓	✓
Artelys Knitro (.jl)	✓		✓	✓	✓
SCIP (.jl)	✓	✓	✓	✓	✓ <sup>cb</sup>

# Other Related Projects

 **JuliaDiffEq**  
An organization for differential equations in Julia  
<http://juliadiffeq.org> [contact@juliadiffeq.org](mailto:contact@juliadiffeq.org)

 **JuliaDiff**  
Differentiation Tools in Julia  
<http://www.juliadiff.org/>

**Repositories 10** **People 4** **Projects 0**



 **JuliaML**  
Julia packages for Machine Learning  
<https://juliaml.github.io>

**Repositories 25** **People 10** **Projects 0**

 **Julia Statistics**  
Statistics and Machine Learning made easy in Julia  
<https://discourse.julialang.org/c/domain/stats>

**Repositories 36** **People 15** **Projects 0**

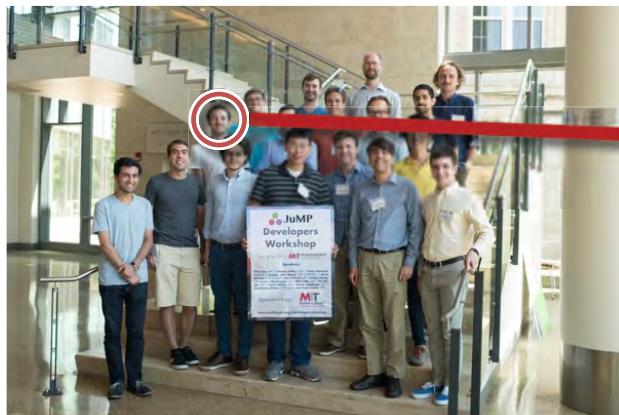
<https://julialang.org/community/>

<https://juliaobserver.com>

Who is using **julia** &  **JUMP** ?

... outside of 

# Technological Solutions for Electrical Markets



**Joaquim Dias Garcia**



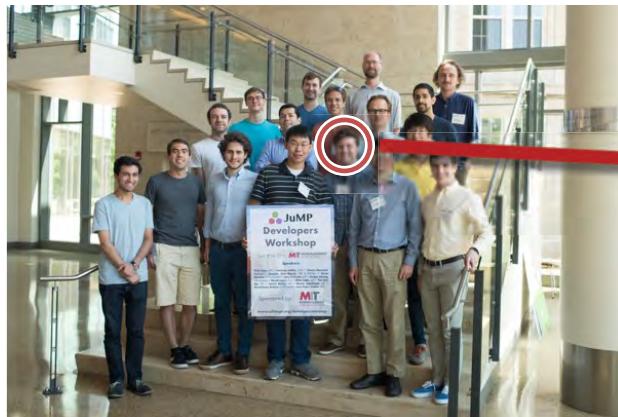
Peruvian Energy  
Ministry



# Advanced Network Science Initiative



- 34 repositories (<https://github.com/lanl-ansi>)
  - PowerModels.jl
  - GraphicalModelLearning.jl
  - Juniper.jl (MINLP solver)
  - GasGridModels.jl
- ...



## A Bit About LANL

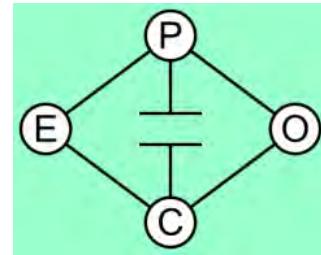


UNCLASSIFIED  
Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA  
LAUR-17-24522

**Carleton Coffrin**

# Milk Output Optimizer, or MOO

---



- From hydroelectric power to dairy farms:
  - rain, price of (milk/electricity) and substitutes (coal/corn)



**Oscar Dowson**

# Optimal Control Using Sum-of-Squares Optimization

- EntropicCone.jl
- SwitchOnSafety.jl
- **SumOfSquares.jl**
- Polyhedra.jl
- MultivariatePolynomials.jl



**Benoît Legat**

### Sum-of-Squares Programming in Julia with JuMP

Benoît Legat\*, Chris Coey<sup>†</sup>, Robin Deits<sup>†</sup>, Joey Huchette<sup>†</sup> and Amelia Perry<sup>†</sup>  
\* UCLouvain, <sup>†</sup> MIT

**Sum-of-Squares (SOS) Programming**

Two implementations: `TypedPolynomials.jl` and `DynamicalPolynomials.jl`. One common, independent interface: `MultivariatePolynomials.jl`.

```

sosvar y # one variable
sosvar x[1:2] # tuple/vector
sosvar x[1:2] # tuple/vector
Build a vector of monomials:
• (x1, x2, x3) → p(x) =  $\sum Q_x$  unique
x12 + 2x1x2 + 5x22 + 4x2x3 + x32 = xT [1 1 0]
[1 2 2] x
[0 2 1] x
p(x) ≥ 0 ⇔ x ≥ 0 cholesky
(x1 + x2)2 + (2x2 + x3)2 → xT [1 1 0] [1 1 0]
[0 2 1] x
[0 2 1] x

```

**Manipulating Polynomials**

Two implementations: `AbstractVectorFunction ∈ SOS(X)` (resp. `WSOS(X)`): SOS constraint without (resp. with) domain equipped with a bridge to `AbstractVectorFunction ∈ PSD` (resp. `SOS(X)`).

**Sum-of-Squares extension**

**MathOptInterface.jl (MOI)**  
MOI is an abstraction layer for mathematical optimization solvers. A constraint is defined by a "function" ∈ "set" pair.

**JuMP**  
JuMP is a domain-specific modeling language for mathematical optimization. It stores the problem directly (a cache can optionally be used) in the solver using MOI.

**JuMP extension**: `p(x) ≥ q(x)` and `p(x) ∈ SOS()` are converted into MOI SOS or WSOS constraints, e.g.  $x^2 - y^2 \geq 2xy$  is rewritten into  $[1, -2, 1] \in \text{SOS}(\omega^2, x, y)$ ,  $p(x) \in \text{WSOS}$  (resp. `SOS3`) is rewritten into linear (resp. second-order cone) constraints.

**SumOfSquares.jl**

**Bridging**

**SumOfSquares.jl** provides a bridge between **MathOptInterface.jl** and **JuMP**.

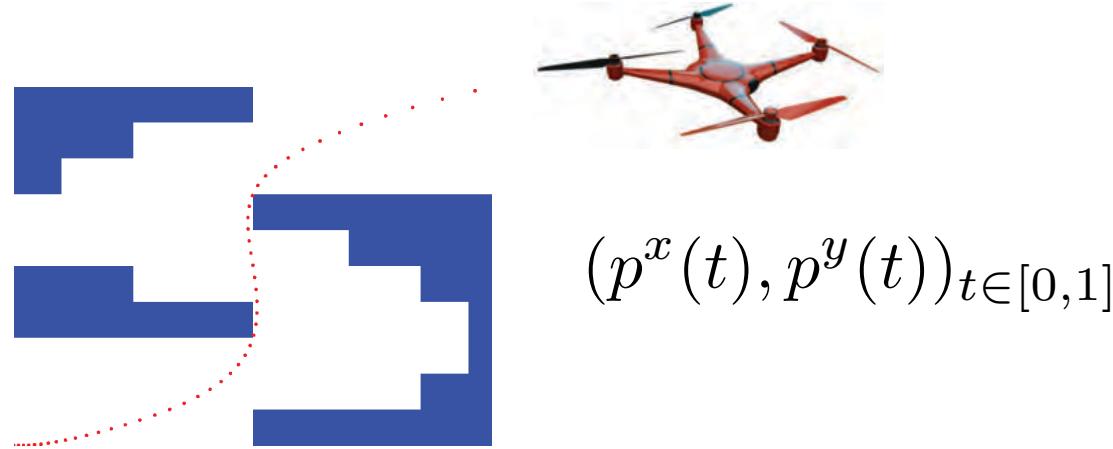
**Caching**

**Solvers**: Mosek, GSDP, SCS.

# Trajectory Planning with Collision Avoidance in a Week!

---

- Motivating: Steering a quadcopter through obstacles [Deits/Tedrake:2015]
  - ~2 week of work by Joey Huchette for SIOPT '17
- Position described by polynomials:



- Solution approach:
  - split domain into “safe” polyhedrons + discretize time into intervals

## Disjunctive *Polynomial* Optimization Formulation

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

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

$$\text{s.t. } p_1(0) = X_0, \ p'(0) = X'_0, \ p''(0) = X''_0 \quad \begin{matrix} \text{Initial/Terminal} \\ \text{Conditions} \end{matrix}$$

$$p_N(1) = X_f, \ p'_N(1) = X'_f, \ p''_N(1) = X''_f \quad \begin{matrix} \text{Initial/Terminal} \\ \text{Conditions} \end{matrix}$$

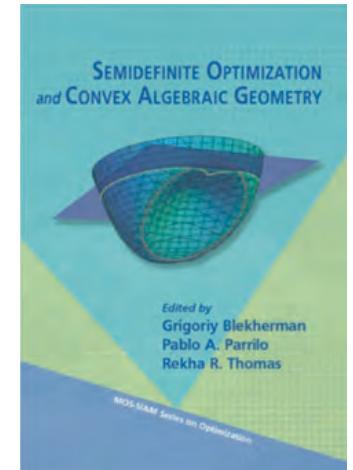
$$p_i(T_{i+1}) = p_{i+1}(T_{i+1}) \quad \forall i \in \{1, \dots, N-1\} \quad \text{Interstitial}$$

$$p'_i(T_{i+1}) = p'_{i+1}(T_{i+1}) \quad \forall i \in \{1, \dots, N-1\} \quad \text{Smoothing}$$

$$p''_i(T_{i+1}) = p''_{i+1}(T_{i+1}) \quad \forall i \in \{1, \dots, N-1\} \quad \text{Conditions}$$

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

MIP  
+



Avoid Collision = Remain in Safe Regions



```

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]] == Xe[ax])
    @constraint(model, differentiate(p[(ax,1)], t)([0], [t]) == Xe'[ax])
    @constraint(model, differentiate(p[(ax,1)], t, 2)([0], [t]) == Xe''[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])] == Xe[ax])
    @constraint(model, differentiate(p[(ax,N)], t)([1], [t]) == Xe'[ax])
    @constraint(model, differentiate(p[(ax,N)], t, 2)([1], [t]) == Xe''[ax])
end

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

```

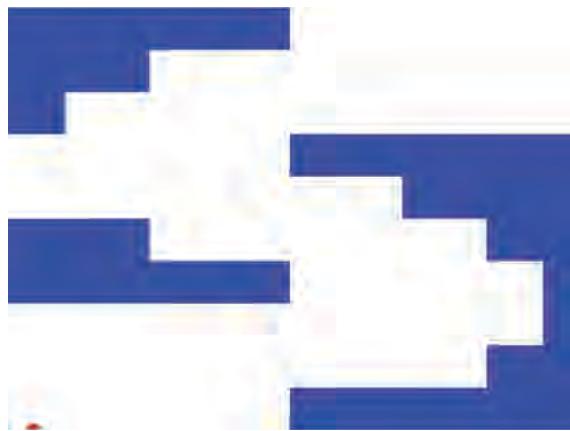
```

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

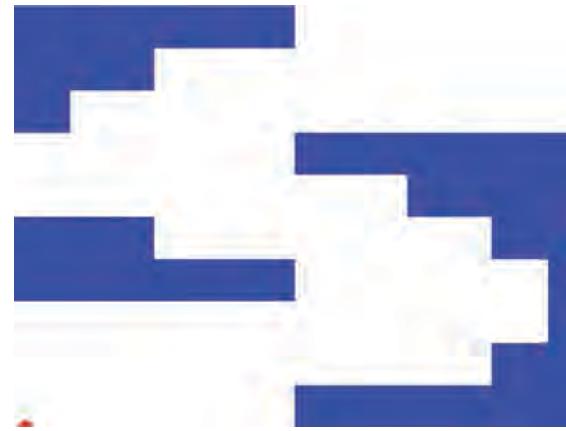
```

## Results for 9 Regions and 8 time steps

---



First Feasible Solution:  
58 seconds



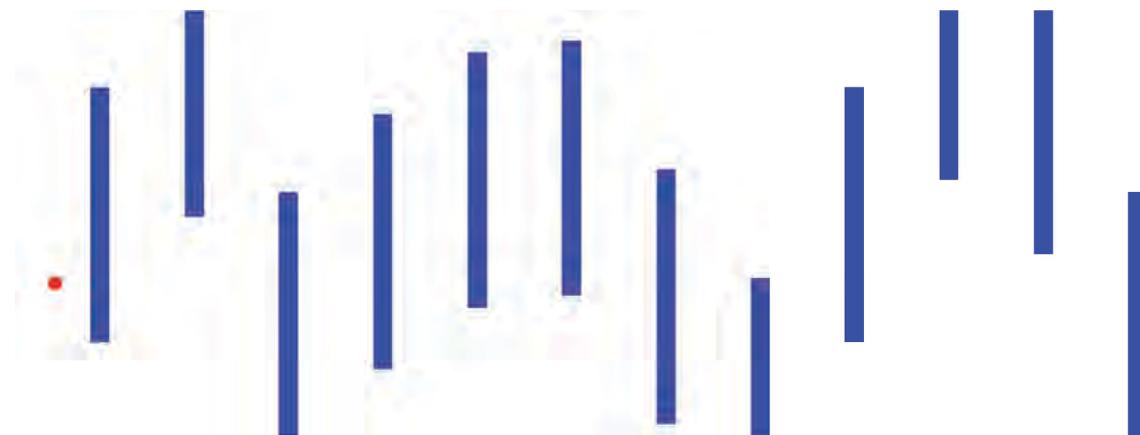
Optimal Solution:  
651 seconds

## Helicopter Game / Flappy Bird

---



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

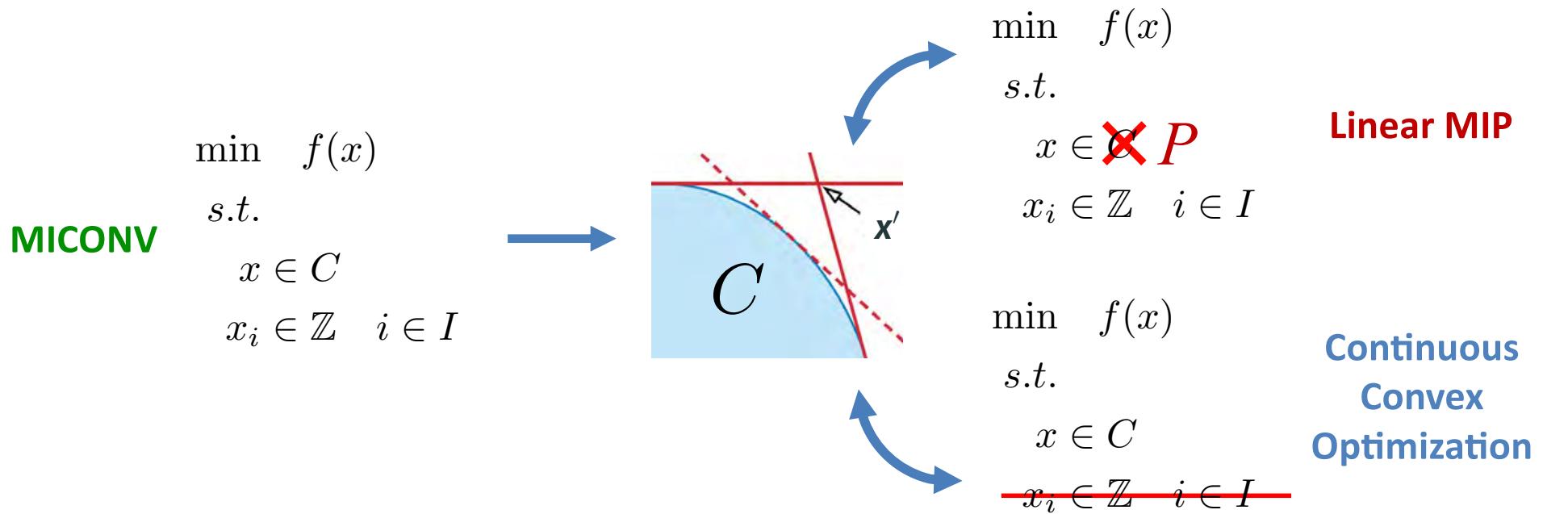


# How can we solve MICONV?

Joint work with Russell Bent, Chris Coey, Iain Dunning,  
Joey Huchette, Lea Kapelevich, Miles Lubin, Emre  
Yamangil, ...

# Polyhedral Outer-Approximation for MICONV

- Linear MIP and Continuous Convex Optimization Solvers > MICONV Solvers



$$C = \{\mathbf{x} \in \mathbb{R}^n : g_j(\mathbf{x}) \leq 0, \forall j \in [J]\}$$

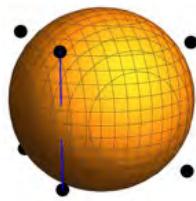
$$P = \{\mathbf{x} \in \mathbb{R}^n : g_j(\mathbf{x}') + \nabla g_j(\mathbf{x}')^\top (\mathbf{x} - \mathbf{x}') \leq 0, \forall \mathbf{x}' \in X_k, j \in [J]\}$$

# Improving OA Algorithms for MICONV

---

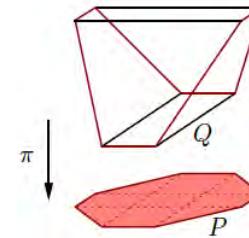
## Problem

1. May need large # of linear inequalities



## Solution

1. Use extended formulations



<https://rjlipton.wordpress.com>

2. MIP formulations can break gradient based continuous solvers
3. How to pick “good” linear inequalities

2. Use Conic Solver
3. Use Conic Duality

## Mixed Integer Conic Programming (MICP)

---

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

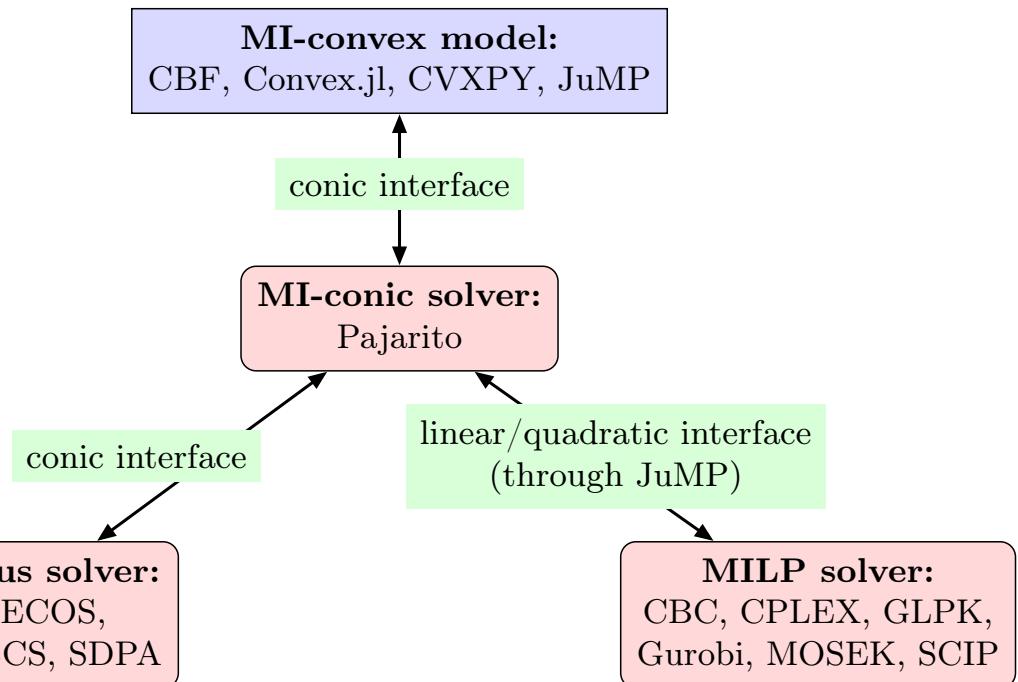
$$\mathbf{b}_k - \mathbf{A}_k \mathbf{x} \in \mathcal{C}_k \quad \forall k \in [M]$$
$$x_i \in \mathbb{Z} \quad \forall i \in [I]$$

- $\mathcal{C}_k$  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
  - SOCP disaggregation technique now standard (v., Dunning, Huchette, Lubin, 2015):

$$\|y\|_2 \leq y_0 \quad \longrightarrow \quad \sum_{i=1}^n z_i \leq y_0 \quad y_i^2 \leq z_i \cdot y_0 \quad \forall i \in [n]$$

# Pajarito: A Julia-based MICP Solver



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

## Performance for MISOCP Instances (120 from CBLIB)

---

		statuses				
solver		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
restricted	Pajarito-CBC-ECOS	78	30	3	9	163
	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

## Exponential Cone + LP / SOCP / SDP

$$x_1 \geq x_2 e^{x_3/x_2}, \quad x_1, x_2 > 0.$$

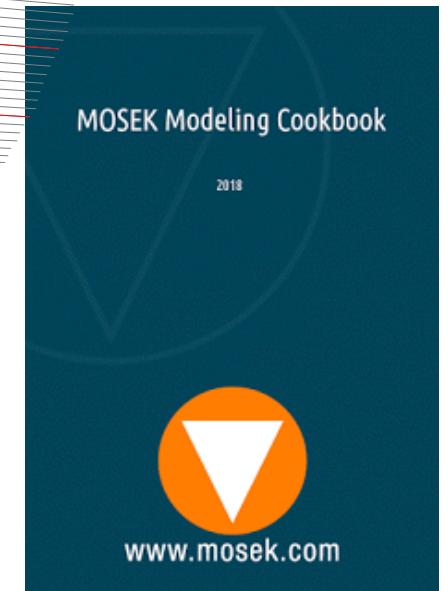
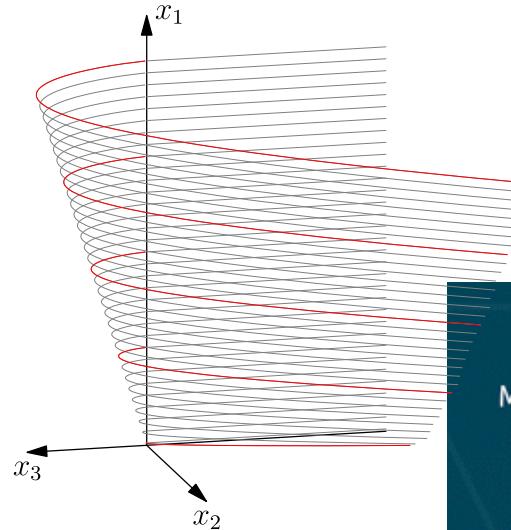
or

$$x_3 \leq x_2 \log(x_1/x_2), \quad x_1, x_2 > 0.$$

- Discrete experimental design

$$x \rightarrow \log \det \left( \sum_{i=1}^n x_i \mathbf{u}_i \mathbf{u}_i^T \right)$$

- Portfolio Optimization with entropic risk constraints
- All 333 MICONVs from MINLPLIB2
- Pajarito with SCS or Mosek (version 7.5.2)



<https://themosekblog.blogspot.com/2018/05/new-modeling-cookbook.html>

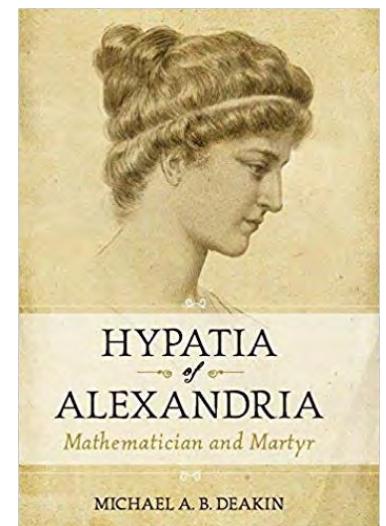
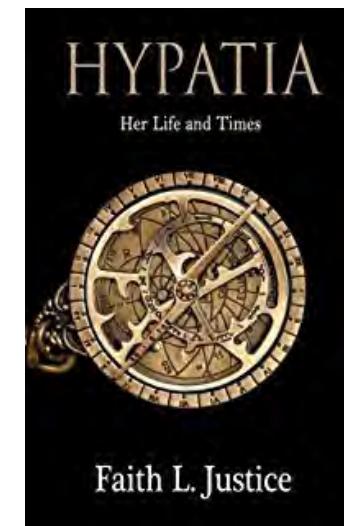
# Hypatia: Pure Julia-based IPM Beyond “Standard” Cones

---

- 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, RSOCOP, 3-dim exponential cone, PSD,  $L_\infty$ , 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

Linear Optimization

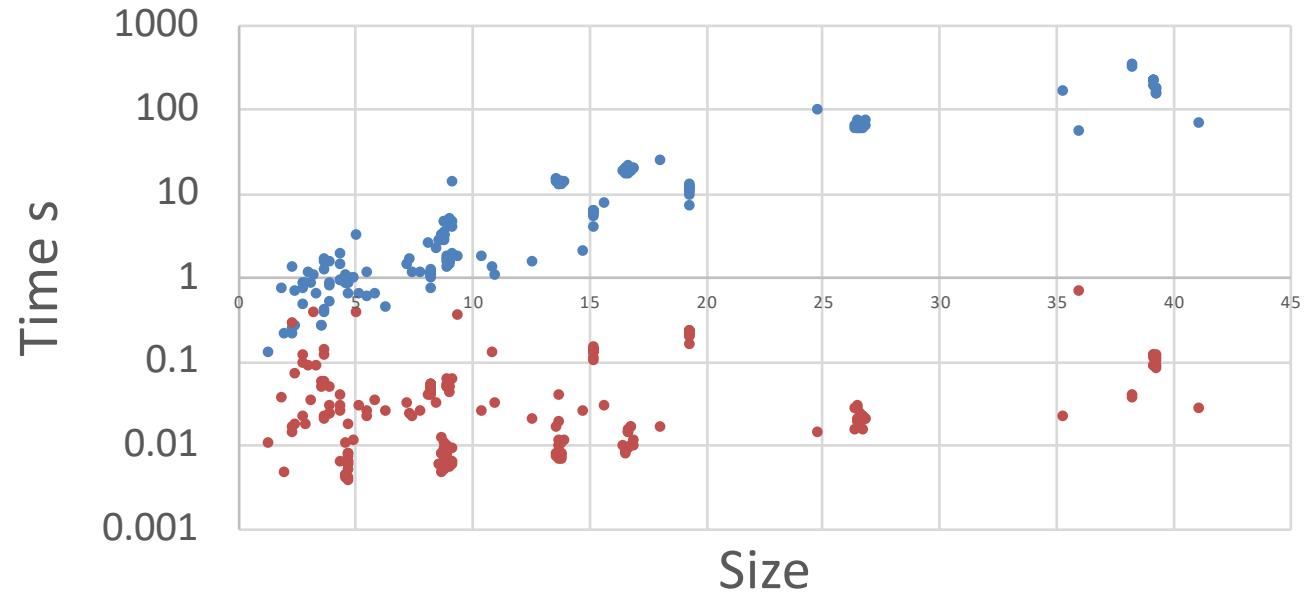
Polynomial Envelope

Polynomial  
Minimization

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

# First Batch of Tests on CBLIB Instances

---



- Instances:
  - SDP
  - SOCP
  - RSOCP
- Only 2 – 10K times slower than Mosek 8!
- `hypatia_time`
- `mosek_time`

## Summary

---

- MICONV can model many problems (but not all)
- How to solve MICONVs? Don't, solve MICPs
- Easy access to optimization modeling and solvers with JuMP
- Advanced solver development with Julia
- Disclaimers:
  - Julia just reached version 1 ( Yay! )
  - ... JuMP is undergoing a major redesign
    - Try in Julia 1.0 through "] add JuMP#v0.19-alpha"