The LowRankQP Package

February 21, 2006

Version 1.0

Date 2005-09-14

Title Low Rank Quadratic Programming

Author J. T. Ormerod <jormerod@maths.unsw.edu.au> M. P. Wand <matt@maths.unsw.edua.au>

Maintainer J. T. Ormerod <jormerod@maths.unsw.edu.au>

Description This package contains routines and documentation for solving quadratic programming problems where the hessian is represented as the product of two matrices.

R topics documented:

`LowRankQP` *Solve Low Rank Quadratic Programming Problems*

Description

This routine implements a primal-dual interior point method solving quadratic programming problems of the form

$$\begin{array}{ll} \text{min} & d^T alpha + 1/2 alpha^T H alpha \\ \text{such that} & A alpha = b \\ & 0 <= alpha <= u \end{array}$$

with dual

$$\begin{array}{ll} \text{min} & 1/2 alpha^T H alpha + beta^T b + xi^T u \\ \text{such that} & H alpha + c + A^T beta - zeta + xi = 0 \\ & xi, zeta >= 0 \end{array}$$

where $H = V$ if V is square and $H = VV^T$ otherwise.

Usage

```
LowRankQP(Vmat,dvec,Amat,bvec,uvec,method="PFCF",verbose=FALSE,niter=200)
```

Arguments

`Vmat`	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 (defaults to zero).
`uvec`	vector holding the values of u .
`method`	Method used for inverting H+D where D is full rank diagonal. If V is square: • 'LU': Use LU factorization. (More stable) • 'CHOL': Use Cholesky factorization. (Faster) If V is not square: • 'SMW': Use Sherman-Morrison-Woodbury (Faster) • 'PFCF': Use Product Form Cholesky Factorization (More stable)
`verbose`	Display iterations of LowRankQP.
`niter`	Number of iteration to perform.

Value

a list with the following components:

`alpha`	vector containing the solution of the quadratic programming problem.
`beta`	vector containing the solution of the dual of quadratic programming problem.
`xi`	vector containing the solution of the dual quadratic programming problem.
`zeta`	vector containing the solution of the dual quadratic programming problem.

References

Ormerod, J.T., Wand, M.P. and Koch, I. (2005). Penalised spline support vector classifiers: computational issues, in A.R. Francis, K.M. Matawie, A. Oshlack, G.K. Smyth (eds). Proceedings of the 20th International Workshop on Statistical Modelling, Sydney, Australia, pp. 33-47.

Boyd, S. and Vandenberghe, L. (2004). Convex Optimization. Cambridge University Press.

Ferris, M. C. and Munson, T. S. (2003). Interior point methods for massive support vector machines. SIAM Journal on Optimization, 13, 783-804.

Fine, S. and Scheinberg, K. (2001). Efficient SVM training using low-rank kernel representations. Journal of Machine Learning Research, 2, 243-264.

B. Schölkopf and A. J. Smola. (2002). Learning with Kernels. The MIT Press, Cambridge, Massachusetts.

See Also

Examples

```
library(LowRankQP)

# Assume we want to minimize: (0 -5 0 0 0 0) %*% alpha + 1/2 alpha[1:3]^T alpha[1:3]
# under the constraints:      A^T alpha = b
# with b = (-8,  2,  0 )^T
# and      (-4   2   0 )
#      A = (-3   1  -2 )
#          ( 0   0   1 )
#          (-1   0   0 )
#          ( 0  -1   0 )
#          ( 0   0  -1 )
#  alpha >= 0
#
# (Same example as used in quadprog)
#
# we can use LowRankQP as follows:

Vmat           <- matrix(0,6,6)
diag(Vmat)     <- c(1, 1,1,0,0,0)
dvec           <- c(0,-5,0,0,0,0)
Amat           <- matrix(c(-4,-3,0,-1,0,0,2,1,0,0,-1,0,0,-2,1,0,0,-1),6,3)
bvec           <- c(-8,2,0)
uvec           <- c(100,100,100,100,100,100)
LowRankQP(Vmat,dvec,t(Amat),bvec,uvec,method="CHOL")

# Now solve the same problem except use low-rank V

Vmat           <- matrix(c(1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0),6,3)
dvec           <- c(0,-5,0,0,0,0)
Amat           <- matrix(c(-4,-3,0,-1,0,0,2,1,0,0,-1,0,0,-2,1,0,0,-1),6,3)
bvec           <- c(-8,2,0)
uvec           <- c(100,100,100,100,100,100)
LowRankQP(Vmat,dvec,t(Amat),bvec,uvec,method="SMW")
```

Index