

Package ‘nonneg.cg’

September 5, 2019

Type Package

Title Non-Negative Conjugate-Gradient Minimizer

Version 0.1.6

Date 2019-09-04

Author David Cortes

Maintainer David Cortes <david.cortes.rivera@gmail.com>

URL https://github.com/david-cortes/nonneg_cg

BugReports https://github.com/david-cortes/nonneg_cg/issues

Description

Minimize a differentiable function subject to all the variables being non-negative (i.e. ≥ 0), using a Conjugate-Gradient algorithm based on a modified Polak-Ribiere-Polyak formula as described in (Li, Can, 2013, <<https://www.hindawi.com/journals/jam/2013/986317/abs/>>).

License BSD_2_clause + file LICENSE

Imports Rcpp ($\geq 0.12.19$)

LinkingTo Rcpp

RoxygenNote 6.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-09-05 04:40:07 UTC

R topics documented:

minimize.nonneg.cg 2

Index 4

 minimize.nonneg.cg *Non-Negative CG Minimizer*

Description

Minimize a differentiable function subject to all the variables being non-negative (i.e. ≥ 0), using a Conjugate-Gradient algorithm based on a modified Polak-Ribiere-Polyak formula (see reference at the bottom for details).

Usage

```
minimize.nonneg.cg(evaluate_function, evaluate_gradient, x0, tol = 1e-04,
  maxnfeval = 1500, maxiter = 200, decr_lnsrch = 0.5,
  lnsrch_const = 0.01, max_ls = 20, extra_nonneg_tol = FALSE,
  nthreads = 1, verbose = FALSE, ...)
```

Arguments

evaluate_function	function(x, ...) objective evaluation function
evaluate_gradient	function(x, ...) gradient evaluation function
x0	Starting point. Must be a feasible point (≥ 0). Be aware that it might be modified in-place.
tol	Tolerance for <gradient, direction>
maxnfeval	Maximum number of function evaluations
maxiter	Maximum number of CG iterations
decr_lnsrch	Number by which to decrease the step size after each unsuccessful line search
lnsrch_const	Acceptance parameter for the line search procedure
max_ls	Maximum number of line search trials per iteration
extra_nonneg_tol	Ensure extra non-negative tolerance by explicitly setting elements that are ≤ 0 to zero at each iteration
nthreads	Number of parallel threads to use (ignored if the package was installed from CRAN)
verbose	Whether to print convergence messages
...	Extra parameters to pass to the objective and gradient functions

Details

The underlying C function can also be called directly from Rcpp with 'R_GetCCallable' (see example of such usage in the source code of the 'zoo' package).

References

Li, C. (2013). A conjugate gradient type method for the nonnegative constraints optimization problems. *Journal of Applied Mathematics*, 2013.

Examples

```
fr <- function(x) { ## Rosenbrock Banana function
  x1 <- x[1]
  x2 <- x[2]
  100 * (x2 - x1 * x1)^2 + (1 - x1)^2
}
grr <- function(x) { ## Gradient of 'fr'
  x1 <- x[1]
  x2 <- x[2]
  c(-400 * x1 * (x2 - x1 * x1) - 2 * (1 - x1),
    200 * (x2 - x1 * x1))
}
minimize.nonneg.cg(fr, grr, x0 = c(0,2), verbose=TRUE, tol=1e-8)
```

Index

`minimize.nonneg.cg`, [2](#)