Package ‘clarabel’

June 24, 2023

Type Package
Title Interior Point Conic Optimization Solver
Version 0.5.1
Description A versatile interior point solver that solves linear programs (LPs), quadratic programs (QPs), second-order cone programs (SOCPs), semidefinite programs (SDPs), and problems with exponential and power cone constraints (<https://oxfordcontrol.github.io/ClarabelDocs/stable/>). For quadratic objectives, unlike interior point solvers based on the standard homogeneous self-dual embedding (HSDE) model, 'Clarabel' handles quadratic objective without requiring any epigraphical reformulation of its objective function. It can therefore be significantly faster than other HSDE-based solvers for problems with quadratic objective functions. Infeasible problems are detected using using a homogeneous embedding technique.
License Apache License (== 2.0)
SystemRequirements Cargo (rustc package manager) and GNU make
Encoding UTF-8
Config/rextendr/version 0.3.1
RoxygenNote 7.2.3
URL https://oxfordcontrol.github.io/clarabel-r/
BugReports https://github.com/oxfordcontrol/clarabel-r/issues
Suggests knitr, Matrix, rmarkdown, tinytest
VignetteBuilder knitr
NeedsCompilation yes
Author Balasubramanian Narasimhan [aut, cre],
Paul Goulart [aut, cph],
Yuwen Chen [aut],
Hiroaki Yutani [ctb] (For vendoring/Makefile hints/R scripts for generating crate authors/licenses),
The authors of the dependency Rust crates [ctb] (see inst/AUTHORS file for details)
Maintainer Balasubramanian Narasimhan <naras@stanford.edu>
Repository CRAN
Date/Publication 2023-06-24 18:30:09 UTC
R topics documented:

clarabel-package .............................................. 2
clarabel .......................................................... 2
clarabel_control .................................................. 4
solver_status_descriptions ................................. 7

Index 8

clarabel-package Interface to Clarabel solver implemented in Rust.

Description

Clarabel is a versatile interior point solver for convex programs using a new homogeneous embedding. It solves linear programs (LPs), quadratic programs (QPs), second-order cone programs (SOCPs), and problems with exponential and power cone constraints. For quadratic objectives, unlike interior point solvers based on the standard homogeneous self-dual embedding (HSDE) model, Clarabel handles quadratic objective without requiring any epigraphical reformulation of its objective function. It can therefore be significantly faster than other HSDE-based solvers for problems with quadratic objective functions. Infeasible problems are detected using a homogeneous embedding technique. See https://oxfordcontrol.github.io/ClarabelDocs/stable/.

Author(s)

Balasubramanian Narasimhan, Paul Goulart, Yuwen Chen

clarabel Interface to ‘Clarabel’, an interior point conic solver

Description

Clarabel solves linear programs (LPs), quadratic programs (QPs), second-order cone programs (SOCPs) and semidefinite programs (SDPs). It also solves problems with exponential and power cone constraints. The specific problem solved is:

Minimize

\[ \frac{1}{2} x^T P x + q^T x \]

subject to

\[ Ax + s = b \]

\[ s \in K \]

where \( x \in R^n \), \( s \in R^m \), \( P = P^T \) and nonnegative-definite, \( q \in R^n \), \( A \in R^{m \times n} \), and \( b \in R^m \). The set \( K \) is a composition of convex cones.
Usage

clarabel(A, b, q, P = NULL, cones, control = list(), strict_cone_order = TRUE)

Arguments

A  a matrix of constraint coefficients.

b  a numeric vector giving the primal constraints

q  a numeric vector giving the primal objective

P  a symmetric positive semidefinite matrix, default NULL

cones a named list giving the cone sizes, see “Cone Parameters" below for specification

control a list giving specific control parameters to use in place of default values, with an empty list indicating the default control parameters. Specified parameters should be correctly named and typed to avoid Rust system panics as no sanitization is done for efficiency reasons

strict_cone_order a logical flag, default TRUE for forcing order of cones described below. If FALSE cones can be specified in any order and even repeated and directly passed to the solver without type and length checks

Details

The order of the rows in matrix $A$ has to correspond to the order given in the table “Cone Parameters”, which means means rows corresponding to primal zero cones should be first, rows corresponding to non-negative cones second, rows corresponding to second-order cone third, rows corresponding to positive semidefinite cones fourth, rows corresponding to exponential cones fifth and rows corresponding to power cones at last.

When the parameter strict_cone_order is FALSE, one can specify the cones in any order and even repeat them in the order they appear in the $A$ matrix. See below.

Clarabel can solve:

1. linear programs (LPs)
2. second-order cone programs (SOCPs)
3. exponential cone programs (ECPs)
4. power cone programs (PCPs)
5. problems with any combination of cones, defined by the parameters listed in “Cone Parameters” below

Cone Parameters: The table below shows the cone parameter specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z$</td>
<td>integer</td>
<td>1</td>
<td>number of primal zero cones (dual free cones), which corresponds to the primal equality constraints</td>
</tr>
<tr>
<td>$l$</td>
<td>integer</td>
<td>1</td>
<td>number of linear cones (non-negative cones)</td>
</tr>
<tr>
<td>$q$</td>
<td>integer</td>
<td>$\geq$ 1</td>
<td>vector of second-order cone sizes</td>
</tr>
<tr>
<td>$s$</td>
<td>integer</td>
<td>$\geq$ 1</td>
<td>vector of positive semidefinite cone sizes</td>
</tr>
<tr>
<td>$ep$</td>
<td>integer</td>
<td>1</td>
<td>number of primal exponential cones</td>
</tr>
<tr>
<td>$p$</td>
<td>numeric</td>
<td>$\geq$ 1</td>
<td>vector of primal power cone parameters</td>
</tr>
</tbody>
</table>
When the parameter `strict_cone_order` is `FALSE`, one can specify the cones in the order they appear in the `A` matrix. The cones argument in such a case should be a named list with names matching `^z*$ indicating primal zero cones, `^l*$ indicating linear cones, and so on. For example, either of the following would be valid: `list(z = 2L, l = 2L, q = 2L, z = 3L, q = 3L)`, or, `list(z = 2L, l = 2L, q = 2L, zb = 3L, qx = 3L)`, indicating a zero cone of size 2, followed by a linear cone of size 2, followed by a second-order cone of size 2, followed by a zero cone of size 3, and finally a second-order cone of size 3.

Value

named list of solution vectors `x`, `y`, `s` and information about run

See Also

`clarabel_control()`

Examples

```r
A <- matrix(c(1, 1), ncol = 1)
b <- c(1, 1)
obj <- 1
cone <- list(z = 2L)
control <- clarabel_control(tol_gap_rel = 1e-7, tol_gap_abs = 1e-7, max_iter = 100)
clarabel(A = A, b = b, q = obj, cones = cone, control = control)
```

---

### Description

Control parameters with default values and types in parenthesis

### Usage

```r
clarabel_control(
  max_iter = 200L,
  time_limit = Inf,
  verbose = TRUE,
  max_step_fraction = 0.99,
  tol_gap_abs = 1e-08,
  tol_gap_rel = 1e-08,
  tol_feas = 1e-08,
  tol_infeas_abs = 1e-08,
  tol_infeas_rel = 1e-08,
  tol_kratio = 1e-06,
  reduced_tol_gap_abs = 5e-05,
  reduced_tol_gap_rel = 5e-05,
)```
clarabel_control

reduced_tol_feas = 1e-04,
reduced_tol_infeas_abs = 5e-05,
reduced_tol_infeas_rel = 5e-05,
reduced_tol_ktratio = 1e-04,
equilbrate_enable = TRUE,
equilbrate_max_iter = 10L,
equilbrate_min_scaling = 1e-04,
equilbrate_max_scaling = 10000,
linesearch_backtrack_step = 0.8,
min_switch_step_length = 0.1,
min_terminate_step_length = 1e-04,
direct_kkt_solver = TRUE,
direct_solve_method = c("qdldl", "mkl", "cholmod"),
static_regularization_enable = TRUE,
static_regularization_constant = 1e-08,
static_regularization_proportional = .Machine$double.eps * .Machine$double.eps,
dynamic_regularization_enable = TRUE,
dynamic_regularization_eps = 1e-13,
dynamic_regularization_delta = 2e-07,
iterative_refinement_enable = TRUE,
iterative_refinement_reltol = 1e-13,
iterative_refinement_abstol = 1e-12,
iterative_refinement_max_iter = 10L,
iterative_refinement_stop_ratio = 5,
presolve_enable = TRUE
)

Arguments

max_iter maximum number of iterations (200L)
time_limit maximum run time (seconds) (Inf)
verbose verbose printing (TRUE)
max_step_fraction maximum interior point step length (0.99)
tol_gap_abs absolute duality gap tolerance (1e-8)
tol_gap_rel relative duality gap tolerance (1e-8)
tol_feas feasibility check tolerance (primal and dual) (1e-8)
tol_infeas_abs absolute infeasibility tolerance (primal and dual) (1e-8)
tol_infeas_rel relative infeasibility tolerance (primal and dual) (1e-8)
tol_ktratio KT tolerance (1e-7)
reduced_tol_gap_abs reduced absolute duality gap tolerance (5e-5)
reduced_tol_gap_rel reduced relative duality gap tolerance (5e-5)
reduced_tol_feas reduced feasibility check tolerance (primal and dual) (1e-4)
reduced_tol_infeas_abs
  reduced absolute infeasibility tolerance (primal and dual) (5e-5)
reduced_tol_infeas_rel
  reduced relative infeasibility tolerance (primal and dual) (5e-5)
reduced_tol_ktratio
  reduced KT tolerance (1e-4)
equilibrate_enable
  enable data equilibration pre-scaling (TRUE)
equilibrate_max_iter
  maximum equilibration scaling iterations (10L)
equilibrate_min_scaling
  minimum equilibration scaling allowed (1e-4)
equilibrate_max_scaling
  maximum equilibration scaling allowed (1e+4)
linesearch_backtrack_step
  linesearch backtracking (0.8)
min_switch_step_length
  minimum step size allowed for asymmetric cones with PrimalDual scaling (1e-1)
min_terminate_step_length
  minimum step size allowed for symmetric cones && asymmetric cones with Dual scaling (1e-4)
direct_kkt_solver
  use a direct linear solver method (required true) (TRUE)
direct_solve_method
  direct linear solver ("qdldl", "mkl" or "cholmod") ("qdldl")
static_regularization_enable
  enable KKT static regularization (TRUE)
static_regularization_constant
  KKT static regularization parameter (1e-8)
static_regularization_proportional
  additional regularization parameter w.r.t. the maximum abs diagonal term (.Machine.double_eps^2)
dynamic_regularization_enable
  enable KKT dynamic regularization (TRUE)
dynamic_regularization_eps
  KKT dynamic regularization threshold (1e-13)
dynamic_regularization_delta
  KKT dynamic regularization shift (2e-7)
iterative_refinement_enable
  KKT solve with iterative refinement (TRUE)
iterative_refinement_reltol
  iterative refinement relative tolerance (1e-12)
iterative_refinement_abstol
  iterative refinement absolute tolerance (1e-12)
iterative_refinement_max_iter
  iterative refinement maximum iterations (10L)
iterative_refinement_stop_ratio
    iterative refinement stalling tolerance (5.0)

presolve_enable
    whether to enable presolve (TRUE)

Value
    a list containing the control parameters.

---

solver_status_descriptions

*Return the solver status description as a named character vector*

Description
   Return the solver status description as a named character vector

Usage
    solver_status_descriptions()

Value
    a named list of solver status descriptions, in order of status codes returned by the solver

Examples
    solver_status_descriptions()[2] ## for solved problem
    solver_status_descriptions()[8] ## for max iterations limit reached
Index

* package
  clarabel-package, 2

clarabel, 2
clarabel-package, 2
clarabel_control, 4
clarabel_control(), 4

solver_status_descriptions, 7