JuMP: A Modeling Language for Mathematical Optimization

Miles Lubin, with Iain Dunning and Joey Huchette
MIT

- Military logistics (diet problem)
- Economic modeling (World Bank)
- Scheduling (nurses, trains, airline crews, power plants)
- Optimal control (autonomous vehicles)
- Statistics (regression/inference)

(JuMP is being used is some of these areas)

Linear Algebra

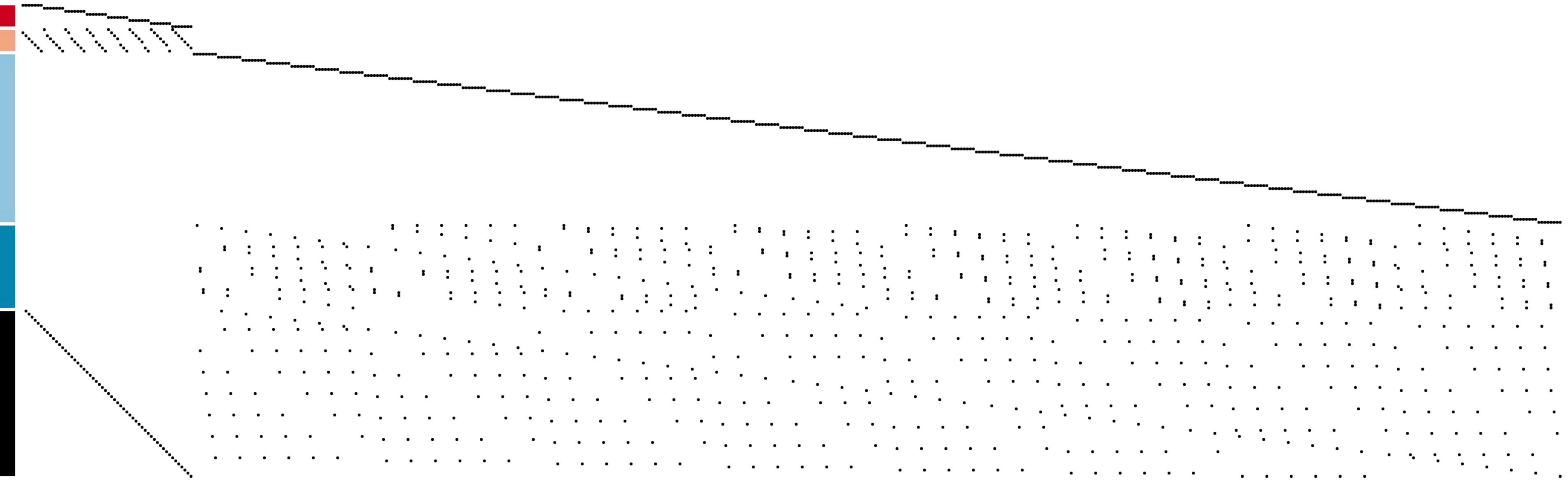
```
y = A\x
```

Optimization

```
Maximize     x_1 + x_2

Subject to:2x_1 + 3x_2 <= 2

            x_1        >= 0

                   x_2 >= 0
```

- **MATLAB** created in late 1970s to provide a user-friendly interface to LINPACK
- **AMPL**, **GAMS**, **LINDO** created in 1970s/1980s to provide a user-friendly interface to optimization *solvers*

Why do you need a modeling language?



Why do you need a modeling language?

$$\begin{aligned}
\min_x \quad & \sum_{(i,j)\in E} c_{i,j} x_{i,j} \\
\text{s.t.} \quad & \sum_{(i,j)\in E} x_{i,j} = \sum_{(j,k)\in E} x_{j,k} \quad j = 2, \ldots, n-1 \\
& \sum_{(i,n)\in E} x_{i,n} = 1 \\
& 0 \le x_{i,j} \le C_{i,j} \quad \forall (i,j) \in E
\end{aligned}$$

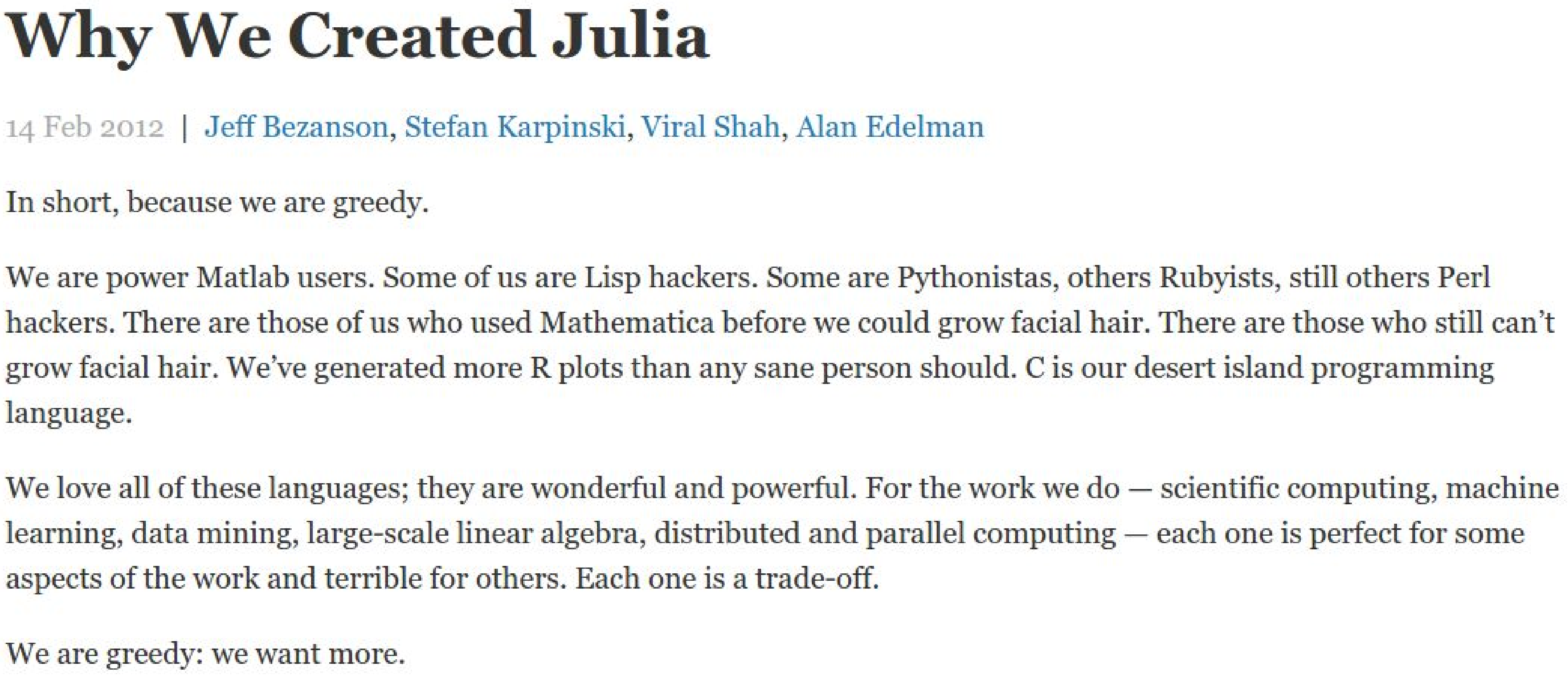
Why do you need a modeling language?

```
mcf = Model()

@variable(mcf, 0 <= flow[e in edges] <= e.capacity)

@constraint(mcf, sum(flow[e] for e in edges if e.to==5)== 1)

@constraint(mcf, flowcon[n=2:4],sum(flow[e] for e in edges
if e.to==node) == sum(flow[e] for e in edges if e.
from==node))

@objective(mcf, Min, sum(e.cost * flow[e] for e in edges))

solve(mcf)
```

What did we want?

- Modern, modular, easy to embed...
  - Within a simulation
  - Within an interactive visualization
- Interact with solvers while they are running
- Easy to extend to specialized problem classes
- Commercial tools are **none of these**
- Open source tools have been **slow**
  - Based on Python and MATLAB

julia



Why We Created Julia

14 Feb 2012 | Jeff Bezanson, Stefan Karpinski, Viral Shah, Alan Edelman

In short, because we are greedy.

We are power Matlab users. Some of us are Lisp hackers. Some are Pythonistas, others Rubyists, still others Perl hackers. There are those of us who used Mathematica before we could grow facial hair. There are those who still can't grow facial hair. We've generated more R plots than any sane person should. C is our desert island programming language.

We love all of these languages; they are wonderful and powerful. For the work we do — scientific computing, machine learning, data mining, large-scale linear algebra, distributed and parallel computing — each one is perfect for some aspects of the work and terrible for others. Each one is a trade-off.

We are greedy: we want more.

Can Julia solve our performance problem?

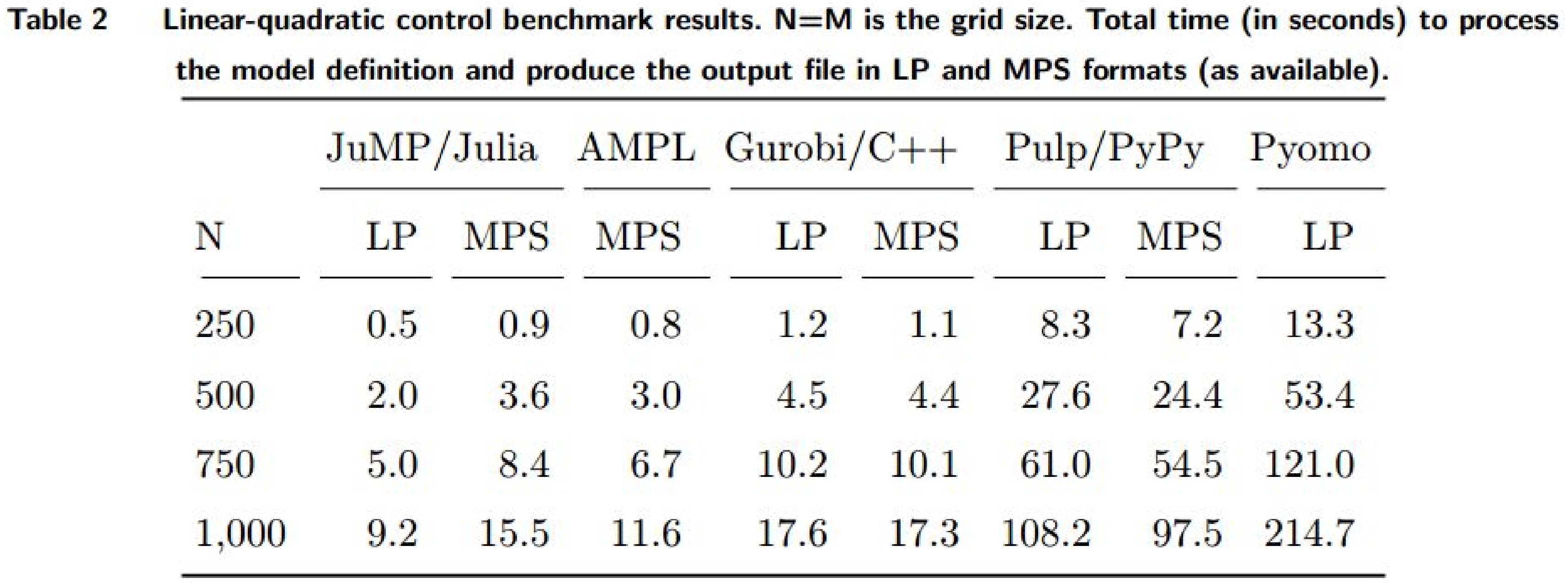


Table 2 **Linear-quadratic control benchmark results. N=M is the grid size. Total time (in seconds) to process the model definition and produce the output file in LP and MPS formats (as available).**

	JuMP/Julia		AMPL	Gurobi/C++		Pulp/PyPy		Pyomo
N	LP	MPS	MPS	LP	MPS	LP	MPS	LP
250	0.5	0.9	0.8	1.2	1.1	8.3	7.2	13.3
500	2.0	3.6	3.0	4.5	4.4	27.6	24.4	53.4
750	5.0	8.4	6.7	10.2	10.1	61.0	54.5	121.0
1,000	9.2	15.5	11.6	17.6	17.3	108.2	97.5	214.7

M. Lubin & I. Dunning, “Computing in Operations Research using Julia”, *INFORMS Journal on Computing*, 2015

JuMP history

Version 0.1: Initial public release (October, 2013)

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)

Who’s using it?

- Gonzales et al.: **Autonomous vehicles** (https://youtu.be/bX4TXWO7dA0)
- Korolko & Sahinoglu: **EV charging schedules**
- Das Gupta et al.: **Railway schedules**
- Giordano et al.: **Variational Bayes**
- Anthoff: **Environmental economics**
- Dvijotham et al.: **Graphical models & optimal power flow**

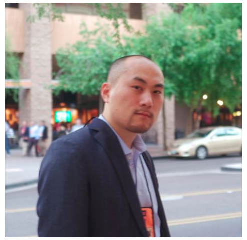
IOP Publishing Plasma Physics and Controlled Fusion

Plasma Phys. Control. Fusion **58** (2016) 045016 (16pp) doi:10.1088/0741-3335/58/4/045016

Inversion methods for fast-ion velocity-space tomography in fusion plasmas

A S Jacobsen[1], L Stagner[2], M Salewski[1], B Geiger[3], W W Heidbrink[2], S B Korsholm[1], F Leipold[1], S K Nielsen[1], J Rasmussen[1], M Stejner[1], H Thomsen[4], M Weiland[3] and the ASDEX Upgrade team[3]

[1] Department of Physics, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark
[2] University of California Irvine, Irvine, CA 92697, USA
[3] Max-Planck-Institute for Plasma Physics, Boltzmannstr. 2, 85748 Garching, Germany
[4] Max-Planck-Institute for Plasma Physics, Wendelsteinstrasse 1, 17491 Greifswald, Germany

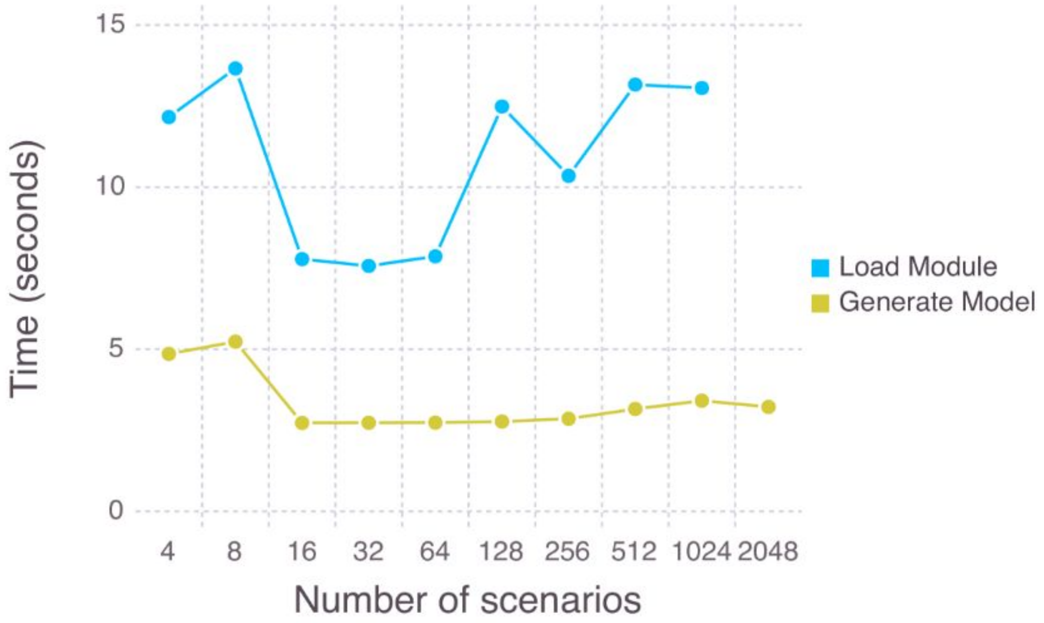


http://www.chkwon.net/julia/

Structured Modeling

$$\begin{array}{llllllll}
\min & c_0^T x + \sum_{i=1}^N c_i^T y_i & & & & & \\
\text{s.t.} & Dx & & & & & = b_0, \\
& T_1 x + & W_1 y_1 & & & & = b_1, \\
& T_2 x + & & W_2 y_2 & & & = b_2, \\
& \vdots & & & \ddots & & \vdots \\
& T_N x + & & & & W_N y_N & = b_N, \\
& x \geq 0, & y_1 \geq 0, & y_2 \geq 0, & \ldots, & y_N \geq 0. &
\end{array}$$

StructJuMP



- Huchette, Petra, Qiang
- Interacts with MPI-based solvers
- Weak scaling test on Blues cluster at Argonne. 1 scenario = 1 core
- Model generation <2% of total solution time

JuMPChance

```
@indepnormal(m, x, mean=0,var=1)
@variable(m, z)

@constraint(m,
            z*x >= -1, with_probability=0.95)
```

Used with Dvorkin, and Backhaus to study a model of integrating wind energy with the power grid (IEEE, 2015)

Thank you!

- CSGF
- Julia developers and community

http://arxiv.org/abs/1508.01982