# Package ‘Rdsdp’

January 20, 2025

**Version** 1.0.5.2.1

**Title** R Interface to DSDP Semidefinite Programming Library

**Author** Zhisu Zhu, Yinyu Ye (DSDP by Steve Benson, Yinyu Ye and Xiong Zhang)

**Maintainer** Zhisu Zhu <zhuzhisu@alumni.stanford.edu>

**Description** R interface to DSDP semidefinite programming library. The DSDP software is a free open source implementation of an interior-point method for semidefinite programming. It provides primal and dual solutions, exploits low-rank structure and sparsity in the data, and has relatively low memory requirements for an interior-point method.

**Imports** utils, methods

**LazyLoad** yes

**License** GPL-3

**URL** https://www.mcs.anl.gov/hs/software/DSDP

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2023-05-19 07:18:42 UTC

## Contents

Rdsdp *R interface to DSDP semidefinite programming library*

## Description

Rdsdp is the R package providing a R interface to DSDP semidefinite programming library. The DSDP package implements a dual-scaling algorithm to find solutions $(X, y)$ to linear and semidefinite optimization problems of the form

$$\text{(P)} \;\; \inf \; \text{tr}(CX)$$

$$\text{subject to } \mathcal{A}X = b$$

$$X \succeq 0$$

with $(\mathcal{A}X)_i = \text{tr}(A_i X)$ where $X \succeq 0$ means X is positive semidefinite, $C$ and all $A_i$ are symmetric matrices of the same size and $b$ is a vector of length $m$.

The dual of the problem is

$$\text{(D)} \;\; \sup \; b^T y$$

$$\text{subject to } \mathcal{A}^* y + S = C$$

$$S \succeq 0$$

where $\mathcal{A}y = \sum_{i=1}^m y_i A_i$.

Matrices $C$ and $A_i$ are assumed to be block diagonal structured, and must be specified that way (see Details).

## References


- https://www.mcs.anl.gov/hs/software/DSDP/
- Steven J. Benson and Yinyu Ye:
  *Algorithm 875: DSDP5 software for semidefinite programming* ACM Transactions on Mathematical Software (TOMS) 34(3), 2008
  http://web.stanford.edu/~yyye/DSDP5-Paper.pdf
- Steven J. Benson and Yinyu Ye and Xiong Zhang:
  *Solving Large-Scale Sparse Semidefinite Programs for Combinatorial Optimization* SIAM Journal on Optimization 10(2):443-461, 2000
  http://web.stanford.edu/~yyye/yyye/largesdp.ps.gz

Rdsdp::dsdp *Solve semidefinite programm with DSDP*

## Description

Interface to DSDP semidefinite programming library.

## Usage

```
dsdp(A,b,C,K,OPTIONS=NULL)
```

## Arguments

`A` An object of class `"matrix"` with $m$ rows defining the block diagonal constraint matrices $A_i$. Each constraint matrix $A_i$ is specified by a row of $A$ as explained in the Details section.

`b` A numeric vector of length $m$ containg the right hand side of the constraints.

`C` An object of class `"matrix"` with one row or a valid class from the class hierarchy in the `"Matrix"` package. It defines the objective coefficient matrix $C$ with the same structure of `A` as explained above.

`K` Describes the sizes of each block of the sdp problem. It is a list with the following elements:

- **`"s"`:** A vector of integers listing the dimension of positive semidefinite cone blocks.
- **`"l"`:** A scaler integer indicating the dimension of the linear nonnegative cone block.

`OPTIONS` A list of OPTIONS parameters passed to dsdp. It may contain any of the following fields:

**print:** `= k` to display output at each k iteration, else = 0 [default 10].

**logsummary:** `= 1` print timing information if set to 1.

**save:** to set the filename to save solution file in SDPA format.

**outputstats:** `= 1` to output full information about the solution statistics in STATS.

**gaptol:** tolerance for duality gap as a fraction of the value of the objective functions [default 1e-6].

**maxit:** maximum number of iterations allowed [default 1000].

Please refer to DSDP User Guide for additional OPTIONS parameters available.

## Details

All problem matrices are assumed to be of block diagonal structure, the input matrix `A` must be specified as follows:

1. The coefficients for nonnegative cone block are put in the first `K$l` columns of `A`.

2. The coefficients for positive semidefinite cone blocks are put after nonnegative cone block in the the same order as those in `K$s`. The `i`th positive semidefinite cone block takes `(K$s)[i]` times `(K$s)[[i]]` columns, with each row defining a symmetric matrix of size `(K$s)[[i]]`.

This function does not check for symmetry in the problem data.

## Value

Returns a list of three objects:

`X` Optimal primal solution $X$. A vector containing blocks in the same structure as explained above.

`y` Optimal dual solution $y$. A vector of the same length as argument `b`

`STATS` A list with three to eight fields that describe the solution of the problem:

**stype:** `PDFeasible` if the solutions to both (D) and (P) are feasible, `Infeasible` if (D) is infeasible, and `Unbounded` if (D) is unbounded.

**dobj:** objective value of (D).

**pobj:** objective value of (P).

**r:** the multiple of the identity element added to $C - \mathcal{A}^* y$ in the final solution to make $S$ positive definite.

**mu:** the final barrier parameter $\nu$.

**pstep:** the final step length in (P)

**dstep:** the final step length in (D)

**pnorm:** the final value $\|P(\nu)\|$.

The last five fields are optional, and only available when `OPTIONS$outputstats` is set to $1$.

## References


- Steven J. Benson and Yinyu Ye:
  *DSDP5 User Guide — Software for Semidefinite Programming* Technical Report ANL/MCS-TM-277, 2005
  https://www.mcs.anl.gov/hs/software/DSDP/DSDP5-Matlab-UserGuide.pdf


## Examples

```
K=NULL
K$s=c(2,3)
K$l=2

C=matrix(c(0,0,2,1,1,2,c(3,0,1,
                       0,2,0,
                       1,0,3)),1,15,byrow=TRUE)
A=matrix(c(0,1,0,0,0,0,c(3,0,1,
                         0,4,0,
                         1,0,5),
           1,0,3,1,1,3,rep(0,9)), 2,15,byrow=TRUE)
b <- c(1,2)
```

```
OPTIONS=NULL
OPTIONS$gaptol=0.000001
OPTIONS$logsummary=0
OPTIONS$outputstats=1

result = dsdp(A,b,C,K,OPTIONS)
```

---

`Rdsdp::dsdp.readsdpa` *Solving semidefinite programs reading from SDPA format files.*

---

## Description

Function to read the semidefinite program input data in SDPA format and solve it.

## Usage

```
dsdp.readsdpa(sdpa_filename, options_filename="")
```

## Arguments

`sdpa_filename` The location of the SDPA input file.

`options_filename`
The location of the OPTIONS file [default ""].

# Index