Package 'gurobi'

November 13, 2016

Version 6.5-1

Date 2016-02-29

Title Gurobi Optimizer 6.5 interface

Author Gurobi Optimization, Inc.

Maintainer Gurobi Optimization, Inc. <support@gurobi.com>

Description R interface to Gurobi Optimizer

License GPL-2

Copyright Copyright (c) 2016, Gurobi Optimization, Inc.

Depends slam (>= 0.1-9)

URL http://www.gurobi.com

R topics documented:

`gurobi` *Solve an LP, QP, or MIP using the Gurobi Optimizer*

Description

Interface to the Gurobi Optimizer, which solves linear, quadratic, and mixed integer programming problems.

The interface can be used to solve optimization problems of the following form:

minimize	x'Qx + c'x	
subject to	Ax = b	(linear constraints)
	l <= x <= u	(bound constraints)
	some xj integral	(integrality constraints)
	some xk lie within second order cones	(cone constraints)

Many of the model components listed here are optional.

Usage

```
result <- gurobi(model, params)
```

Arguments

The gurobi function takes a pair of list variables, each consisting of multiple named components.

The first argument, model, contains the optimization model to be solved. Many of model's named components are optional. The following is an enumeration of all the named components of the model argument.

	The linear constraint matrix. This can be dense or sparse. Sparse matrices should be built using either sparseMatrix from the Matrix package, or simple_triplet_matrix from the slam package.
model$obj	The linear objective vector (the c vector in the problem statement above). You must specify one value for each column of A.
model$sense	The senses of the linear constraints. Allowed values are '=', '<=', or '>='. You must specify one value for each row of A.
model$rhs	The right-hand side vector for the linear constraints (the b vector in the problem statement above). You must specify one value for each row of A.
model$lb	Optional. The lower bound vector. When present, you must specify one value for each column of A. When absent, each variable has a lower bound of 0.
model$ub	Optional. The upper bound vector. When present, you must specify one value for each column of A. When absent, the variables have infinite upper bounds.
model$vtypes	Optional. The variable type vector. This vector is used to capture variable integrality constraints. Allowed values are 'C' (continuous), 'B' (binary), 'I' (integer), 'S' (semi-continuous), or 'N' (semi-integer). Binary variables must be either 0 or 1. Integer variables can take any integer value between the specified lower and upper bounds. Semi-continuous variables can take any value between the specified lower and upper bounds, or a value of zero. Semi-integer variables can take any integer value between the specified lower and upper bounds, or a value of zero. When present, you must specify one value for each column of A. When absent, each variable is treated as being continuous.
model$modelsense	Optional. The optimization sense. Allowed values are 'min' (minimize) or 'max' (maximize). When absent, the default optimization sense is minimization.
model$modelname	Optional. The name of the model. The name appears in the Gurobi log, and when writing a model to a file.
model$objcon	Optional. The constant offset in the objective function.
model$start	Optional. The MIP start vector. The MIP solver will attempt to build an initial solution from this vector. When present, you must specify a start value for each variable. Note that you can set the start value for a variable to NA, which instructs the MIP solver to try to fill in a value for that variable.

`model$vbasis` Optional. The variable basis status vector. Used to provide an advanced starting point for the simplex algorithm. You would generally never concern yourself with the contents of this array, but would instead simply pass it from the result of a previous optimization run to the input of a subsequent run. When present, you must specify one value for each column of A.

`model$cbasis` Optional. The constraint basis status vector. Used to provide an advanced starting point for the simplex algorithm. Consult the *vbasis* description for details. When present, you must specify one value for each row of A.

`model$Q` Optional. The quadratic objective matrix. When present, Q must be a square matrix whose row and column counts are equal to the number of columns in A.

`model$cones` Optional. Second-order cone constraints. A list of lists. Each member list defines a single cone constraint: sum x_i^2 <= y^2. The first integer in the list gives the column index for variable y, and the remainder give the column indices for the x variables.

The second argument, `params` is an optional list of Gurobi parameters to be modified during the solution process.

`params` A list of Gurobi parameter changes.

Details

The Gurobi Optimizer is a commercial library for solving linear, quadratic, and mixed integer programming problems. More information on Gurobi Optimization, and online documentation can be found at http://www.gurobi.com.

Value

A list `result` containing the optimal solution, with the following components:

`result$status` The status of the optimization, returned as a string. The desired result is "OPTIMAL", which indicates that an optimal solution to the model was found. Other status are possible, for example if the model has no feasible solution or if you set a Gurobi parameter that leads to early solver termination. Status codes are documented in the Gurobi Reference Manual.

`result$objval` The value of the objective function for the computed solution. Not populated if optimization terminated without finding a feasible solution.

`result$x` Variable values for the best solution found. One entry per column of A. Not present if optimization terminated without finding a feasible solution.

`result$slack` Constraint slacks. One entry per row of A.

`result$pi` Dual multipliers for the constraints. One entry per row of A. Only returned for continuous models.

`result$rc` Variable reduced costs. One entry per column of A. Only returned for continuous models.

`result$vbasis` Variable basis status values for the computed optimal basis. You generally should not concern yourself with the contents of this array. If you wish to use an advanced start later, you would simply copy the *vbasis* and *cbasis* arrays into the corresponding components for the next model. This array contains one entry for each column of A.

`result$cbasis` Constraint basis status values for the computed optimal basis. This array contains one entry for each row of A.

Author(s)

Gurobi Optimization

References

Gurobi Optimization (http://www.gurobi.com).

Gurobi Optimizer Reference Manual (http://www.gurobi.com/documentation/6.5/reference-manual/).

Examples

```
# minimize:   x +   y + 2 z
# subject to: x + 2 y + 3 z <= 4
#             x +   y       >= 1
#             x, y, z binary

library("gurobi")
model  <- list()

model$A      <- matrix(c(1, 2, 3, 1, 1, 0), nrow = 2, ncol=3, byrow=T)
model$obj    <- c(1, 1, 2)
model$sense  <- c("<=", ">=")
model$rhs    <- c(4, 1)
model$vtype  <- "B"

params <- list(Presolve=2, TimeLimit=100.0)

result <- gurobi(model, params)

print(result$objval)
print(result$x)
```

Index