



Differentiating Parametric JuMP Models

JuMP-dev 2024 at Montreal, Canada, July 19th, 2024

Differentiable optimization

What?

Given: $(\theta_0, \theta_1, \dots, \theta_m)$

Define: $\mathcal{P} : \min_x F_0(x, \theta_0)$
s.t. $G_i(x, \theta_i) \in \mathcal{S}_i$

Solve: $x^* \in \operatorname{argmin}_x \mathcal{P}$

Compute: $\frac{\partial x^*}{\partial \theta} \equiv \left(\frac{\partial x_j^*}{\partial \theta_i} \right)_{ij}$

Or...

Differentiate the solution of
parametrized optimization
problems

Differentiable optimization

Why?

Classic OR

- Sensitivities

- Perturbation analysis

- Bilevel Optimization

New trends in ML

- Optimization problem as layer

- Hyper parameter tuning

- Learning

Differentiable optimization

Why?

Classic OR




- Sensitivities
- Perturbation analysis
- Bilevel Optimization

New trends in ML

- Optimization problem as layer
- Hyper parameter tuning
- Learning

Home > INFORMS Journal on Computing > Vol. 36, No. 2 >

BilevelJuMP.jl: Modeling and Solving Bilevel Optimization Problems in Julia

Joaquim Dias Garcia , Guilherme Bodin , Alexandre Street 

Published Online: 13 Dec 2023 | <https://doi.org/10.1287/ijoc.2022.0135>

Mathematics > Optimization and Control

[Submitted on 26 Feb 2021 (v1), last revised 8 Apr 2024 (this version, v5)]

Application-Driven Learning: A Closed-Loop Prediction and Optimization Approach Applied to Dynamic Reserves and Demand Forecasting

Joaquim Dias Garcia, Alexandre Street, Tito Homem-de-Mello, Francisco D. Muñoz

(At arXiv but just accepted in Operations Research)

Differentiable optimization

In JuMP

But remember the sentence:

Differentiate the solution of
parametrized optimization problems

Differentiable optimization

In JuMP

But remember the sentence:

Differentiate the solution of
parametrized optimization problems

Which is a lot of work, but we can get some help

Differentiable optimization

In JuMP

But remember the sentence:

Differentiate the solution of
parametrized optimization problems

Which is a lot of work, but we can get some help



Google
Summer of Code

Differentiable optimization: *Differentiate*

[2020 Program](#) | [NumFOCUS](#)

Contributor
AkshaySharma

[View Code](#)

Optimization problem differentiation

Mentors

Joaquim, Benoît Legat, Mario Souto

Organization

NumFOCUS

Differentiable optimization: *Differentiate*

9

[2020 Program](#) | [NumFOCUS](#)



Contributor
AkshaySharma

[View Code](#)

Optimization problem differentiation

[Home](#) > [INFORMS Journal on Computing](#) > [Vol. 36, No. 2](#) >

Flexible Differentiable Optimization via Model Transformations

Mathieu Besançon , Joaquim Dias Garcia , Benoît Legat , Akshay Sharma 

Published Online: 14 Nov 2023 | <https://doi.org/10.1287/ijoc.2022.0283>

Differentiable optimization: *Differentiate*

```
using JuMP, DiffOpt, HiGHS
model = Model(() -> DiffOpt.Optimizer(HiGHS.Optimizer()))
set_silent(model)
@variable(model, x)
@constraint(model, cons, x >= 3 * 3)
@objective(model, Min, 2x)
optimize!(model)

sensitivity = convert(MOI.ScalarAffineFunction{Float64}, 1.0)
MOI.set(model, DiffOpt.ForwardConstraintFunction(), cons, sensitivity)
DiffOpt.forward_differentiate!(model)
MOI.get(model, DiffOpt.ForwardVariablePrimal(), x) # ≈ - 1.0
```

Necessary to differentiate with respect to arbitrary coefficients.

But might be cumbersome to use.

The function is “normalized”
So, it is a sensitivity wrt changing a constant on the LEFT-hand-side

Differentiable optimization: *Parametric*

11

[2020 Program](#) | [NumFOCUS](#)

Contributor
Tomás Gutierrez

[View Code](#)

Adding parameters to JuMP and MathOptInterface

Mentors

Joaquim, Oscar Dowson, Benoît Legat

Organization

NumFOCUS

Differentiable optimization: *Parametric*

12

2020 Program | NumFOCUS

Contributor
Tomás Gutierrez

[View Code](#)

Adding parameters to JuMP and MathOptInterface

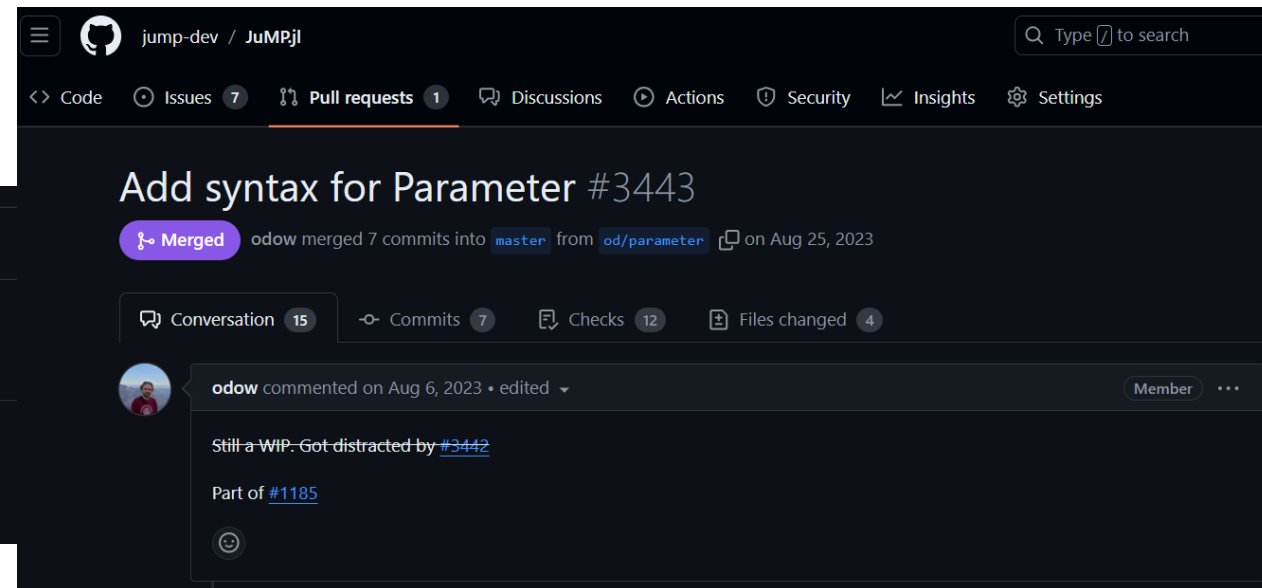


README MIT license

ParametricOptInterface.jl

docs stable docs dev CI no status codecov 95%

[ParametricOptInterface.jl](#) is a package that adds parameters to models in JuMP and MathOptInterface.



jump-dev / JuMP.jl

Code Issues 7 Pull requests 1 Discussions Actions Security Insights Settings

Add syntax for Parameter #3443

Merged odow merged 7 commits into master from od/parameter on Aug 25, 2023

Conversation 15 Commits 7 Checks 12 Files changed 4

odow commented on Aug 6, 2023 • edited

Still a WIP. Got distracted by #3442

Part of #1185

Differentiable optimization: *Parametric*

```
using JuMP, HiGHS
import ParametricOptInterface as POI
model = Model(() -> POI.Optimizer(HiGHS.Optimizer()))
set_silent(model)
@variable(model, x)
@variable(model, p in Parameter(3.0))
@constraint(model, cons, x >= 3 * p)
@objective(model, Min, 2x)
optimize!(model)
value(x) # ≈ 9
set_parameter_value(p, 2.0)
optimize!(model)
value(x) # ≈ 6
```

Differentiating Parametric JuMP Models

14

```
using JuMP, DiffOpt, HiGHS
import ParametricOptInterface as POI
model = Model(() -> POI.Optimizer(DiffOpt.Optimizer(HiGHS.Optimizer())))
set_silent(model)
@variable(model, x)
@variable(model, p in MOI.Parameter(3.0))
@constraint(model, cons, x >= 3 * p)
@objective(model, Min, 2x)
optimize!(model)
value(x) # ≈ 9
# the function is:  $x(p) = 3p$ , hence  $x'(p) = 3$ 
# differentiate w.r.t. p
MOI.set(model, POI.ForwardParameter(), p, 1)
DiffOpt.forward_differentiate!(model)
MOI.get(model, DiffOpt.ForwardVariablePrimal(), x) # ≈ 3
```

Differentiating Parametric JuMP Models

```
using JuMP, DiffOpt, HiGHS
import ParametricOptInterface as POI
model = Model(() -> POI.Optimizer(DiffOpt.Optimizer(HiGHS.Optimizer())))
set_silent(model)
@variable(model, x)
@variable(model, p in MOI.Parameter(3.0))
@constraint(model, cons, x >= 3 * p)
@objective(model, Min, 2x)
optimize!(model)
value(x) # ≈ 9
# the function is: x(p) = 3p,
# differentiate w.r.t. p
MOI.set(model, POI.ForwardParameter(), p, 2.0)
DiffOpt.forward_differentiate!(model)
MOI.get(model, DiffOpt.ForwardVariablePrimal(), x) # ≈ 6
```

```
# update parameter
set_parameter_value(p, 2.0)
optimize!(model)
value(x) # ≈ 6
# differentiate w.r.t. p
MOI.set(model, POI.ForwardParameter(), p, 1)
DiffOpt.forward_differentiate!(model)
MOI.get(model, DiffOpt.ForwardVariablePrimal(), x) # ≈ 3
```

Differentiating Parametric JuMP Models

16

[WIP] POI + DiffOpt = S2 #143

Open

joaquim wants to merge 12 commits into `master` from `jg/diff`

Currently,
DiffOpt needs to be a dependency of POI

But
POI only has MOI as deps
DiffOpt is heavy



Next step,
Move to weakdep + extension approach

Also,
Needs more adjustments in DiffOpt (add bridges and tests)

Differentiating Parametric JuMP Models

17

[WIP] POI + DiffOpt = S2 #143

 Open joaquim wants to merge 12 commits into `master` from `jg/diff` 

Simpler to set sensitivities

More direct approach

Simpler to query derivatives

One parameter might appear in multiple places

Derivatives are accumulated

Use cases typically require multiple solves


Parameters facilitate and speedup the updates

Easier to generalize

To nonlinear(!) (thanks to nonlinear refactor)

See <https://github.com/andrewrosemberg/DiffOpt.jl/pull/1>

(WIP) Test script sIPOPT #1

 Open andrewrosemberg wants to merge 45 commits into `master` from `ar/sipopt`