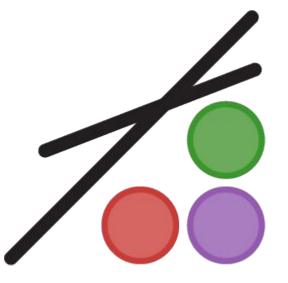


The Roadmap for JuMP 1.0

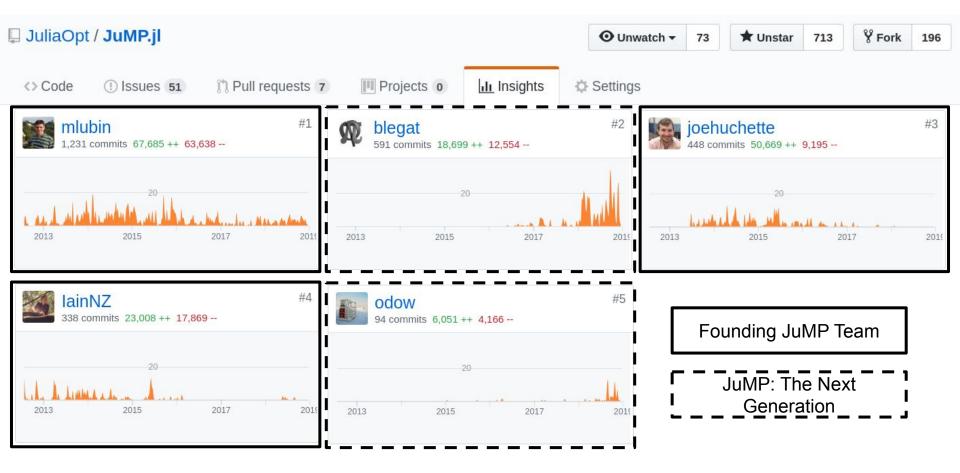
Miles Lubin Google¹ JuMP-dev 2019



¹JuMP is not a Google product.

Talk outline

- Introduction
- The 0.19 release
- The roadmap for JuMP 1.0



JuMP 0.19 contributors: Chris Coey, Carleton Coffrin, Joaquim Dias Garcia, Twan Koolen, Vitor Nesello, François Pacaud (Artelys), Robert Schwarz, Issam Tahiri, Juan Pablo Vielma, Ulf Worsøe (Mosek)

Pre-0.19 contributors: Carlo Baldassi, Jack Dunn, Jenny Hong, Steven G. Johnson, Tony Kelman, Dahua Lin, Karanveer Mohan, Yee Sian Ng, Elliot Saba, João Felipe Santos, Felipe Serrano, Madeleine Udell, and David Zeng

JuMP-dev 2019 local committee: Victor Albornoz, Gustavo Angulo, Rodolfo Carvajal, Diego Moran

What is JuMP?

JuMP is a modeling language for mathematical optimization, like:

AIMMS, AMPL, CMPL, CPLEX Concert, CVX, CVXPY, GAMS,

GNU MathProg, Gravity, Gurobi C++/Python API, LINGO,

MATLAB "problem-based workflow", Mosek Fusion, MOSEL, MPL, ompr,

OPTMODEL, PuLP, Pyomo, YALMIP, and ZIMPL.

Why is it interesting?

- Open source
- Embedded in Julia
- Low overhead for model generation
- Solver independent
- Easy to connect new solvers
- Extensible to new problem classes
- Supports interacting with solvers while they're running
- Active community

```
In [1]: using JuMP, GLPK
        struct Edge
            from
            to
            cost
            capacity
        end
        edges = [Edge(1, 2, 1, 0.5), Edge(1, 3, 2, 0.4), Edge(1, 4, 3, 0.6),
                 Edge(2, 5, 2, 0.3), Edge(3, 5, 2, 0.6), Edge(4, 5, 2, 0.5)]
        model = Model(with optimizer(GLPK.Optimizer))
        @variables model begin
            0 <= flow[e in edges] <= e.capacity
        end
        @constraints model begin
            sink constr, sum(flow[e] for e in edges if e.to == 5) == 1
            balance constr[node = 2:4], sum(flow[e] for e in edges if e.to == node) ==
                                        sum(flow[e] for e in edges if e.from == node)
        end
        @objective(model, Min, sum(e.cost * flow[e] for e in edges))
        optimize!(model)
        @show termination status(model)
        value.(flow)
        termination status(model) = OPTIMAL::TerminationStatusCode = 1
Out[1]: 1-dimensional DenseAxisArray{Float64,1,...} with index sets:
            Dimension 1, Edge[Edge(1, 2, 1, 0.5), Edge(1, 3, 2, 0.4), Edge(1, 4, 3, 0.6), Edge(2, 5, 2, 0.3), Edge(3, 5,
        2, 0.6), Edge(4, 5, 2, 0.5)]
        And data, a 6-element Array{Float64,1}:
         0.3
         0.4
         0.299999999999999993
         0.3
         0.4
         0.299999999999999993
```

Building up to JuMP 0.19

Version 0.2: Solver callbacks (December, 2013)

Version 0.5: Nonlinear optimization (May, 2014)

Version 0.10: Semidefinite optimization (August, 2015)

Version 0.12: Rewrote nonlinear optimization (February, 2016)

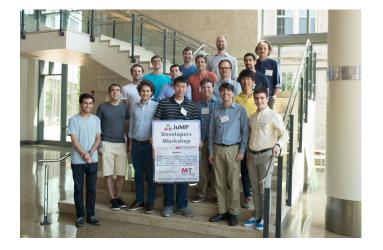
Version 0.13: Renamed everything from camelCase (April, 2016)

Version 0.15: sum { } becomes sum () (December, 2016)

JuMP (≤ 0.18) is great, but how do I ...

- add support for a new type of constraint?
- combine NLP constraints with conic constraints?
- delete a constraint or variable?
- know if my incremental changes are passed efficiently to the solver?
- modify coefficients in the constraint matrix?
- provide a dual warm-start?
- access the irreducible inconsistent subsystem (IIS) from Gurobi?
- distinguish between a solver that stopped because of the time limit 1) *with* a solution and 2) *without*?
- check if a solution is feasible?
- access multiple solutions found by a solver?

It took a few meetings to answer these questions...



JuMP-dev 2017 (MIT)



JuMP-dev 2018 (U. de Bordeaux)



JuMP @JuMPjl · Feb 17

We are thrilled to announce the release of **#JuMPjl** v0.19! This release includes many new features and improvements, and we are extremely grateful to the many contributors that made this release possible.

	Ē		We v0.1	JuMP v0.19 has been released We are thrilled to announce the release of JuMP v0.19. Work on this release started nearly two & discourse.julialang.org				
\mathcal{Q}		17	23	♡ 4	8	⚠		

What happened in JuMP 0.19?

- Replaced MathProgBase (MPB) with MathOptInterface (MOI)
- Introduced bridges (automatic constraint transformations)
- Redesigned the JuMP containers (DenseAxisArray and SparseAxisArray)
- Redesigned AffExpr and QuadExpr
- Rewrote documentation using docstrings and documenter
- Improved code quality and testing (e.g., the JuMP style guide)

See my talk from JuMP-dev 2018.

A few statistics on JuMP 0.19

- LOC in JuMP.jl: 11,408 added, 10,618 deleted, 2,858 modified, 1,520 unchg.
- LOC in MOI.jl: 15,593 added (50% is for testing)
- 13 major contributors
- 0 major bugs since the release
- 0.19 documentation: 10,000+ page views from 1,500+ unique users

LOC = lines of code

The roadmap for JuMP 1.0

What does 1.0 mean?

JuMP is stable and ready for use. We will follow <u>Semantic Versioning</u> for future breaking changes.

Who decides what's on the roadmap?

Core developers, by consensus.

Who is a core developer?

Good question. We may need to define more governing structures.

Major roadmap points

- Create a website for JuMP and deprecate JuliaOpt
- Deliver on promises from MOI
- Address major regressions from JuMP 0.18
- Address easy usability issues
- Documentation
- MOI 1.0 (will skip)

This roadmap will be posted in the JuMP documentation with corresponding issue references.

Create a website for JuMP and deprecate JuliaOpt

JuMP's new website will be hosted at jump.dev!

- Tell us what you want to see on it.
- We need help designing it. Funding is available.

The JuliaOpt brand and GitHub organization will be deprecated.

Deliver on promises from MOI

- "How do I check if a solution is feasible?"
- "How do I access the irreducible inconsistent subsystem (IIS)?"
- "How do I access multiple solutions found by a solver?"

Address major regressions from JuMP 0.18

- Model generation performance
- Callbacks
- Column generation
- Solvers that are still not compatible with MOI

Address easy usability issues

- Better error messages (e.g., correct line numbers)
- LP sensitivity summary
- Solver-independent options (e.g., verbosity)

This is where feedback can make a big difference. Let us know about weird error messages and missing functionality.

Documentation

Divio propose to separate documentation into:

- 1. Tutorials
- 2. How-to guides
- 3. Concept explanation
- 4. Technical reference

JuMP's documentation is mostly complete, but it's a jumble of (2)-(4). We'd like to improve it.

Past 1.0

- Unit testing for optimization models
- Make nonlinear programming first-class in JuMP and MOI
- Rewrite and clean up JuMP's macros
- Better support for variable transformations
- Generalize math with mutables (e.g., AffExpr)
- Constraint programming
- Disciplined convex programming (c.f. Convex.jl)

These are aspirational and may or may not happen without your help. Talk to us if you're interested.

How can you get involved?

- Funding is available through Google Summer of Code and NumFOCUS.
- Contribute code. Ask for feedback early in the <u>Gitter channel</u>.
- Contribute time. Post your code examples and answer questions on Discourse.
- Contribute \$ (via NumFOCUS).
- Give us your feedback! We take it seriously.
- See <u>CONTRIBUTING.md</u>.

Gracias y que disfruten de JuMP-dev!