Optoy Documentation

Release 0.1

Joris Gillis

September 22, 2015

Contents

Contents:

CHAPTER 1

optoy package

1.1 Overview

Home page: http://optoy.casadi.org Examples: http://nbviewer.ipython.org/github/casadi/optoy/tree/master/examples/

1.2 Static optimization

This is the module for static optimization problems

class optoy.static.**OptimizationParameter**(*shape=1*, *value=0*, *name='p'*)
Create a parameter, ie a thing that is fixed during optimization

Parameters **shape: integer or (integer,integer)**

Matrix shape of the symbol

name: string

A name for the symbol to be used in printing. Not required to be unique

value: number or matrix

Value that the parameter should take during optimization May also be set after initialization as 'x.value = number'

Attributes

Methods

sol
Gets the solution

class optoy.static.**OptimizationVariable**(*shape=1*, *lb=-inf*, *ub=inf*, *name='v'*, *init=0*)
Create a decision variable

Parameters **shape: integer or (integer,integer)**

Matrix shape of the symbol

name: string

A name for the symbol to be used in printing. Not required to be unique

lb: number

Lower bound on the decision variable May also be set after initialization as 'x.lb = number'

ub: number

Upper bound on the decision variable May also be set after initialization as 'x.ub = number'

init: number

Initial guess for the optimization solver May also be set after initialization as 'x.init = number'

Attributes

Methods

optoy.static.**minimize**(*f*, *gl=[]*, *verbose=False*)

Miminimizes an objective function subject to a list of constraints. The standard NLP form reads:

```
    mininimze       f(x,p)
        x

     subject to     g(x,p) <= 0
                    h(x,p)  = 0
```

with x the decision variables, p constant parameters, f the objective, g the inequality constraints, and h the equality constraints.

Parameters **f** : symbolic expression

objective function

gl : list of constraints, optional

Equality and inequality constraints can be mixed. Each entry in the constraint list should be

lhs<=rhs , lhs>=rhs or lhs==rhs

where lhs and rhs are expressions.

verbose : bool, optional

Specify the verbosity of the output

Returns If numerical solution was succesful,

returns cost at the optimal solution.

Otherwise raises an exception.

See also:

maximize flip the sign of the objective

optoy.static.**par**

alias of *OptimizationParameter*

optoy.static.**sort_constraints**(*gl*)
Rewrites and determines nature of constraints, either g(x)<=0 or g(x)==0.

A user may write x>=y where x and y are variables. In the *gl_pure* output, everything is brought to the left hand side

Parameters **gl** : list of constraints, optional

Returns **gl_pure** : list of constraints in standard form

The constraints are rewritten as g(x)<=0 or g(x)==0

gl_equality : list of bools

For each entry in *gl_pure*, this list contains a boolean.

optoy.static.**var**
alias of *OptimizationVariable*

1.3 Dynamic optimization

class optoy.dynamic.**OptimizationControl**(*shape=1*, *lb=-inf*, *ub=inf*, *name='u'*, *init=0*)
Create a control variable

Parameters **shape: integer or (integer,integer)**

Matrix shape of the symbol

name: string

A name for the symbol to be used in printing. Not required to be unique

lb: number

Lower bound on the decision variable May also be set after initialization as 'x.lb = number'

ub: number

Upper bound on the decision variable May also be set after initialization as 'x.ub = number'

init: number

Initial guess for the optimization solver May also be set after initialization as 'x.init = number'

Attributes

Methods

class optoy.dynamic.**OptimizationState**(*shape=1*, *lb=-inf*, *ub=inf*, *name='x'*, *init=0*)
Create a state variable

Parameters **shape: integer or (integer,integer)**

Matrix shape of the symbol

name: string

A name for the symbol to be used in printing. Not required to be unique

lb: number

Lower bound on the decision variable May also be set after initialization as 'x.lb = number'

ub: number

Upper bound on the decision variable May also be set after initialization as 'x.ub = number'

init: number

Initial guess for the optimization solver May also be set after initialization as 'x.init = number'

Attributes

Methods

class optoy.dynamic.**OptimizationTime**
time

Attributes

Methods

optoy.dynamic.**control**
alias of *OptimizationControl*

optoy.dynamic.**ocp**(*f*, *gl=[]*, *regularize=[]*, *verbose=False*, *N=20*, *T=1.0*, *periodic=False*, *integration_intervals=1*, *exact_hessian=None*)
Solves an optimal control problem (OCP):

```
    mininimze          E(x(T),v)
     x(t), u(t), v

     subject to            dx/dt       = f(x(t),u(t),v,p)
                 h(x(t),u(t),v,p)    <= 0
                   r(x(0),x(T),v,p)  <= 0
```

with x states, u controls, p static parameters (constant, not optimized for), v variables (constant, optimized for), f the system dynamics, h the path constraints, and r boundary conditions.

In optoy, the system dynamics is specified with the .dot attribute on a state:

```
    >>> x = state()
    >>> x.dot = 1-x**2
```

Parameters **N** : int, optional

number of control intervals

T : float, symbolic expression, optional

time horizon

periodic : bool

indicate whether the problem is periodic

regularize: list of symbolic vector expressions

f : symbolic expression

A major objective function. Make use of the .end attribute of expressions

gl : list of constraints, optional

Equality and inequality constraints can be mixed. Each entry in the constraint list should be

lhs<=rhs , lhs>=rhs or lhs==rhs

where lhs and rhs are expressions. Path constraints and boundary constraints can be mixed. Use .start and .end to obtain the value of a state at the boundaries

verbose : bool, optional

Specify the verbosity of the output

Returns If numerical solution was succesful,

returns cost at the optimal solution.

Otherwise raises an exception.

optoy.dynamic.**state**
alias of *OptimizationState*

1.4 Extensions

class optoy.extensions.robustness.**OptimizationDisturbance**(*shape=1*, *name='w'*, *cov=None*)

Create a disturbance source term

Parameters **shape: integer or (integer,integer)**

Matrix shape of the symbol

name: string

A name for the symbol to be used in printing. Not required to be unique

cov: symmertric matrix

Disturbance covariance matrix

Attributes

Methods

optoy.extensions.robustness.**Prob**(*e*)
h <= 0

optoy.extensions.robustness.**Sigma**(*e*, *nums=None*)
Evaluates the covariance of an expression numerically

Parameters **e: symbolic expression**

the quantity you want the covariance of

nums: dictionary, optional

dictionary denoting the values of variables if not supplied, the optimal values are assumed

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

O

C

M

O

P

S

V