

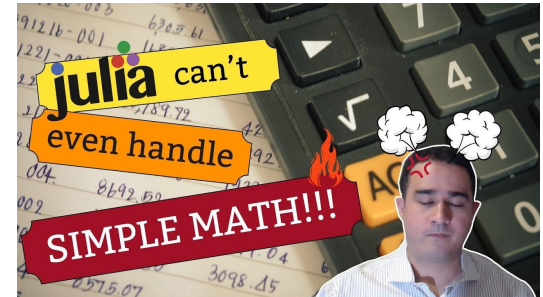
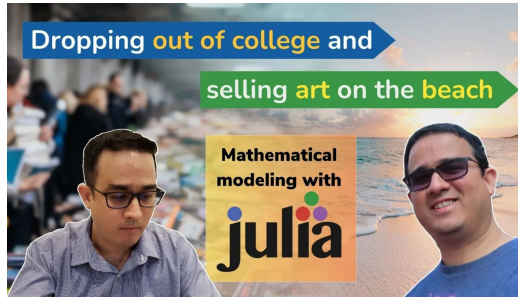
Julia Smooth Optimizers

ISO-compliant solvers,
news, and current challenges
@Julia Opt Days Paris 2023

Abel Soares Siqueira

Brief Personal Introduction

- Brazilian, Former professor (2014-2021)
- Sr Research Software Engineer - since 2021 @ Netherlands eScience Center
- YouTuber - since 2020
 - @AbelSiqueira
- <https://abelsiqueira.com>



Netherlands eScience Center

- ~80 Research Software Engineers
- Funded by gov., provide in-kind support to researchers in NL
- Various kinds of projects
 - Typescript + Vue + Quasar, bit of C++, Python (Graph NN), Julia/JuMP for energy models
- Hiring again soon





JuliaSmoothOptimizers

60+ repositories

eScience center

(Abel Siqueira)

PartitionedArrays.jl



Gridap.jl



gridap

NamedArrays.jl

JuliaCon Local Eindhoven 2023

Eindhoven, Netherlands. December 1, 2023.
Co-host with PyData Eindhoven 2023

Smart Wind

(Uwe Fechner)

KiteSimulators.jl

KiteViewers.jl

KiteModels.jl

ConformalPre

Lapla

CounterfactualExpla

Delta

JuliaHub

(Tim Besard)

CUDA.jl



LLVM.jl



oneAPI.jl



Wflow.jl

BEAST.jl

CxxWrap.jl

(Bart Janssens)

Pluto.jl

(Fons van der Plas)

Ghent

Brussels

Louvain-la-Neuve

Electa

FlexPlan.jl

PowerModels

Genk

Liege

Gmsh.jl

Eindhoven

Jokes.jl

(Gareth Thomas)

BIASlab



RxInfer.jl



Rocket.jl



ForneyLab.jl

Kroki Kroki.jl

JuliaHub

(Joris Kraak)

ASML Julia in production with more than 150 private packages



EindhovenLogo.jl

(yes, there is a package to draw a logo of Eindhoven)



PPTX.jl

WeatherReport.jl

CryptoDashApp.jl



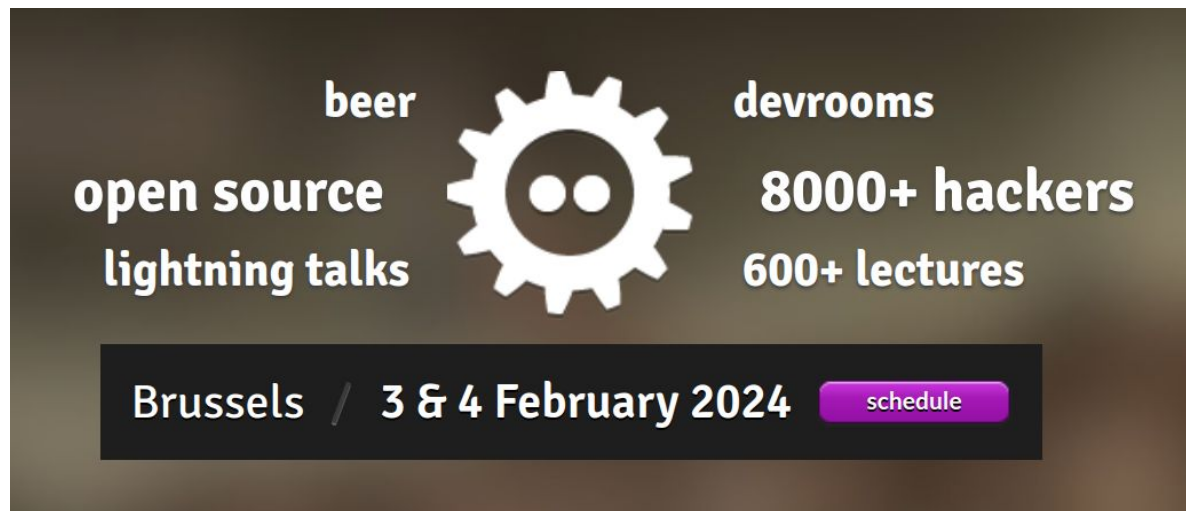
julia in the region* eindhoven

*many packages are international :)

Send us a message if you want to add something to the list!

FOSDEM - Julia dev room?

- Still applying for it
- Luca Ferranti (@lucaferranti on Slack)



A promotional graphic for FOSDEM 2024. It features a central gear icon with two dots inside, representing a face. The text is arranged around the gear. On the left, it says "beer", "open source", and "lightning talks". On the right, it says "devrooms", "8000+ hackers", and "600+ lectures". At the bottom, it says "Brussels / 3 & 4 February 2024" and includes a purple button labeled "schedule".

beer

open source

lightning talks

devrooms

8000+ hackers

600+ lectures

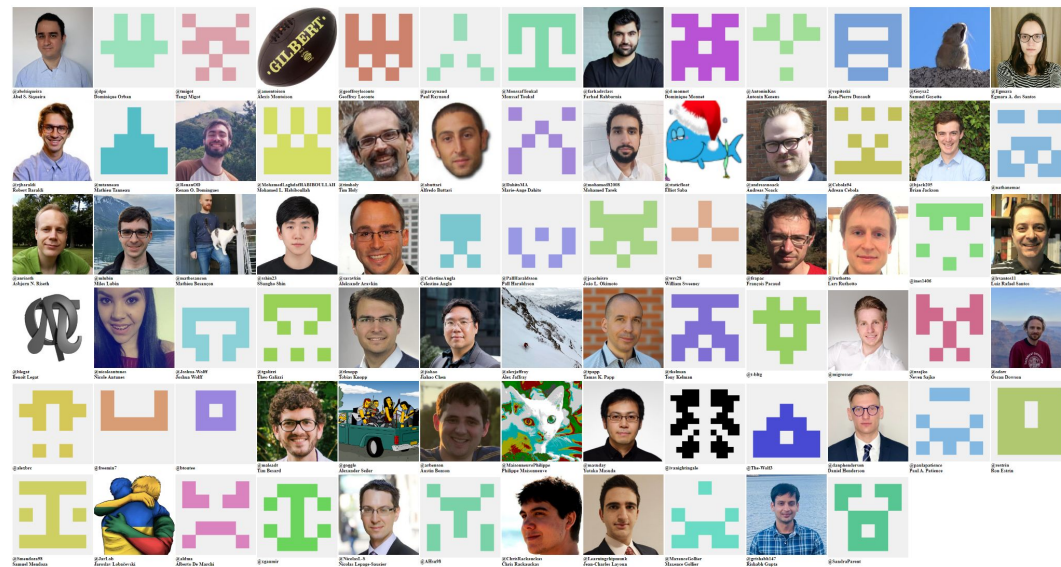
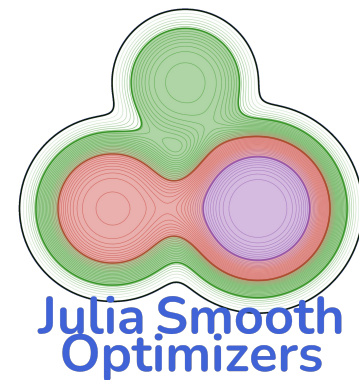
Brussels / 3 & 4 February 2024

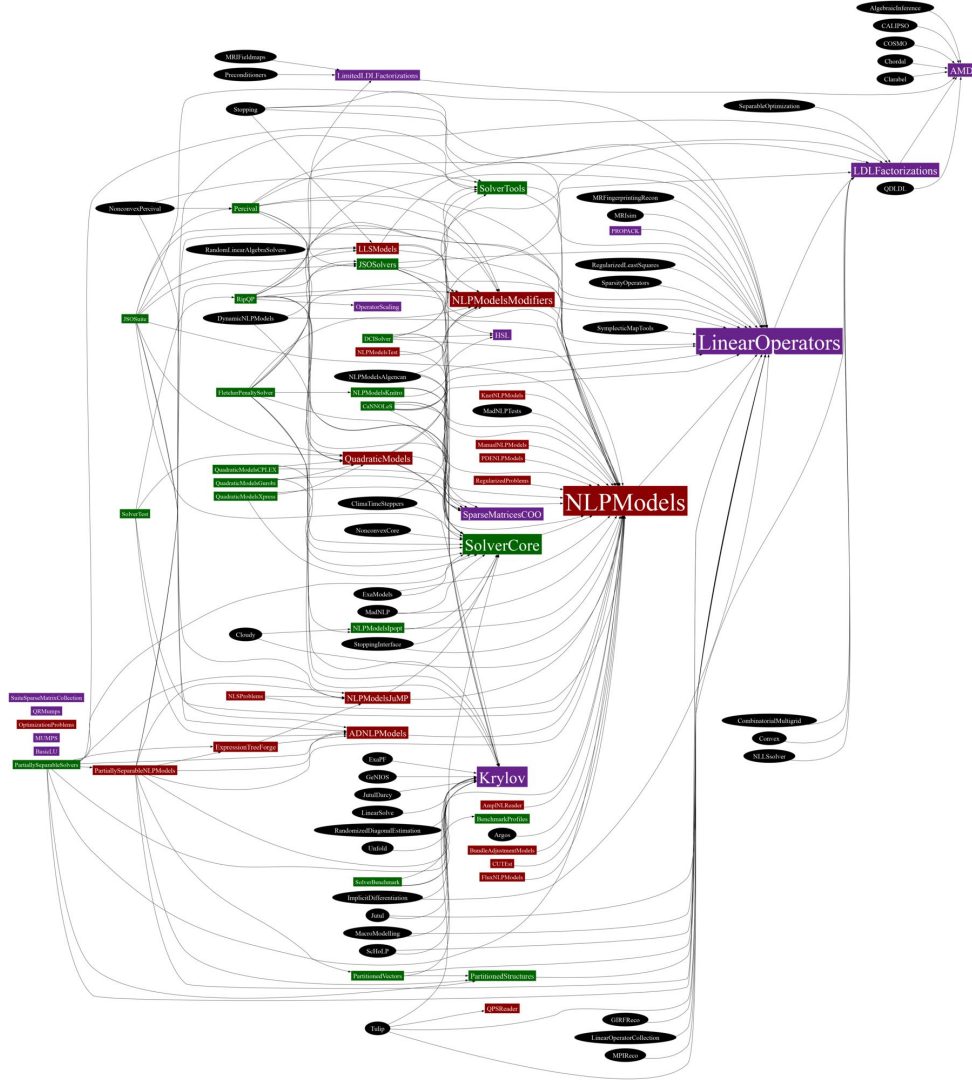
[schedule](#)



Julia Smooth Optimizers

- Over 50 pkg, 1000 stars, 75 contributors, 10k commits
- Started in **2015** with CUTEst.jl
- Research oriented
- Optimization and Linear Algebra





Nonlinear optimization

$$\begin{array}{ll} \min & f(x) \\ \text{s. to} & c_L \leq c(x) \leq c_U \\ & \underline{\ell} \leq x \leq \underline{u} \end{array}$$

$$f(x)$$

$$c(x)$$

$$\nabla f(x)$$

$$J(x) = [\nabla c_i(x)^T]_i$$

$$\nabla^2 f(x)$$

$$\nabla^2 c_i(x)$$



Stage	Input	Method/Linear Algebra	Validation
Prototyping	By hand	Dense matrices Backslash ($A \setminus b$)	Finds a solution?
Theoretical paper	Small selection	-	Comparison to hand-coded alternative
Computational paper	Collection of problems	Sparse factorization Matrix-free methods	Comparison to established solvers
Maintainable package	Modeling language? Customized model?	-	CI

JSO-compliant solver - streamlined solver development

AbstractNLPModel

- Single mandatory input
- API from NLPModels.jl
- Over 10 implementations

Your method

- JSOSolverSkeleton.jl
- Sparse Factorizations
- Matrix-free

GenericExecutionStats

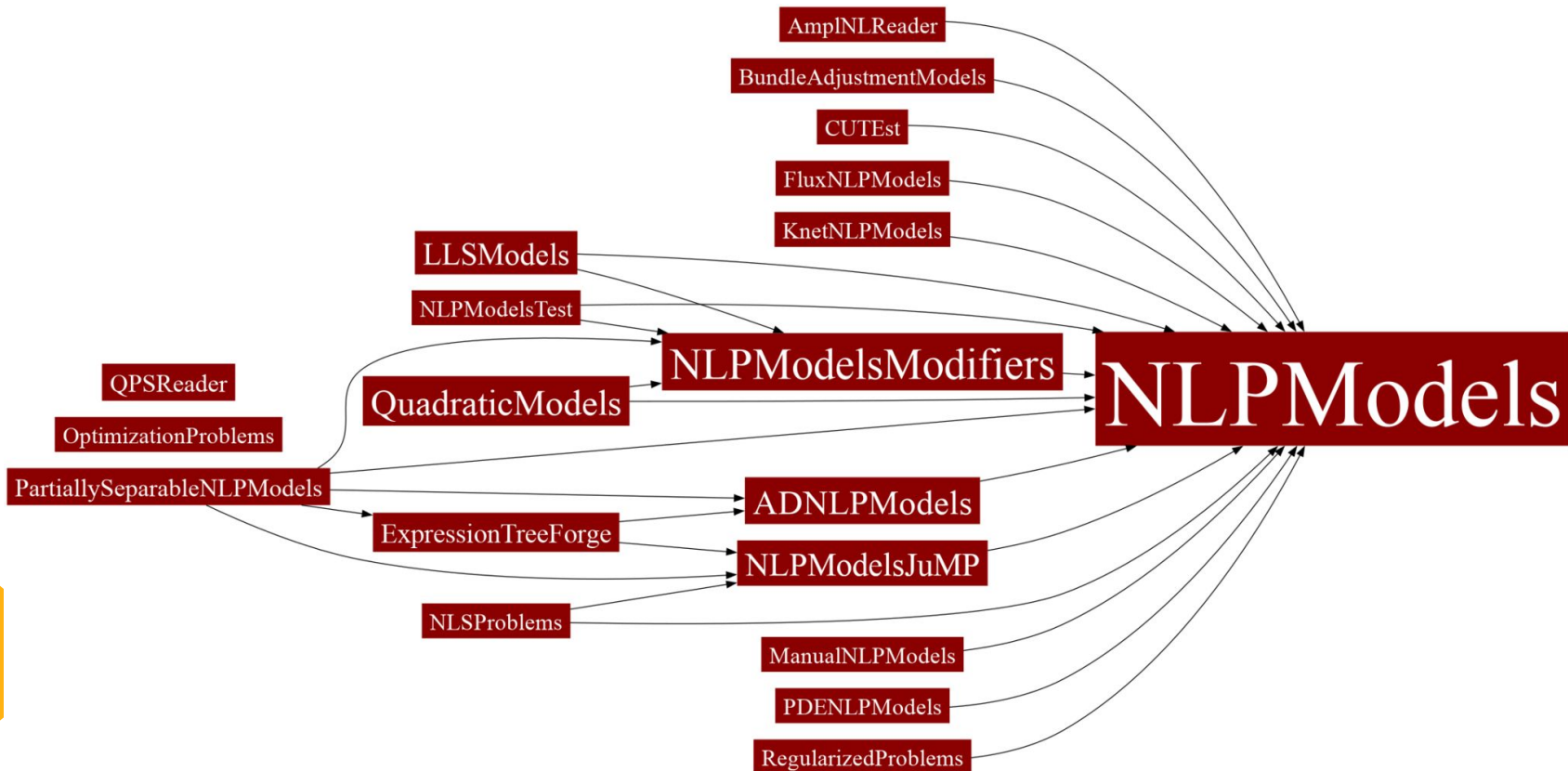
- status
- solution
- eval counters
- solver_specific Dict



Our 3 ecosystems

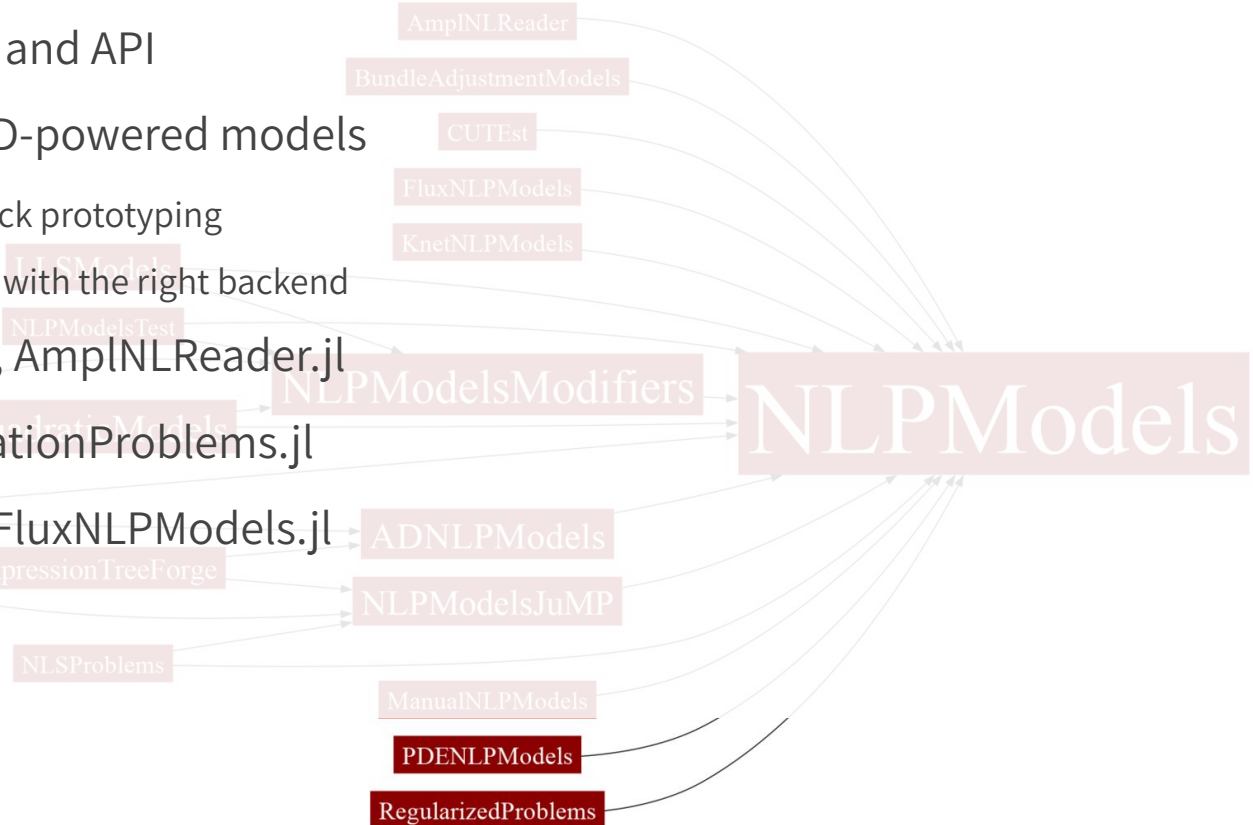


Model ecosystem - NLPModels.jl



Model ecosystem - NLPModels.jl

- AbstractNLPModel and API
- ADNLPModels.jl: AD-powered models
 - Simple API for quick prototyping
 - Sparsity available with the right backend
- NLPModelsJuMP.jl, AmplNLReader.jl
- CUTEst.jl, OptimizationProblems.jl
- PDENLPModels.jl, FluxNLPModels.jl
- NLPModelsTest.jl



```
using ADNLPModels, NLPModels

nlp = ADNLPModel(
  x -> (x[1] - 1)^2 + 4 * (x[2] - x[1]^2)^2, # objective
  [-1.2; 1.0], # starting point
)

x = copy(nlp.meta.x0)
@info x
for i = 1:10
  x .= hess(nlp, x) \ grad(nlp, x)
  @info x
end
```

```
[ Info: [-1.2, 1.0]
[ Info: [-0.7132743362831855, 0.27185840707964526]
[ Info: [-0.12151367194279328, -0.33541511139156144]
[ Info: [0.17350928196578896, -0.05693307240466178]
[ Info: [0.6607383400906399, 0.19918299898450476]
[ Info: [0.7777599336711202, 0.5912164610599902]
[ Info: [0.9780569732145443, 0.9164765388037358]
[ Info: [0.9946685075749813, 0.9890894968876326]
[ Info: [0.9999882564172207, 0.9999482132446086]
[ Info: [0.9999999973418803, 0.9999999945459112]
[ Info: [1.0, 1.0]
```

```
using Symbolics
```

```
n = 5000
```

```
nlp = ADNLPMoDel(  
  x -> sum(  
    (x[2i-1] - 1)^2 + 4 * (x[2i] - x[2i-1]^2)^2  
    for i = 1:div(n, 2)  
  ),  
  [i % 2 == 0 ? 1.0 : -1.2 for i = 1:n],  
  hessian_backend=ADNLPMoDels.SparseADHessian,  
)
```

```
x = copy(nlp.meta.x0)
```

```
for i = 1:10
```

```
  x .-= hess(nlp, x) \ grad(nlp, x)
```

```
  @info obj(nlp, x)
```

```
end
```

```
[ Info: 7899.497346663763  
[ Info: 4370.747404377399  
[ Info: 1783.4743477470408  
[ Info: 851.2965376522218  
[ Info: 125.3518886800629  
[ Info: 17.2990056828858  
[ Info: 0.07182347449490721  
[ Info: 8.353525245325948e-6  
[ Info: 1.7854025739025582e-14  
[ Info: 0.0
```

```
using JuMP, NLPModelsJuMP, Percival
```

```
n = 100
```

```
jmp = Model()
```

```
@variable(jmp, x[1:n])
```

```
@variable(jmp, y[1:n])
```

```
@objective(jmp, Min,
```

```
    sum((x[i] - 1)^2 for i = 1:n) + sum(y[i]^2 for i = 1:n)
```

```
)
```

```
@constraint(jmp, [i = 1:n, j = i+1:n], x[i] + x[j] == y[i] - y[j])
```

```
nlp = MathOptNLPModel(jmp)
```

```
output = percival(nlp)
```

```
println(output)
```

```
Generic Execution stats
```

```
status: first-order stationary
```

```
objective value: 99.00000000020633
```

```
primal feasibility: 3.412464464769002e-10
```

```
dual feasibility: 3.878167262606719e-13
```

```
solution: [0.500000000000522 -9.552671071799871e-11
```

```
5.165716031422584e-11 -8.901878211431562e-12 ... -0.4999999999994779]
```

```
multipliers: [-1.000000000190153 -0.494897959131124
```

```
-0.32479486640496774 -0.23885736639922792 ... -1.0000000001901488]
```

```
iterations: 7
```

```
elapsed time: 0.3596501350402832
```


Linear Algebra ecosystem

SuiteSparseMatrixCollection

SparseMatricesCOO

QRMumps

OperatorScaling

MUMPS

Krylov

HSL

BasicLU

PROPACK

LinearOperators

LDLFactorizations

LimitedLDLFactorizations

AMD

Linear Algebra ecosystem

- jac, hess returns matrices
- jac_coord, hess_coord returns sparse arrays
- LDLFactorizations.jl, HSL.jl, MUMPS.jl
- jprod, jtprod, hprod are matrix-vector products
- LinearOperators.jl - jac_op, hess_op
- Krylov.jl



```
using Krylov
```

```
n = 100
```

```
A = [i == j ? 1.0n : -1.0 for i = 1:n, j = 1:n]
```

```
b = [1.0i for i = 1:n]
```

```
x, cg_stats = Krylov.cg(A, b)
```

```
using LinearOperators, LinearAlgebra
```

```
function Av(res, v)
```

```
    res .= (n + 1) * v .- sum(v)
```

```
    return res
```

```
end
```

```
A = LinearOperator{Float64}(n, n, true, true, Av)
```

```
x, cg_stats = Krylov.cg(A, b)
```

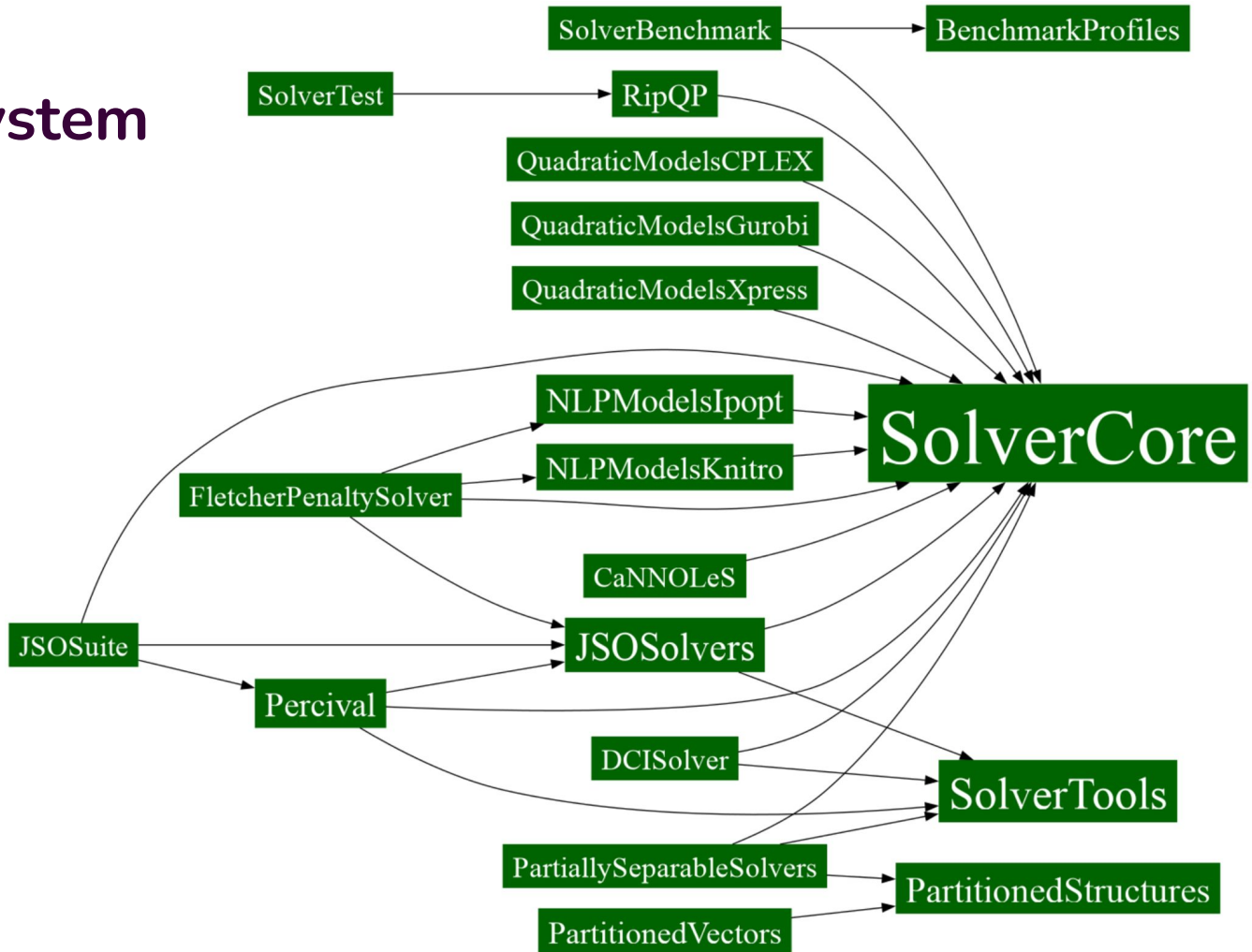
```
([50.00990099009915, 50.01980198019815,  
50.029702970297166 ... 50.970297029703104,  
50.980198019802124, 50.99009900990112], SimpleStats  
niter: 2  
solved: true  
inconsistent: false  
residuals: []  
Aresiduals: []  
 $\kappa_2(A)$ : []  
timer: 81.56 $\mu$ s  
status: solution good enough given atol and rtol  
)
```

```
n = 5000
nlp = ADNLPMoel(
  x -> sum(
    (x[2i-1] - 1)^2 + 4 * (x[2i] - x[2i-1]^2)^2
    for i = 1:div(n, 2)
  ),
  [i % 2 == 0 ? 1.0 : -1.2 for i = 1:n],
)

x = copy(nlp.meta.x0)
for i = 1:10
  d, cg_stats = Krylov.cg(hess_op(nlp, x), grad(nlp, x))
  x .-= d
  @info obj(nlp, x)
end
```

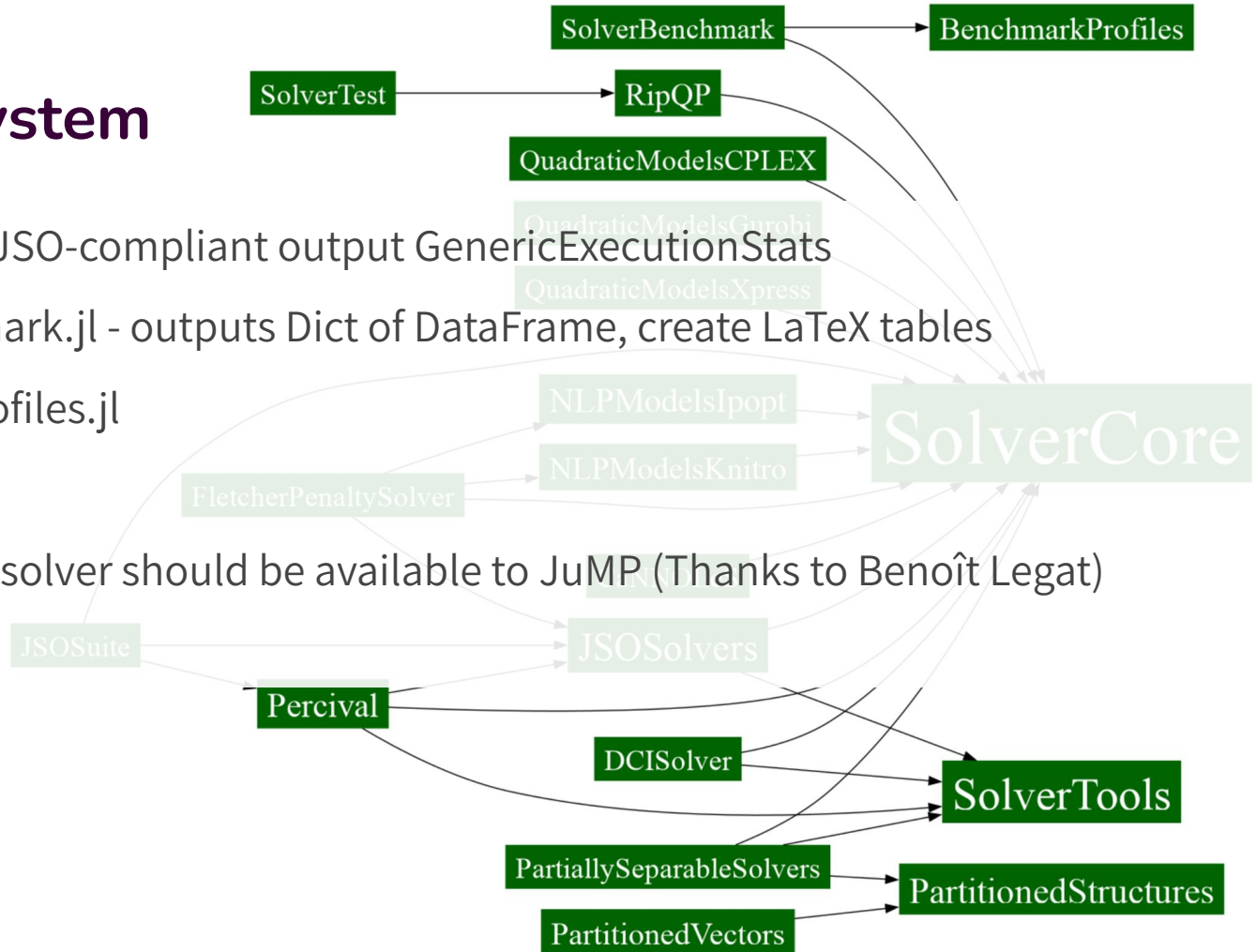
```
[ Info: 7899.497346662107
[ Info: 4370.747404377589
[ Info: 1783.474347746636
[ Info: 851.2965376527912
[ Info: 125.35188867993855
[ Info: 17.299005682867094
[ Info: 0.07182347449456006
[ Info: 8.353525245361876e-6
[ Info: 1.7854025740406985e-14
[ Info: 0.0
```

Solver ecosystem



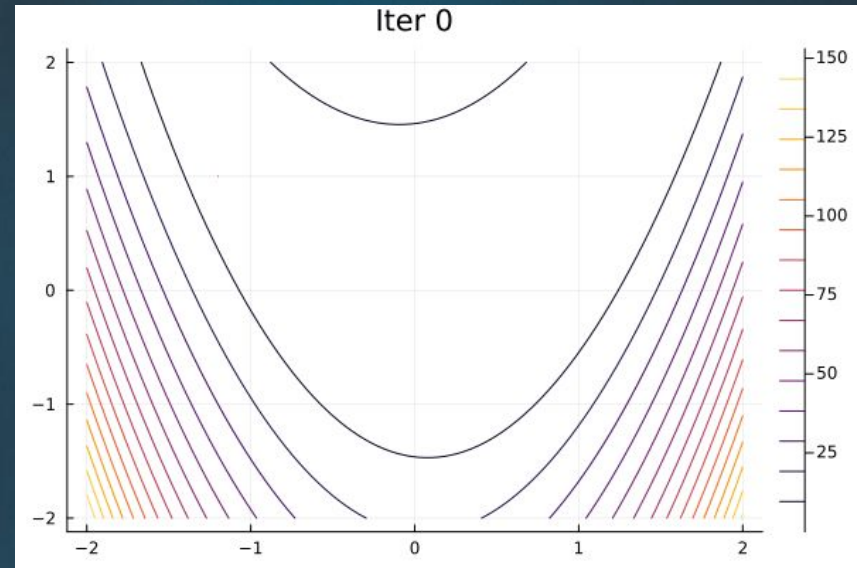
Solver ecosystem

- SolverCore.jl: JSO-compliant output GenericExecutionStats
- SolverBenchmark.jl - outputs Dict of DataFrame, create LaTeX tables
- BenchmarkProfiles.jl
- SolverTest.jl
- Recent: These solver should be available to JuMP (Thanks to Benoît Legat)



```
using Plots, JSOSolvers, ADNLPModels, NLPModels
```

```
function wrapper()  
  anim = Animation()  
  nlp = ADNLPModel(  
    x -> (x[1] - 1)^2 + 4 * (x[2] - x[1]^2)^2,  
    [-1.2; 1.0],  
  )  
  X = zeros(2, 1)  
  X[:, 1] .= nlp.meta.x0  
  cb = (nlp, solver, stats) -> begin  
    X = [X solver.x]  
    contour(  
      range(-2, 2, length=100),  
      range(-2, 2, length=100),  
      (x, y) -> obj(nlp, [x; y]),  
    )  
    plot!(X[1, :], X[2, :], c=:red, l=:arrow, lab=  
    title!("Iter $(stats.iter)")  
    frame(anim)  
  end  
  output = trunk(nlp, callback=cb, max_iter=100)  
  gif(anim, "callback.gif", fps=5)  
end  
wrapper()
```



```
using CUTEst, SolverBenchmark, BenchmarkTools
```

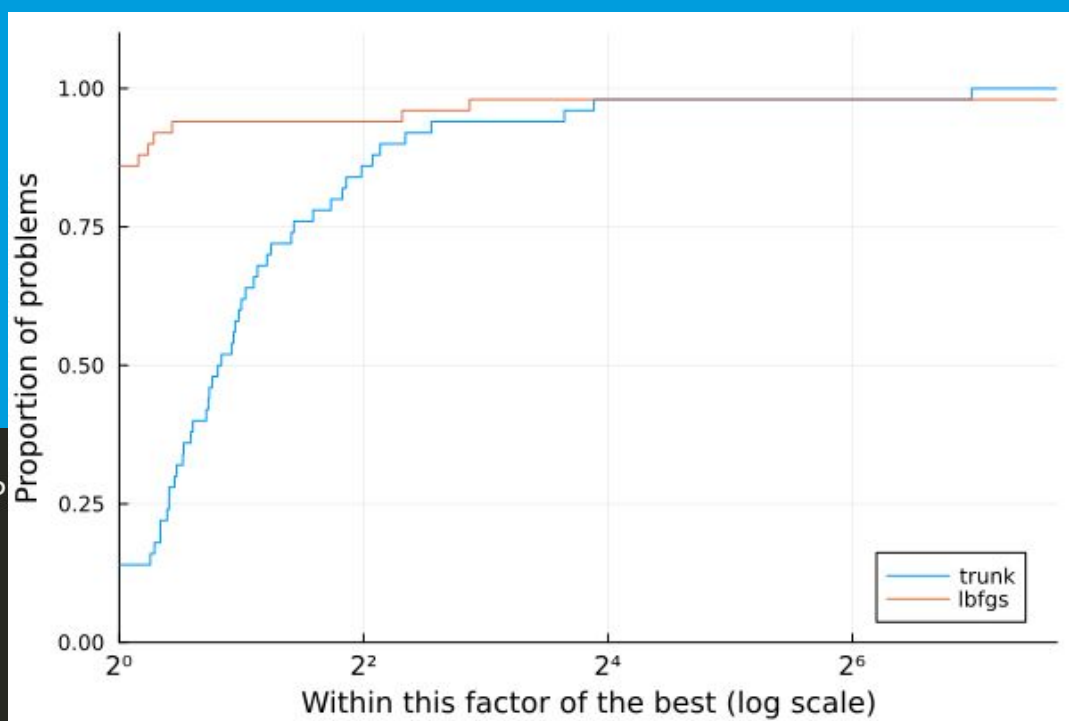
```
solvers = Dict(
  :lbfgs => lbfgs,
  :trunk => trunk,
)
```

```
problem_names = CUTEst.select(max_var=2, max_con=0, only_free_var=true)
```

```
problems = (CUTEstModel(p) for p in problem_names)
```

```
df_per_solver = with_logger(NullLogger()) do
  bmark_solvers(solvers, problems)
end
```

```
performance_profile(df_per_solver, df -> (df.status .!= :first_order) * Inf + df.elapsed_time)
png("perprof")
```



What's new / future plans



JSOSuite.jl

- User-friendly interface
- Select solver amongst loaded
- Wider community
- You can still use JSO solvers through Nonconvex.jl

```
using JSOSuite

output = minimize(
  x -> (x[1] - 1)^2 + 4 * (x[2] - x[1]^2)^2,
  [-1.2; 1.0],
)
```



Challenges

- More core developers
- Maintaining 50+ packages
 - **COPIERTemplate.jl**
 - **Breakage.yml**
 - Jira?



[AUTO] COPIERTemplate.jl update #37

Merged main ← auto-copier-template-update 3 days ago

Conversation 0 Commits 1 Checks 2 Files changed 3

abelsiqueira 3 days ago • edited

Automated changed by Copier.yml workflow

github-actions bot commented 2 weeks ago

Package name	latest	stable
ADNLPMODELS.jl	latest Pass	v0.7.0 Pass
AmpNLReader.jl	latest Fail	v0.11.2 Fail
CUTEst.jl	latest Pass	v0.13.2 Pass
CaNNOLeS.jl	latest Pass	v0.7.5 Pass
DCISolver.jl	latest Pass	v0.4.3 Pass
FletcherPenaltySolver.jl	latest Fail	v0.2.4 Fail
JSOSolvers.jl	latest Pass	v0.11.0 Pass

<https://jso.dev>

- News and tutorials
- Comm. channels

Talk to us

You can find us in the following channels:

- [JuliaLang Slack](#) in the #math-optimization, #linear-algebra, or #smooth-optimizers.
- [Julia Zulip](#) in the #math-optimization.
- [GitHub Discussions](#).
- [Abel Siqueira's YouTube](#).

Tutorials and how-to guides

This is a curated list of tutorials.

Show tags

Introduction to SolverBenchmark

In this tutorial we illustrate the main uses of SolverBenchmark.

[solver](#) [benchmark](#) [profile](#) [latex](#) [SolverBenchmark](#)

Creating an ADNLPModels backend that supports multiple precisions

One of the main strengths of Julia for scientific computing is its native usage of arbitrary precision arithmetic, the same optimization models and solvers.

[models](#) [automatic differentiation](#) [multi-precision](#) [tests](#) [ADNLPModels](#) [JSOSolvers](#) [NLPModels](#) [OptimizationProblems](#)

Introduction to JSOSolvers

This package provides optimization solvers curated by the JuliaSmoothOptimizers organization.

[solvers](#) [nlpmodels](#) [least-squares](#) [nismodels](#) [test set](#) [ADNLPModels](#) [JSOSolvers](#) [SolverCore](#)

Adjustment in the large repository

Thank you



<https://abelsiqueira.com>



abel.s.siqueira@gmail.com



Like and subscribe