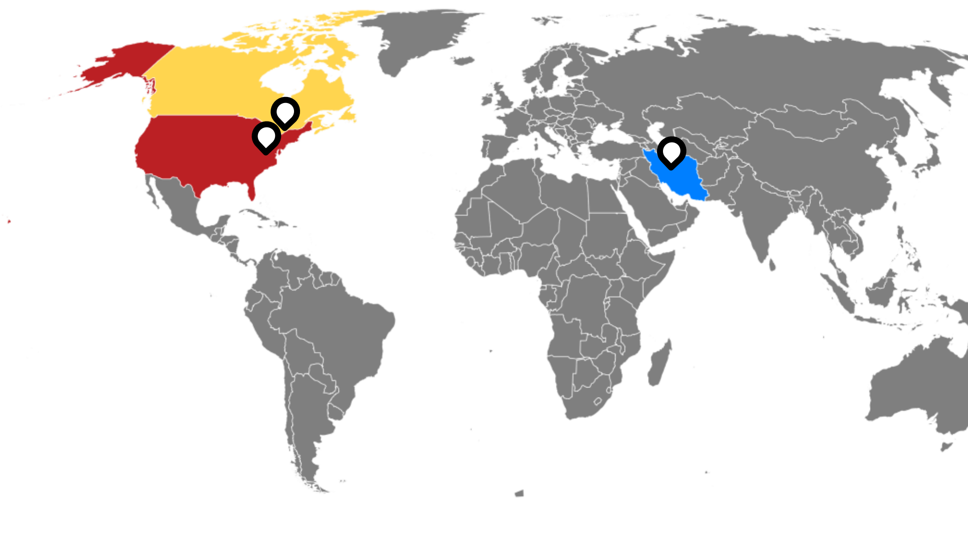
Julia for Mathematical Programming (JuMP)

Milad Dehghani Filabadi
PhD. Candidate
ISE, OSU
March 2024

Education Background

B.Sc. in Industrial Engineering

Specializations:
- Statistics, Operations research
- Reliability Analysis in Transportation Networks



M.Sc. in Management Sciences

Specialization:
- Optimization under uncertainty
- Power system management and economy

Ph.D. in Industrial Engineering, exc. Spring 2024

Specialization:
- Mixed-integer programming
- Computational analysis
- Reliability analysis, Statistical learning

OR and Data Scientist Intern

- Data cleaning
- Machine learning algorithms
- Developing forecast algorithms
- Optimization modeling

Research Background

THE OHIO STATE UNIVERSITY

Milad Dehghani Filabadi

The
Ohio State University

Cutting Plane Algorithms
Mixed-Integer Programming
Polynomial Optimization
Robust Optimization
Power System Optimization

	All	Since 2019
Citations	90	90
h-index	6	6
i10-index	5	5

0 articles	1 article
not available	available

Based on funding mandates

TITLE	CITED BY	YEAR	
Robust optimisation framework for SCED problem in mixed AC-HVDC power systems with wind uncertainty MD Filabadi, SP Azad IET Renewable Power Generation 14 (14), 2563 – 2572	24	2020	C++
Effective budget of uncertainty for classes of robust optimization M Dehghani Filabadi, H Mahmoudzadeh INFORMS Journal on Optimization 4 (3), 249-277	22	2022	C++
A new stochastic model for bus rapid transit scheduling with uncertainty MD Filabadi, A Asadi, R Giahi, AT Ardakani, A Azadeh Future Transportation 2 (1), 165-183	14	2022	MATLAB
Robust-and-cheap framework for network resilience: A novel mixed-integer formulation and solution method MD Filabadi arXiv preprint arXiv:2110.09694	12	2021	Julia
Robust optimization for SCED in AC-HVDC power systems M Dehghani Filabadi University of Waterloo	12	2019	C++
A new paradigm in addressing data uncertainty: Discussion and future research M Dehghani Filabadi Acad Lett 2, 4775	6	2022	Python/Julia
Mixed-integer exponential conic optimization for reliability enhancement of power distribution systems MD Filabadi, C Chen, A Conejo Optimization and Engineering, 1-27		2023	Julia

Working papers implemented in Julia

- Filabadi, M. D., Chen, C.. (2024a). An Exponential Conic Programming Relaxation for Signomial Programs.
- Filabadi, M. D., Chen, C., & Conejo, A. (2024b). Mixed-Integer Exponential Conic Relaxation for Optimal Power-Gas Problem.

Why Julia for Mathematical Programming?

- Julia is a high-performance programming language known for its:
  - Speed (near C++ performance)
  - Ease of use (is user-friendly such as Python or MATLAB)
  - Very powerful for computational research

- Julia for Mathematical Programming (JuMP):
  - simplifies the formulation and solution of mathematical optimization problems.
  - provides a convenient syntax for defining optimization variables

- Julia community is very active and growing, ensuring continuous development and support for mathematical programming tasks and packages.

Julia Installation

❑Download and Install Julia

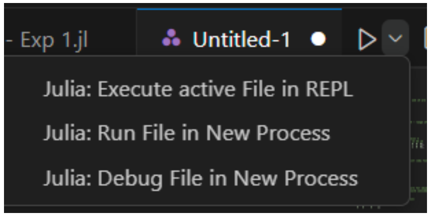
- ✓ Step 1: Visit the JuliaLang website (https://julialang.org/).
- ✓ Step 2: Run the downloaded installer.

❑Setting Up Visual Studio Code

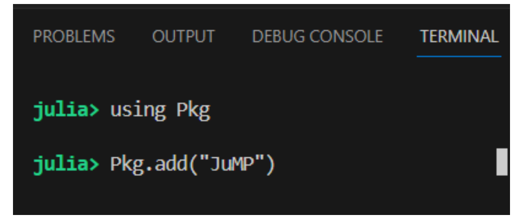
- ✓ Step 1 Download and install Visual Studio Code from https://code.visualstudio.com/
- ✓ Step 2: Open Visual Studio Code and navigate to the Extensions view (**Ctrl+Shift+X**).
- ✓ Step 3: Search for "Julia" and click "Install" on the Julia extension by julialang.

❑Open Julia REPL in Visual Studio Code

- ✓ Navigate the Julia REPL from the top-right icon
- ✓ Click on “Julia: Execute active File in REPL”

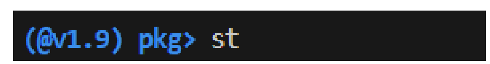


- ❑ Add Required Packages: After installation and opening a Julia REPL, install packages in Terminal:



- ❑ Open package environment: Type] in the Terminal.

- ❑ Check Installed Packages: Type st in package environment



- ❑ Exit Package environment: Type **Ctrl+C**.

❑Required Packages for Optimization:

- ❑JuMP
- ❑MathOptInterface
- ❑Solver: Gurobi, Mosek, or your preferred solver
  - ➢ You need to download the solver and install it first
- ❑MosekTools for special type of problems: Conic programming
- ❑Plots for visualizations

My packages:

```
(@v1.9) pkg> st
Status `C:\Users\gospw\.julia\environments\v1.9\Project.toml`
^ [4076af6c] JuMP v1.17.0
^ [b8f27783] MathOptInterface v1.23.0
^ [6405355b] Mosek v10.1.3
  [1ec41992] MosekTools v0.15.1
⌅ [91a5bcdd] Plots v1.39.0
  [24249f21] SymPy v2.0.1
```

Example 1:

$$
\begin{aligned}
Max \quad & 8x_1 + 10x_2 + 9x_3 \\
s.t. \quad & x_1 + 3x_2 + 2x_3 \leq 14 \\
& x_1 + 5x_2 + 3x_3 \leq 12.5 \\
& x_1, x_2, x_3 \geq 0
\end{aligned}
$$

$$
\begin{aligned}
Max \quad & cx \\
s.t. \quad & Ax \leq b \\
& x \geq 0
\end{aligned}
$$

$$A = \begin{bmatrix} 1 & 3 & 2 \\ 1 & 5 & 3 \end{bmatrix}$$

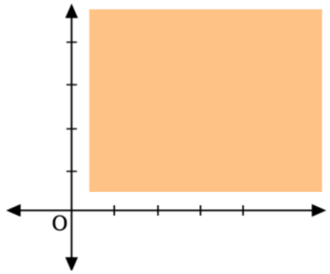
$$c = (8,10,9)$$

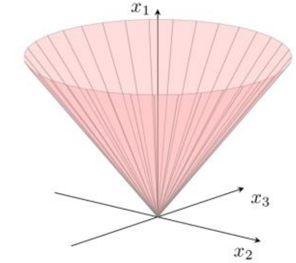
$$b = (14, 12.5)^T$$

Conic Programming: An extension of linear programming

$$\min\ c^T x$$
$$Ax \leq b$$
$$x \in K$$

where K is a convex cone.

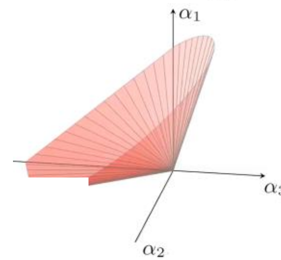




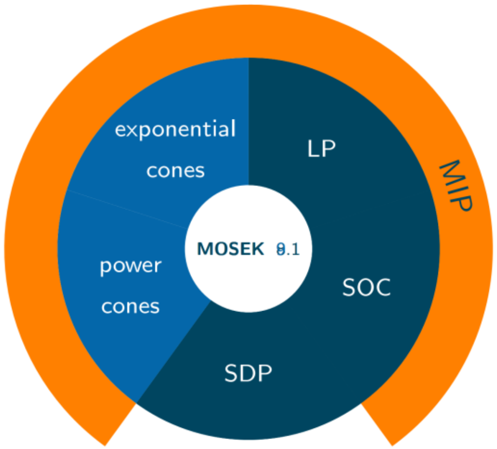
1. Non-negative orthant: $K_N = \{x \in R^n:\ x_i \geq 0, i = 1, \ldots, n\}$

2. Second-order cone: $K_{SOC} = \{x \in R^n:\ x_1^2 + \cdots + x_{n-1}^2 \leq x_n^2\}$

3. Exponential cone: $K_{\exp} = cl\{x \in R^3:\ x_2 e^{\frac{x_1}{x_2}} \leq x_3\}$



Mosek is the only solver to solve problems with K_{exp}



Example 2:

$$K_{SOC} = \{x \in R^n:\ x_1^2 + \cdots + x_{n-1}^2 \leq x_n^2\}$$

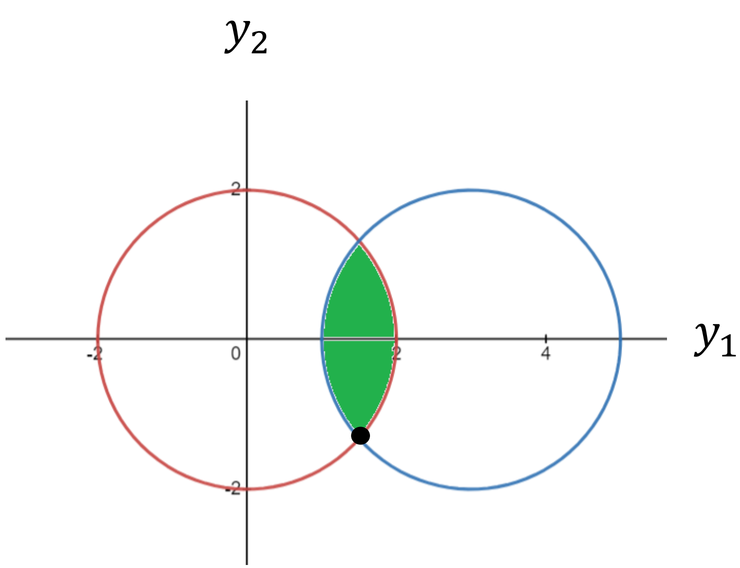
Implementation → $[x_n; x_1; \ldots; x_{n-1}] \in K_{SOC}$

$$Min \quad y_1 + y_2$$

$$s.t. \quad y_1^2 + y_2^2 \leq 4 \quad \longrightarrow \quad [2;\ y_1; y_2] \in K_{SOC}$$

$$(y_1 - 3)^2 + y_2^2 \leq 4 \quad \longrightarrow \quad [2;\ y_1 - 3; y_2] \in K_{SOC}$$

$$y_1, y_2 \text{ free}$$



$$(y_1^*, y_2^*) = (1.5, -1.3229)$$

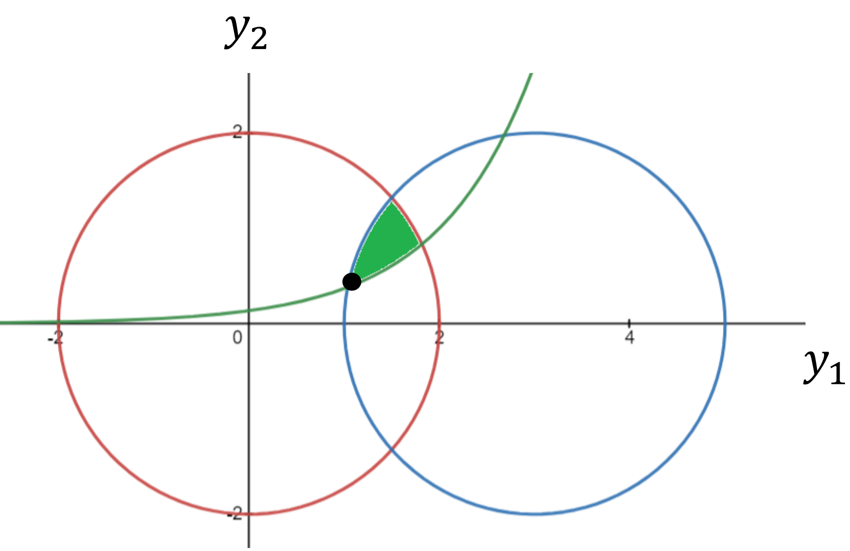
Example 3:

$$K_{\exp} = cl\{x \in R^3: \; x_2 e^{\frac{x_1}{x_2}} \le x_3\}$$

Implementation

$$[x_1; x_2; \; x_3] \in K_{\exp}$$

Min	$y_1 + y_2$	
$s.t.$	$y_1^2 + y_2^2 \le 4$	$[2; \; x_1 \, ; x_2] \in \; K_{SOC}$
	$(y_1 - 3)^2 + y_2^2 \le 4$	$[2; \; x_1 - 3 \, ; x_2] \in \; K_{SOC}$
	$e^{y_1 - 2} \le y_2$	$[y_1 - 2; 1; \; y_2] \in K_{\exp}$
	y_1, y_2 free	



$$(y_1^*, y_2^*) = (1.036, 0.3816)$$

❑Reading materials:

- ✓ Mosek website: https://docs.mosek.com/modeling-cookbook/index.html
- ✓ Mosek cookbook: https://docs.mosek.com/MOSEKModelingCookbook-v
- ✓ My sample codes on github: https://github.com/miladdf94/Julia_Examp

Thank you