Package ‘CLSOCP’

February 19, 2015

Type Package

Title A smoothing Newton method SOCP solver

Version 1.0

Date 2010-11-05

Author Jason Rudy

Maintainer Jason Rudy <jcrudy@gmail.com>

Description This package provides and implementation of a one step smoothing newton method for the solution of second order cone programming problems, originally described by Xiaoni Chi and Sanyang Liu.

Depends Matrix

License GPL-3

LazyLoad yes

Repository CRAN

Date/Publication 2011-07-23 13:01:34

NeedsCompilation no

R topics documented:

`socp` *Solve Second Order Cone Programs*

Description

Solves Second Order Cone Programs (SOCP) using the one step smoothing Newton method of Chi and Liu.

Usage

```
socp(A, b, c, kvec, type = rep('q',length(kvec)), use_sparse=TRUE, gamma_fac=.95, delta0 = .75, sigma0
```

Arguments

`A`	`A` is a matrix containing the coefficients for the linear and second order cone constraints. `A` should have m rows, where m is the number of constraints. The number of columns in `A` should be `sum(kvec)`. `A` must have full row rank.
`b`	`b` is a vector containing the affine terms of the constraints.
`c`	`c` is a vector containing the coefficients of the objective function to be minimized.
`kvec`	`kvec` is a vector containing the length of each constraint block.
`type`	`type` is a vector of the same length as `kvec` containing the type of each constraint block. Possible values are "q" for second order cone blocks or "l" for linear blocks.
`use_sparse`	`use_sparse` indicates whether or not to use sparse matrices (via the Matrix package) for computations.
`gamma_fac`	`gamma_fac` is used to calculate gamma (see Chi and Liu, 2009) by `gamma <- gamma_fac*min(1,1/nrm_H`
`delta0`	A parameter affecting the behavior of the algorithm. See Chi and Liu, 2009.
`sigma0`	A parameter affecting the behavior of the algorithm. See Chi and Liu, 2009.
`mu0`	A parameter affecting the behavior of the algorithm. See Chi and Liu, 2009.
`zero_tol`	The threshold for completion of the algorithm. See Chi and Liu, 2009.
`max_iter`	The maximum number of allowed iterations if `zero_tol` is not reached.
`min_progress`	The minimum progress that must be made on each iteration to continue execution.

Details

A second order cone program (SOCP) is an optimization problem similar to a linear program (LP), except that some variables can be constrained by second order cones. An exact mathematical definition can be found in Chi and Liu, 2009. This function implements the algorithm given in that paper. The algorithm has been extended here to allow for multiple second order cone constraints as well as linear constraints. The objective function is given by `sum(c*x)` while the constraints are `A%*%x == b`, with `x` belonging to the cartesian product of second order cones described by `kvec` and `type`.

Value

A list containing named elements:

`x`	The optimal solution to the SOCP. See details.
`y`	The dual solution. See Chi and Liu, 2009.
`s`	Given by `c - t(A)%*%y`. See Chi and Liu, 2009.
`obj`	The value of the objective for the optimal solution.

`code` The status of the result. 0 indicates that the function completed with no problems. 1 indicates that a singularity occured. 2 indicates termination due to lack of progress. 3 indicates termination due to the maximum number of iterations being reached. Only solutions with a code of 0 should be relied upon.

`mu` The final value of the smoothing parameter. See Chi and Liu, 2009.

`iter` The number of iterations performed.

Note

No attempt is made to check the feasibility of the SOCP. Infeasible inputs may result in unexpected behavior, although usually they will result in a failure code.

Author(s)

Jason Rudy

References

Chi and Liu. A one-step smoothing Newton method for second-order cone programming. Journal of Computational and Applied Mathematics (2009) vol. 223 (1) pp. 114-123

Examples

```
#Load an example SOCP
data(prob)

#Solve the socp
soln <- socp(prob$A, prob$b, prob$c, dim(prob$A)[2])
```

Index