Dualization.jl

Dualização automática para problemas no MathOptInterface.jl

Guilherme Bodin - @guilhermebodin

Um pouco de otimização para entrar no contexto

Problemas de otimização são uma idéia que é escrita matematicamente como:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p, \end{array}$$

Em otimização existe um conceito chamado dualidade, muito útil em algumas aplicações. Existe a idéia do problema dual que parte de um problema primal de otimização e é escrito como:

$$\begin{array}{ll} \text{maximize} & g(\lambda, \nu) \\ \text{subject to} & \lambda \succeq 0. \end{array}$$

Aqui o mais importante é entender que existe uma classe de problemas de otimização chamados cônicos, nessa classe de problemas escrever o problema dual tem uma forma fechada.

Introdução

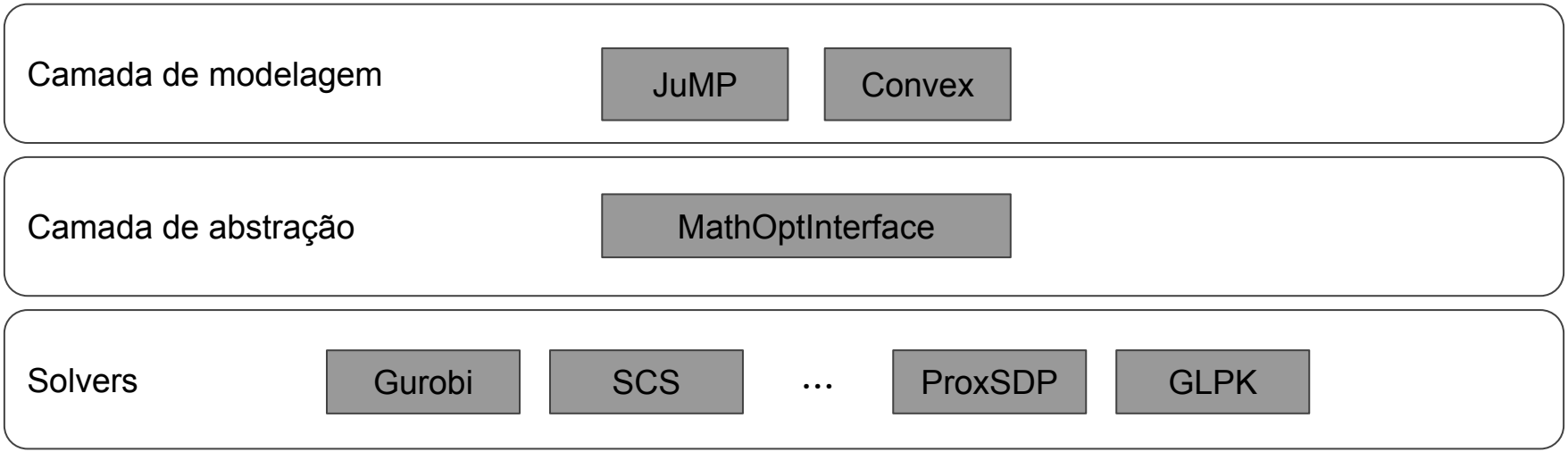
Dualization.jl foi o resultado de um projeto no Google of Summer Of Code (GSOC) chamado “JuMP Automatic Dualization”. Durante o projeto tive três mentores Benoît Legat (@blegat), Chris Coey (@chriscoey) e Joaquim Garcia (@joaquimg). Todos membros do grupo JuliaOpt

O objetivo do projeto era criar um pacote que dependesse apenas do MathOptInterface.jl (MOI) para dualizar problemas de otimização cônicos.

Nessa apresentação vou falar sobre o ecossistema JuliaOpt. Mostrar a motivação de fazer esse pacote e falar um pouco sobre as suas funcionalidades.

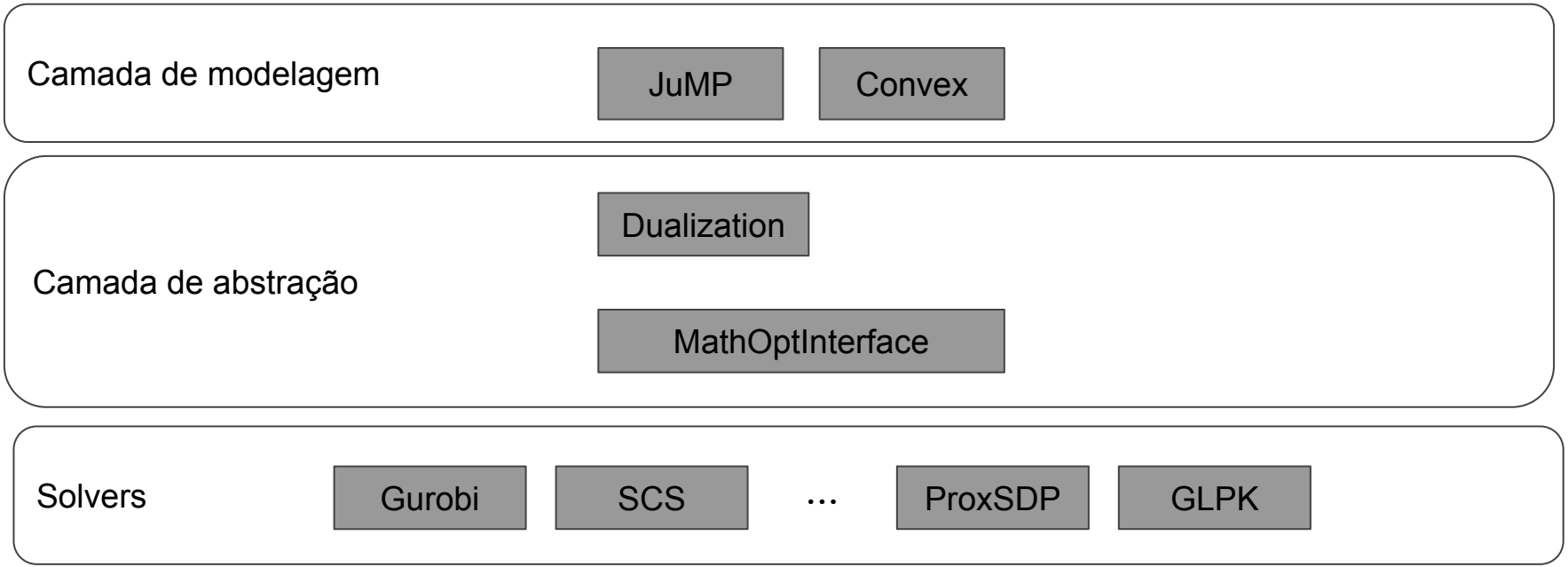
O que é MathOptInterface?

Para explicar melhor o ecossistema JuliaOpt vamos considerar o diagrama



Onde o Dualization entra?

O pacote Dualization.jl é um adendo a camada de abstração



Porque precisamos do pacote Dualization.jl?

Escrever o problema dual automaticamente tem algumas motivações

- Chegar um passo mais próximo de escrever as condições de KKT de um problema definido no MathOptInterface
- Ter as condições de KKT é útil para extender ainda mais a classe de problemas modelados usando pacotes do JuliaOpt, problemas das classes:
  - Extended Mathematical Programming (EMP)
  - Multiple Optimization Problems with Complementarity Constraints (MOPEC)
- Explorar resolver problemas primais usando via o problema dual associado (especialmente quando usamos solvers que olham apenas para o primal)
- Ensinar otimização na faculdade, um pacote como esse pode ser útil para alunos em cursos de otimização convexa.

Resumo da API

Dualization.jl tem duas funcionalidades principais

- A função `dualize`

  Recebe um problema do tipo `MOI.ModelLike` ou `JuMP.Model` e dualiza.

- O “solver” `DualOptimizer`

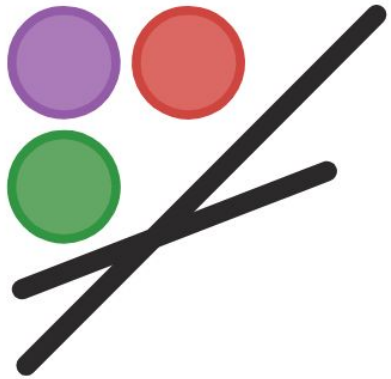
  Resolve um problema de otimização usando a sua representação dual. Isso significa que ele recebe um problema na forma primal, dualiza e resolve usando o solver que você escolher como argumento.

See the docs!

https://www.juliaopt.org/Dualization.jl/latest/

Obrigado!

Não hesitem em usar o pacote, postar issues, Pull Requests, comentários, etc :)



LAMPS

NUMFOCUS
OPEN CODE = BETTER SCIENCE