The quadprog Package

July 13, 2007

Version 1.4-11

Date 2007-07-12

Title Functions to solve Quadratic Programming Problems.

Author S original by Berwin A. Turlach <berwin.turlach@anu.edu.au> R port by Andreas Weingessel <Andreas.Weingessel@ci.tuwien.ac.at>

Maintainer Andreas Weingessel <Andreas.Weingessel@ci.tuwien.ac.at>

Description This package contains routines and documentation for solving quadratic programming problems.

License GPL-2

R topics documented:

solve.QP ................................................................. 1
solve.QP.compact ...................................................... 3

Index 5

---

solve.QP Solve a Quadratic Programming Problem

Description

This routine implements the dual method of Goldfarb and Idnani (1982, 1983) for solving quadratic programming problems of the form \( \min(-d^T b + 1/2 b^T Db) \) with the constraints \( A^T b \geq b_0 \).

Usage

solve.QP(Dmat, dvec, Amat, bvec, meq=0, factorized=FALSE)
Arguments

- **Dmat**: matrix appearing in the quadratic function to be minimized.
- **dvec**: vector appearing in the quadratic function to be minimized.
- **Amat**: matrix defining the constraints under which we want to minimize the quadratic function.
- **bvec**: vector holding the values of $b_0$ (defaults to zero).
- **meq**: the first `meq` constraints are treated as equality constraints, all further as inequality constraints (defaults to 0).
- **factorized**: logical flag: if `TRUE`, then we are passing $R^{-1}$ (where $D = R^T R$) instead of the matrix $D$ in the argument `Dmat`.

Value

- a list with the following components:
  - **solution**: vector containing the solution of the quadratic programming problem.
  - **value**: scalar, the value of the quadratic function at the solution.
  - **unconstrained.solution**: vector containing the unconstrained minimizer of the quadratic function.
  - **iterations**: vector of length 2, the first component contains the number of iterations the algorithm needed, the second indicates how often constraints became inactive after becoming active first. vector with the indices of the active constraints at the solution.

References


See Also

`solve.QP.compact`

Examples

```r
# Assume we want to minimize: -(0 5 0) %*% b + 1/2 b^T b
# under the constraints: A^T b >= b0
# with b0 = (-8,2,0)^T
# and
# A = (-4 2 0)
# ( -3 1 -2)
# ( 0 0 1)
# we can use solve.QP as follows:
#
Dmat <- matrix(0,3,3)
```
solve.QP.compact

```r
diag(Dmat) <- 1
dvec <- c(0,5,0)
Amat <- matrix(c(-4,-3,0,2,1,0,0,-2,1),3,3)
bvec <- c(-8,2,0)
solve.QP(Dmat,dvec,Amat,bvec=bvec)
```

---

**solve.QP.compact**   Solve a Quadratic Programming Problem

### Description

This routine implements the dual method of Goldfarb and Idnani (1982, 1983) for solving quadratic programming problems of the form

\[
\min (-d^T b + 1/2 b^T D b)
\]

with the constraints \( A^T b \geq b_0 \).

### Usage

```r
solve.QP.compact(Dmat, dvec, Amat, Aind, bvec, meq=0, factorized=FALSE)
```

### Arguments

- **Dmat**: matrix appearing in the quadratic function to be minimized.
- **dvec**: vector appearing in the quadratic function to be minimized.
- **Amat**: matrix containing the non-zero elements of the matrix \( A \) that defines the constraints. If \( m_i \) denotes the number of non-zero elements in the \( i \)-th column of \( A \) then the first \( m_i \) entries of the \( i \)-th column of \( Amat \) hold these non-zero elements. (If \( \max m_i \) denotes the maximum of all \( m_i \), then each column of \( Amat \) may have arbitrary elements from row \( m_i+1 \) to row \( \max m_i \) in the \( i \)-th column.)
- **Aind**: matrix of integers. The first element of each column gives the number of non-zero elements in the corresponding column of the matrix \( A \). The following entries in each column contain the indexes of the rows in which these non-zero elements are.
- **bvec**: vector holding the values of \( b_0 \) (defaults to zero).
- **meq**: the first \( \text{meq} \) constraints are treated as equality constraints, all further as inequality constraints (defaults to 0).
- **factorized**: logical flag: if TRUE, then we are passing \( R^{-1} \) (where \( D = R^T R \)) instead of the matrix \( D \) in the argument \( Dmat \).

### Value

A list with the following components:

- **solution**: vector containing the solution of the quadratic programming problem.
- **value**: scalar, the value of the quadratic function at the solution
- **unconstrained.solution**: vector containing the unconstrained minimizer of the quadratic function.
iterations vector of length 2, the first component contains the number of iterations the algorithm needed, the second indicates how often constraints became inactive after becoming active first. vector with the indices of the active constraints at the solution.

References


See Also

solve.QP

Examples

```r
# Assume we want to minimize: -(0 5 0) %*% b + 1/2 b^T b
# under the constraints: A^T b >= b0
# with b0 = (-8,2,0)^T
# and
# A = (-3 1 -2)
# ( 0 0 1)
# we can use solve.QP.compact as follows:
#
# Dmat <- matrix(0,3,3)
diag(Dmat) <- 1
dvec <- c(0,5,0)
Aind <- rbind(c(2,2,2),c(1,1,2),c(2,2,3))
Amat <- rbind(c(-4,2,-2),c(-3,1,0))
bvec <- c(-8,2,0)
solve.QP.compact(Dmat,dvec,Amat,Aind,bvec=bvec)
```
Index

*Topic optimize
  solve.QP, 1
  solve.QP.compact, 3

solve.QP, 1, 4
solve.QP.compact, 2, 3