Operations research with Julia and JuMP

Pedro Belin Castellucci

February, 2017

Online material at github.com/pedrocastellucci/athena.

Mathematical programming

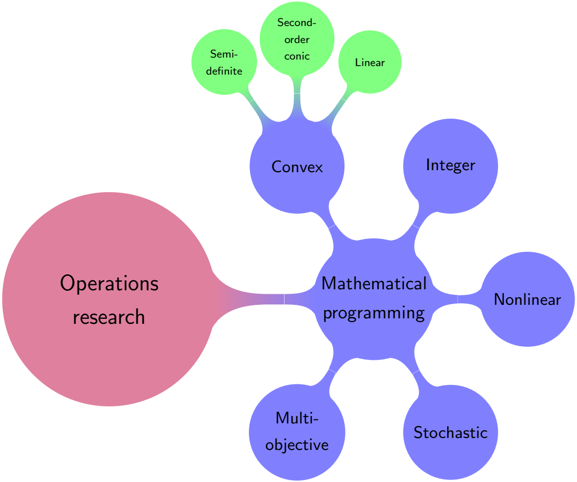

Figure 1: Information from https://en.wikipedia.org/wiki/Mathematical_optimization.

Which can JuMP handle?

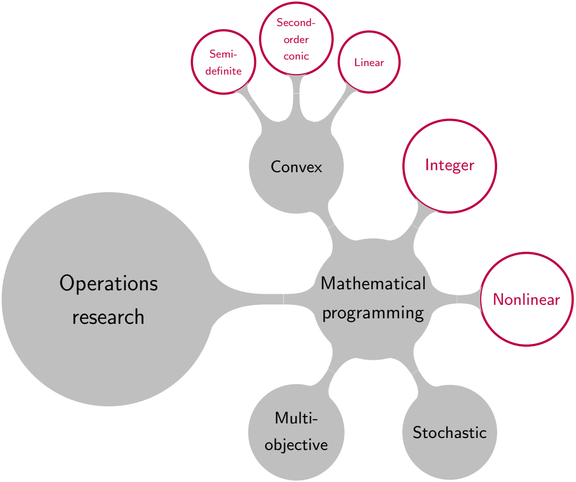

Figure 2: For JuMP documentation go to
http://www.juliaopt.org/JuMP.jl/0.15/.

What will we do?

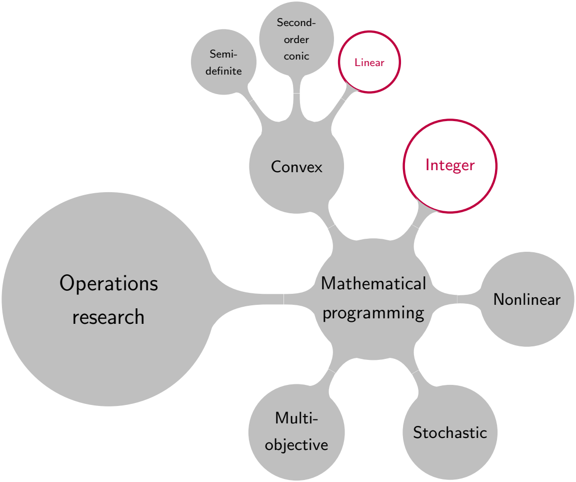

Figure 3: For JuMP documentation go to `http://www.juliaopt.org/JuMP.jl/0.15/`.

What is mixed linear integer optimization?

What is mixed linear integer optimization?

$$\text{Min } c^T(x+y),$$

subject to:

$$Ax + By = D,$$

$$x \in \mathbb{R}^n,$$

$$y \in \mathbb{Z}^m.$$

What is mixed linear integer optimization?

Min $c^T(x+y)$,

Optimization criteria

subject to:

$$Ax + By = D,$$

$$x \in \mathbb{R}^n,$$

$$y \in \mathbb{Z}^m.$$

What is mixed linear integer optimization?

$$\text{Min } c^T(x+y),$$

Optimization criteria

subject to:

$$Ax + By = D,$$

Technological constraints

$$x \in \mathbb{R}^n,$$

$$y \in \mathbb{Z}^m.$$

What is mixed linear integer optimization?

Min $c^T(x+y)$, Optimization criteria

subject to:

$Ax + By = D,$ Technological constraints

$x \in \mathbb{R}^n,$ Real variables

$y \in \mathbb{Z}^m.$

What is mixed linear integer optimization?

Min $c^T(x+y)$, → Optimization criteria

subject to:

$Ax + By = D$, → Technological constraints

$x \in \mathbb{R}^n$, → Real variables

$y \in \mathbb{Z}^m$. → Integer variables

An example

$$\text{Max } x + y,$$

subject to:

$$2x + 3y \leq 10,$$

$$3x + 2y \leq 10,$$

$$x \in \mathbb{R}_+,$$

$$y \in \mathbb{Z}_+.$$

An example

$$\text{Max } x + y,$$ Optimization criteria

subject to:

$$2x + 3y \leq 10,$$

$$3x + 2y \leq 10,$$ Technological constraints

$$x \in \mathbb{R}_+,$$ Real variable

$$y \in \mathbb{Z}_+.$$ Integer variable

JuMP - Julia for Mathematical Optimization

JuMP - Julia for Mathematical Optimization

Domain-specific modeling language.

JuMP - Julia for Mathematical Optimization

Domain-specific modeling language.

User friendliness.

JuMP - Julia for Mathematical Optimization

Domain-specific modeling language.

User friendliness.

Speed:

- Creates problems at similar speed of other modeling languages (e. g. AMPL).
- Communicates with solver in memory.

JuMP - Julia for Mathematical Optimization

- Domain-specific modeling language.
- User friendliness.
- Speed:
  - Creates problems at similar speed of other modeling languages (e. g. AMPL).
  - Communicates with solver in memory.
- Solver independence:
  - Current supports Artelys Knitro, Bonmin, Cbc, Clp, Couenne, CPLEX, ECOS, FICO Xpress, GLPK, Gurobi, Ipopt, MOSEK, NLopt, and SCS.


TIME FOR A LIVE DEMO
WHAT COULD GO WRONG?
memegenerator.net

Exercise - The knapsack problem

Let $i \in I$ be an item with value v_i and weight c_i. We want to choose the most valuable subset of items to carry in a knaspack without violating its capacity C. Let $x_i \in \{0, 1\}$, $i \in I$, indicate whether item i is put into the knapsack. The following integer program solves our knapsack problem.

$$Max \sum_{i \in I} v_i x_i : \sum_{i \in I} c_i x_i \leq C,\ x_i \in \{0, 1\}, i \in I.$$

1. Solve the instance on the file example.dat.
2. As output, provide how many and which items were taken and their aggregated value.

Exercise - Scheduling TV commercials

You are in charge of a scheduling commercials during a TV show. By contract, you must run all the commercials $c \in C$ during the show. Each commercial c has the duration $t_c \leq 3$, $c \in C$, in minutes. The show may have any number of intervals, however, you know that there might be an audience drop for every interval. Also, to prevent audience loss, intervals must not be greater than 3 minutes.

Exercise - Scheduling TV commercials

Let $x_{ic} \in \{0, 1\}$ indicate whether commercial c is schedule to interval i. The following integer program solves the problem.

$$Min \sum_{i=1}^{|C|} y_i,$$

subject to:

$$\sum_{c \in C} t_c x_{ic} \leq 3y_i \quad i \in \{1, \dots, |C|\},$$

$$\sum_{i=1}^{|C|} x_{ic} = 1, \qquad c \in C,$$

$$y_i \in \{0, 1\}, \qquad i \in \{1, \dots, |C|\},$$

$$x_{ic} \in \{0, 1\}, \qquad i \in \{1, \dots, |C|\}, c \in C.$$

Implement the model and solve it using random input data.

One step further – callbacks

Lazy constraints.

User cuts.

User heuristics.

Solver progress.

Informational.

More info at: https://jump.readthedocs.io/en/latest/callbacks.html

Lazy constraints

Called when new solutions are found.

Code:

```
function myLazyConstraint(cb)
    ...
    @lazyconstraint(cb, myconstraint, localcut=false)
end
addlazycallback(m, myLazyConstraint)
solve(m)
```

User cuts

Called when solver reaches a new node in the branch-and-bound tree.

Code:

```
function myUserCut(cb)
    ...
    @usercut(cb, myconstraint, localcut=false)
end
addcutcallback(m, myUserCut)
solve(m)
```

User heuristics

Create solutions and submit them back to the solver.

Code:

```
function myHeuristic(cb)
    ...
    setsolutionvalue(cb, x, value)
    addsolution(cb)
end
addheuristiccallback(m, myHeuristic)
solve(m)
```