Package ‘highs’

May 16, 2023

Type Package

Title 'HiGHS' Optimization Solver

Version 0.1-10

Description R interface to 'HiGHS', an optimization solver for solving mixed integer optimization problems with quadratic or linear objective and linear constraints.

License GPL (>= 2)

Imports Rcpp (>= 1.0.7), checkmate

SystemRequirements Bash, PkgConfig, CMAKE (>=3.16), C++11

URL `https://gitlab.com/roigrp/solver/highs`

BugReports `https://gitlab.com/roigrp/solver/highs/-/issues`

Suggests tinytest

Biarch FALSE

LinkingTo Rcpp

RoxygenNote 7.2.1

Encoding UTF-8

NeedsCompilation yes

Author Florian Schwendinger [aut, cre],
Dirk Schumacher [aut],
Julian Hall [cph],
Ivet Galabova [cph],
Leona Gottwald [cph],
Michael Feldmeier [cph]

Maintainer Florian Schwendinger <`FlorianSchwendinger@gmx.at`>

Repository CRAN

Date/Publication 2023-05-16 15:30:02 UTC

R topics documented:

`highs_available_solver_options` *Available Solver Options*

Description

Reference for the available solver options.

Usage

```
highs_available_solver_options()
```

Value

A `data.frame` containing the available solver options.

Examples

```
highs_available_solver_options()
```

`highs_control` *Highs Control*

Description

Highs Control

Usage

```
highs_control(threads = 1L, time_limit = Inf, log_to_console = FALSE, ...)
```

Arguments

`threads`	an integer giving the number of threads to be used.
`time_limit`	a double giving the time limit.
`log_to_console`	a logical giving if the output should be shown in the console.
`...`	other arguments supported by the `HiGHS` solver.

Examples

```
control <- highs_control()
```

`highs_model` *Create a Highs Model*

Description

Solve linear and quadratic mixed integer optimization problems.

Usage

```
highs_model(
  Q = NULL,
  L,
  lower,
  upper,
  A,
  lhs,
  rhs,
  types,
  maximum = FALSE,
  offset = 0
)
```

Arguments

`Q`	a numeric symmetric matrix giving the quadratic part of the objective.
`L`	a numeric vector giving the linear part of the objective function.
`lower`	a numeric vector giving the lower bounds of the variables.
`upper`	a numeric vector giving the upper bounds of the variables.
`A`	a numeric matrix giving the linear part of the constraints. Rows are constraints, and columns are decision variables.
`lhs`	a numeric vector giving the left hand-side of the linear constraints.
`rhs`	a numeric vector giving the right hand-side of the linear constraints.
`types`	a integer vector or character vector giving the variable types. `'C'` or `'1'` for continuous, `'I'` or `'2'` for integer, `'SC'` or `'3'` for semi continuous, `'SI'` or `'4'` for semi integer and `'II'` or `'5'` for implicit integer.
`maximum`	a logical if `TRUE` the solver searches for a maximum, if `FALSE` the solver searches for a minimum.
`offset`	a numeric value giving the offset (default is `0`).

Value

A an object of class `highs_model`.

Examples

```
library("highs")
# Minimize:
#  x_0 +  x_1 + 3
# Subject to:
#               x_1 <=  7
#  5 <=  x_0 + 2x_1 <= 15
#  6 <= 3x_0 + 2x_1
#  0 <= x_0 <= 4
#  1 <= x_1
A <- rbind(c(0, 1), c(1, 2), c(3, 2))
m <- highs_model(L = c(1.0, 1), lower = c(0, 1), upper = c(4, Inf),
                 A = A, lhs = c(-Inf, 5, 6), rhs = c(7, 15, Inf),
                 offset = 3)
m

# Minimize:
#  -x_2 - 3x_3 + (1/2) * (2 x_1^2 - 2 x_1x_3 + 0.2 x_2^2 + 2 x_3^2)
# Subject to:
#  x_1 + x_3 <= 2
#  0 <= x
L <- c(0, -1, -3)
Q <- rbind(c(2, 0.0, -1), c(0, 0.2, 0), c(-1, 0.0, 2))
A <- cbind(1, 0, 1)
m <- highs_model(Q = Q, L = L, lower = 0, A = A, rhs = 2)
m
```

`highs_solve` *Solve an Optimization Problems*

Description

Solve linear and quadratic mixed integer optimization problems.

Usage

```
highs_solve(
  Q = NULL,
  L,
  lower,
  upper,
  A,
  lhs,
  rhs,
  types,
  maximum = FALSE,
  offset = 0,
  control = highs_control()
)
```

Arguments

`Q`	a numeric symmetric matrix giving the quadratic part of the objective.
`L`	a numeric vector giving the linear part of the objective function.
`lower`	a numeric vector giving the lower bounds of the variables.
`upper`	a numeric vector giving the upper bounds of the variables.
`A`	a numeric matrix giving the linear part of the constraints. Rows are constraints, and columns are decision variables.
`lhs`	a numeric vector giving the left hand-side of the linear constraints.
`rhs`	a numeric vector giving the right hand-side of the linear constraints.
`types`	a integer vector or character vector giving the variable types. `'C'` or `'1'` for continuous, `'I'` or `'2'` for integer, `'SC'` or `'3'` for semi continuous, `'SI'` or `'4'` for semi integer and `'II'` or `'5'` for implicit integer.
`maximum`	a logical if `TRUE` the solver searches for a maximum, if `FALSE` the solver searches for a minimum.
`offset`	a numeric value giving the offset (default is `0`).
`control`	a list giving additional options for the solver, see highs_available_solver_options or the `README` file for a list of all available options.

Value

A `list` containing the result provided by the solver, containing the following named objects:

`primal_solution`	a numeric vector giving the primal solution.
`objective_value`	a numeric giving the objective value.
`status`	an integer giving the status code
`status_message`	a character string giving the status message (explanation of the `status_code`).
`solver_msg`	a list giving the original (not canonicalized) solver message.
`info`	a list giving additional information provided by the solver.

Additional information on can be found in the `README` file.

Examples

```
library("highs")
# Minimize:
#  x_0 +  x_1 + 3
# Subject to:
#               x_1 <=  7
#  5 <=  x_0 + 2x_1 <= 15
#  6 <= 3x_0 + 2x_1
#  0 <= x_0 <= 4
#  1 <= x_1
A <- rbind(c(0, 1), c(1, 2), c(3, 2))
s <- highs_solve(L = c(1.0, 1), lower = c(0, 1), upper = c(4, Inf),
                 A = A, lhs = c(-Inf, 5, 6), rhs = c(7, 15, Inf),
                 offset = 3)
s[["objective_value"]]
s[["primal_solution"]]

# Minimize:
#  -x_2 - 3x_3 + (1/2) * (2 x_1^2 - 2 x_1x_3 + 0.2 x_2^2 + 2 x_3^2)
# Subject to:
#  x_1 + x_3 <= 2
#  0 <= x
L <- c(0, -1, -3)
Q <- rbind(c(2, 0.0, -1), c(0, 0.2, 0), c(-1, 0.0, 2))
A <- cbind(1, 0, 1)
s <- highs_solve(Q = Q, L = L, lower = 0, A = A, rhs = 2)
s[["objective_value"]]
s[["primal_solution"]]
```

highs_solver *Highs Solver*

Description

Create a wrapper around the `HiGHS` solver. Manly usefull if one wants a low level wrapper around highs with hot-start capabilities.

Usage

```
highs_solver(model, control = highs_control())
```

Arguments

`model`	an object of class `"highs_model"` created with `highs_model()`.
`control`	an object of class `"highs_control"` created with `highs_control()`.

Details

Methods

The following methods are provided by the `"highs_solver"` class.

- `solve(...)` method to be called to solve the optimization problem. Returns an integer giving the status code returned by **HiGHS**.
- `status()` method to obtain the status from the solver.
- `status_message()` method to obtain the status message from the solver.
- `solution()` method to obtain the solution from the solver.
- `info()` info to obtain addtional information from the solver.
- `L(i, v)` method to get and set the linear part of the objective.

- `A(i, j, v)` method to get and set the constraint matrix coefficients.
- `cbounds(i, lhs, rhs)` method to get and set the constraint bounds (left hand-side and right hand-side).
- `types(i, v)` method to get and set the variable types.
- `vbounds(i, lower, upper)` method to get and set the variable bounds.
- `maximum(maximize)` method to get and set the sense of the problem.

Method arguments

- `...` optional control arguments, which can be used to alter the options set via the `control` argument when initializing the solver.
- `i` a vector of integers giving the index (vector index or row index) of the coeficcients to be altered.
- `j` a vector of integers giving the index (column index) of the coeficcients to be altered.
- `v` a vector of doubles giving the values of the coeficcients to be altered.
- `lhs` a vector of doubles giving left hand-side.
- `rhs` a vector of doubles giving right hand-side.
- `lower` a vector of doubles giving the lower bounds to be altered.
- `upper` a vector of doubles giving the upper bounds to be altered.

Value

an object of class `"highs_solver"`.

Examples

```
A <- rbind(c(0, 1), c(1, 2), c(3, 2))
m <- highs_model(L = c(1.0, 1), lower = c(0, 1), upper = c(4, Inf),
                 A = A, lhs = c(-Inf, 5, 6), rhs = c(7, 15, Inf),
                 offset = 3)
solver <- highs_solver(m)
```

Index