

# Package: sdpt3r (via r-universe)

November 25, 2024

**Type** Package

**Title** Semi-Definite Quadratic Linear Programming Solver

**Version** 0.3

**Date** 2019-02-08

**Author** Kim-Chuan Toh (Matlab/C), Micheal Todd (Matlab/C), Reha Tutunco (Matlab/C), Adam Rahman (R/C Headers), Timothy A. Davis (symamd C code), Stefan I. Larimore (symamd C code)

**Maintainer** Adam Rahman <a45rahma@uwaterloo.ca>

**Description** Solves the general Semi-Definite Linear Programming formulation using an R implementation of SDPT3 (K.C. Toh, M.J. Todd, and R.H. Tutunco (1999) <doi:10.1080/10556789908805762>). This includes problems such as the nearest correlation matrix problem (Higham (2002) <doi:10.1093/imanum/22.3.329>), D-optimal experimental design (Smith (1918) <doi:10.2307/2331929>), Distance Weighted Discrimination (Marron and Todd (2012) <doi:10.1198/016214507000001120>), as well as graph theory problems including the maximum cut problem. Technical details surrounding SDPT3 can be found in R.H Tutunco, K.C. Toh, and M.J. Todd (2003) <doi:10.1007/s10107-002-0347-5>.

**License** GPL-2 | GPL-3

**Depends** Matrix, R (>= 2.10)

**RoxygenNote** 6.1.1

**NeedsCompilation** yes

**Imports** methods, stats

**Date/Publication** 2019-02-11 08:50:03 UTC

**Additional\_repositories** <https://cranhaven.r-universe.dev>

**Repository** <https://cranhaven.r-universe.dev>

**RemoteUrl** <https://github.com/cranhaven/cranhaven.r-universe.dev>

**RemoteRef** package/sdpt3r

**RemoteSha** 54d76560486abed24f22a578ed4dd1eceed5435e

## Contents

Andwd . . . . .	2
Apdwd . . . . .	3
Betp . . . . .	3
Bgpp . . . . .	3
Blogcheby . . . . .	4
Bmaxcut . . . . .	4
Bmaxkcut . . . . .	4
control_theory . . . . .	5
DoptDesign . . . . .	6
doptimal . . . . .	6
dwd . . . . .	7
etp . . . . .	8
flogcheby . . . . .	9
Ftoep . . . . .	9
Glovasz . . . . .	9
gpp . . . . .	10
Hnearcorr . . . . .	11
lmi1 . . . . .	11
lmi2 . . . . .	12
lmi3 . . . . .	13
logcheby . . . . .	14
lovasz . . . . .	15
maxcut . . . . .	15
maxkcut . . . . .	16
minelips . . . . .	17
nearcorr . . . . .	18
smat . . . . .	19
sqlp . . . . .	19
svec . . . . .	21
toep . . . . .	21
Vminelips . . . . .	22
<b>Index</b>	<b>23</b>

---

Andwd

*An Configuration Matrix for Distance Weighted Discrimination*

---

### Description

An Configuration Matrix for Distance Weighted Discrimination

### Usage

data(Andwd)

**Format**

A matrix with 50 rows and 3 columns

---

Apdwd

*Ap Configuration Matrix for Distance Weighted Discrimination*

---

**Description**

Ap Configuration Matrix for Distance Weighted Discrimination

**Usage**

data(Apdwd)

**Format**

A matrix with 50 rows and 3 columns

---

Betp

*Symmetric Matrix for Educational Testing Problem*

---

**Description**

Symmetric Matrix for Educational Testing Problem

**Usage**

data(Betp)

**Format**

A matrix with 5 rows and 5 columns

---

Bgpp

*Adjacency Matrix for Graph Partitioning Problem*

---

**Description**

Adjacency Matrix for Graph Partitioning Problem

**Usage**

data(Bgpp)

**Format**

A matrix with 10 rows and 10 columns

Blogcheby

*B Matrix for the Log Chebyshev Approximation Problem*

---

**Description**

B Matrix for the Log Chebyshev Approximation Problem

**Usage**

data(Blogcheby)

**Format**A matrix with 10 rows and 10 columns

---

Bmaxcut

*Adjacency Matrix for Max-Cut*

---

**Description**

Adjacency Matrix for Max-Cut

**Usage**

data(Bmaxcut)

**Format**A matrix with 10 rows and 10 columns

---

Bmaxkcut

*Adjacency Matrix for Max-kCut*

---

**Description**

Adjacency Matrix for Max-kCut

**Usage**

data(Bmaxkcut)

**Format**

A matrix with 10 rows and 10 columns

---

control_theory	<i>Control Theory</i>
----------------	-----------------------

---

## Description

control\_theory creates input for sqlp to solve the Control Theory Problem

## Usage

```
control_theory(B)
```

## Arguments

B a matrix object containing square matrices of size n

## Details

Solves the control theory problem. Mathematical and implementation details can be found in the vignette

## Value

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

## Examples

```
B <- matrix(list(),2,1)
B[[1]] <- matrix(c(-.8,1.2,-.5,-1.1,-1,-2.5,2,.2,-1),nrow=3,byrow=TRUE)
B[[2]] <- matrix(c(-1.5,.5,-2,1.1,-2,.2,-1.4,1.1,-1.5),nrow=3,byrow=TRUE)

out <- control_theory(B)
```

---

DoptDesign

*Test Vector Matrix for D-Optimal Design*


---

**Description**

Test Vector Matrix for D-Optimal Design

**Usage**

data(DoptDesign)

**Format**

A matrix with 3 rows and 25 columns

---

doptimal

*D-Optimal Experimental Design*


---

**Description**

doptimal creates input for sqlp to solve the D-Optimal Experimental Design problem - given an  $n \times p$  matrix with  $p \leq n$ , find the portion of points that maximizes  $\det(A'A)$

**Usage**

doptimal(V)

**Arguments**

V a  $p \times n$  matrix containing a set of  $n$  test vectors in dimension  $p$  (with  $p \leq n$ )

**Details**

Solves the D-optimal experimental design problem. Mathematical and implementation details can be found in the vignette

**Value**

X A list containing the solution matrix to the primal problem  
y A list containing the solution vector to the dual problem  
Z A list containing the solution matrix to the dual problem  
pobj The achieved value of the primary objective function  
dobj The achieved value of the dual objective function

**Examples**

```
data(DoptDesign)

out <- doptimal(DoptDesign)
```

dwd

*Distance Weighted Discrimination***Description**

dwd creates input for `sqlp` to solve the Distance Weighted Discrimination problem - Given two sets of points  $A_n$  and  $A_p$ , find an optimal classification rule to group the points as accurately as possible for future classification.

**Usage**

```
dwd(Ap, An, penalty)
```

**Arguments**

Ap	An nxp point configuration matrix
An	An nxp point configuration matrix
penalty	A real valued scalar penalty for moving points across classification rule

**Details**

Solves the distance weighted discrimination problem. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
data(Andwd)
data(Apdwd)
penalty <- 0.5

#Not Run
#out <- dwd(Apdwd, Andwd, penalty)
```

---

`etp`*Educational Testing Problem*

---

**Description**

`etp` creates input for `sqlp` to solve the Educational Testing Problem - given a symmetric positive definite matrix  $S$ , how much can be subtracted from the diagonal elements of  $S$  such that the resulting matrix is positive semidefinite.

**Usage**`etp(B)`**Arguments**

`B`                    A symmetric positive definite matrix

**Details**

Solves the education testing problem. Mathematical and implementation details can be found in the vignette

**Value**

`X`                    A list containing the solution matrix to the primal problem  
`y`                    A list containing the solution vector to the dual problem  
`Z`                    A list containing the solution matrix to the dual problem  
`pobj`                The achieved value of the primary objective function  
`dobj`                The achieved value of the dual objective function

**Examples**

```
data(Betp)
out <- etp(Betp)
```



---

flogcheby	<i>f vector for the Log Chebyshev Approximation Problem</i>
-----------	---

---

**Description**

f vector for the Log Chebyshev Approximation Problem

**Usage**

```
data(flogcheby)
```

**Format**

A vector with length 20

---

Ftoep	<i>Symmetric Matrix for the Toeplitz Approximatin Problem</i>
-------	---

---

**Description**

Symmetric Matrix for the Toeplitz Approximatin Problem

**Usage**

```
data(Ftoep)
```

**Format**

A matrix with 10 rows and 10 columns

---

Glovasz	<i>Adjacency Matrix on which to find the Lovasz Number</i>
---------	--

---

**Description**

Adjacency Matrix on which to find the Lovasz Number

**Usage**

```
data(Glovasz)
```

**Format**

A matrix with 10 rows and 10 columns

---

`gpp`*Graph Partitioning Problem*

---

**Description**

`gpp` creates input for `sqlp` to solve the graph partitioning problem.

**Usage**

```
gpp(B, alpha)
```

**Arguments**

<code>B</code>	A weighted adjacency matrix
<code>alpha</code>	Any real value in $(0, n^2)$

**Details**

Solves the graph partitioning problem. Mathematical and implementation details can be found in the vignette

**Value**

<code>X</code>	A list containing the solution matrix to the primal problem
<code>y</code>	A list containing the solution vector to the dual problem
<code>Z</code>	A list containing the solution matrix to the dual problem
<code>pobj</code>	The achieved value of the primary objective function
<code>dobj</code>	The achieved value of the dual objective function

**Examples**

```
data(Bgpp)
alpha <- nrow(Bgpp)

out <- gpp(Bgpp, alpha)
```

---

Hnearcorr	<i>Approximate Correlation Matrix for Nearest Correlation Matrix Problem</i>
-----------	--

---

**Description**

Approximate Correlation Matrix for Nearest Correlation Matrix Problem

**Usage**

data(Hnearcorr)

**Format**

A matrix with 5 rows and 5 columns

---

lmi1	<i>Linear Matrix Inequality 1</i>
------	-----------------------------------

---

**Description**

lmi1 creates input for sqp to solve a linear matrix inequality problem

**Usage**

lmi1(B)

**Arguments**

B                    An mxn real valued matrix

**Details**

Solves the type-1 linear matrix inequality problem. Mathematical and implementation details can be found in the vignette

**Value**

x	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
B <- matrix(c(-1,5,1,0,-2,1,0,0,-1), nrow=3)

#Not Run
#out <- lmi1(B)
```

---

lmi2

---

*Linear Matrix Inequality 2*


---

**Description**

lmi2 creates input for sqp to solve a linear matrix inequality problem

**Usage**

```
lmi2(A1, A2, B)
```

**Arguments**

A1	An nxm real valued matrix
A2	An nxm real valued matrix
B	An nxp real valued matrix

**Details**

Solves the type-2 linear matrix inequality problem. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
A1 <- matrix(c(-1,0,1,0,-2,1,0,0,-1),3,3)
A2 <- A1 + 0.1*t(A1)
B <- matrix(c(1,3,5,2,4,6),3,2)

out <- lmi2(A1,A2,B)
```

**Description**

lmi3 creates input for sqlp to solve a linear matrix inequality problem

**Usage**

```
lmi3(A, B, G)
```

**Arguments**

A	An nxn real valued matrix
B	An mxn real valued matrix
G	An nxn real valued matrix

**Details**

Solves the type-3 linear matrix inequality problem. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
A <- matrix(c(-1,0,1,0,-2,1,0,0,-1),3,3)
B <- matrix(c(1,2,3,4,5,6), 2, 3)
G <- matrix(1,3,3)

out <- lmi3(A,B,G)
```

---

`logcheby`*Log Chebyshev Approximation*

---

**Description**

logcheby creates input for `sqlp` to solve the Chebyshev Approximation Problem

**Usage**

```
logcheby(B, f)
```

**Arguments**

B	A pxm real valued matrix with $p > m$
f	A vector of length p

**Details**

Solves the log Chebyshev approximation problem. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
data(Blogcheby)
data(flogcheby)

#Not Run
#out <- logcheby(Blogcheby, flogcheby)
```

---

lovasz	<i>Lovasz Number of a Graph</i>
--------	---------------------------------

---

**Description**

lovasz creates input for sqlp to find the Lovasz Number of a graph

**Usage**

```
lovasz(G)
```

**Arguments**

G                    An adjacency matrix corresponding to a graph

**Details**

Finds the maximum Shannon entropy of a graph, more commonly known as the Lovasz number. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
data(Glovasz)

out <- lovasz(Glovasz)
```

---

maxcut	<i>Max-Cut Problem</i>
--------	------------------------

---

**Description**

maxcut creates input for sqlp to solve the Max-Cut problem - given a graph B, find the maximum cut of the graph

**Usage**

```
maxcut(B)
```

**Arguments**

B                    A (weighted) adjacency matrix corresponding to a graph

**Details**

Determines the maximum cut for a graph B. Mathematical and implementation details can be found in the vignette

**Value**

X                    A list containing the solution matrix to the primal problem  
 y                    A list containing the solution vector to the dual problem  
 Z                    A list containing the solution matrix to the dual problem  
 pobj                The achieved value of the primary objective function  
 dobj                The achieved value of the dual objective function

**Examples**

```
data(Bmaxcut)

out <- maxkcut(Bmaxcut)
```

---

 maxkcut

*Max-kCut Problem*


---

**Description**

maxkcut creates input for sqlp to solve the Max-kCut Problem - given a graph object B, determine if a cut of at least size k exists.

**Usage**

```
maxkcut(B, K)
```

**Arguments**

B                    A (weighted) adjacency matrix  
 K                    An integer value, the minimum number of cuts in B

**Details**

Determines if a cut of at least size k exists for a graph B. Mathematical and implementation details can be found in the vignette



**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
data(Bmaxkcut)
out <- maxkcut(Bmaxkcut,2)
```

---

minelips	<i>The Minimum Ellipsoid Problem</i>
----------	--------------------------------------

---

**Description**

minelips creates input for `sqlp` to solve the minimum ellipsoid problem - given a set of  $n$  points, find the minimum volume ellipsoid that contains all the points

**Usage**

```
minelips(V)
```

**Arguments**

V An  $n \times p$  matrix consisting of the points to be contained in the ellipsoid

**Details**

for a set of points  $(x_1, \dots, x_n)$  determines the ellipse of minimum volume that contains all points. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
data(Vminelips)

#Not Run
#out <- minelips(Vminelips)
```

---

nearcorr

*Nearest Correlation Matrix Problem*


---

**Description**

nearcorr creates input for `sqp` to solve the nearest correlation matrix problem - given a approximate correlation matrix  $H$ , find the nearest correlation matrix  $X$ .

**Usage**

```
nearcorr(H)
```

**Arguments**

$H$                     A symmetric matrix

**Details**

For a given approximate correlation matrix  $H$ , determines the nearest correlation matrix  $X$ . Mathematical and implementation details can be found in the vignette

**Value**

$X$                     A list containing the solution matrix to the primal problem  
 $y$                     A list containing the solution vector to the dual problem  
 $Z$                     A list containing the solution matrix to the dual problem  
`pobj`                The achieved value of the primary objective function  
`dobj`                The achieved value of the dual objective function

**Examples**

```
data(Hnearcorr)

out <- nearcorr(Hnearcorr)
```

---

smat *Create a Symmetrix Matrix*

---

### Description

smat takes a vector and creates a symmetrix matrix

### Usage

```
smat(blk, p, At, isspM = NULL)
```

### Arguments

blk	Lx2 matrix detailing the type of matrices ("s", "q", "l", "u"), and the size of each matrix
p	Row of blk to be used during matrix creation
At	vector to be turned into a symmetric matrix
isspM	if At is sparse, isspx = 1, 0 otherwise. Default is to assume M is dense.

### Value

M	A Symmetric Matrix
---	--------------------

### Examples

```
y <- c(1,0.00000279, 3.245, 2.140, 2.44, 2.321, 4.566)

blk <- matrix(list(),1,2)
blk[[1,1]] <- "s"
blk[[1,2]] <- 3

P <- smat(blk,1, y)
```

---

sqlp *Semidefinite Quadratic Linear Programming Solver*

---

### Description

sqlp solves a semidefinite quadratic linear programming problem using the SDPT3 algorithm of Toh et. al. (1999) returning both the primal solution X and dual solution Z.

### Usage

```
sqlp(blk = NULL, At = NULL, C = NULL, b = NULL, control = NULL,
     X0 = NULL, y0 = NULL, Z0 = NULL)
```

**Arguments**

<code>blk</code>	A named-list object describing the block diagonal structure of the SQLP data
<code>At</code>	A list object containing constraint matrices for the primal-dual problem
<code>C</code>	A list object containing the constant $c$ matrices in the primal objective function
<code>b</code>	A vector containing the right hand side of the equality constraints in the primal problem
<code>control</code>	A list object specifying the values of certain parameters. If not provided, default values are used
<code>X0</code>	An initial iterate for the primal solution variable $X$ . If not provided, an initial iterate is computed internally.
<code>y0</code>	An initial iterate for the dual solution variable $y$ . If not provided, an initial iterate is computed internally.
<code>Z0</code>	An initial iterate for the dual solution variable $Z$ . If not provided, an initial iterate is computed internally.

**Details**

A full mathematical description of the problem to be solved, details surrounding the input variables, and discussion regarding the output variables can be found in the accompanying vignette.

**Value**

<code>X</code>	A list containing the solution matrix to the primal problem
<code>y</code>	The solution vector to the dual problem
<code>Z</code>	A list containing the solution matrix to the dual problem
<code>pobj</code>	The achieved value of the primary objective function
<code>dobj</code>	The achieved value of the dual objective function

**References**

K.C. Toh, M.J. Todd, and R.H. Tutuncu, SDPT3 — a Matlab software package for semidefinite programming, *Optimization Methods and Software*, 11 (1999), pp. 545–581. R.H Tutuncu, K.C. Toh, and M.J. Todd, Solving semidefinite-quadratic-linear programs using SDPT3, *Mathematical Programming Ser. B*, 95 (2003), pp. 189–217.

**Examples**

```
blk = c("1" = 2)
C = matrix(c(1,1),nrow=1)
A = matrix(c(1,3,4,-1), nrow=2)
At = t(A)
b = c(12,10)
out = sqlp(blk,list(At),list(C),b)
```

---

svec *Upper Triangular Vectorization*

---

**Description**

svec takes the upper triangular matrix (including the diagonal) and vectorizes it column-wise.

**Usage**

```
svec(blk, M, isspx = NULL)
```

**Arguments**

blk	1x2 matrix detailing the type of matrix ("s", "q", "l", "u"), and the size of the matrix
M	matrix which is to be vectorized
isspx	if M is sparse, isspx = 1, 0 otherwise. Default is to assume M is dense.

**Value**

x	vector of upper triangular components of x
---	--

**Examples**

```
data(Hnearcorr)
blk <- matrix(list(),1,2)
blk[[1]] <- "s"
blk[[2]] <- nrow(Hnearcorr)

svec(blk,Hnearcorr)
```

---

toep *Toeplitz Approximation Problem*

---

**Description**

toep creates input for sqlp to solve the Toeplitz approximation problem - given a symmetric matrix F, find the nearest symmetric positive definite Toeplitz matrix.

**Usage**

```
toep(A)
```

**Arguments**

A	A symmetric matrix
---	--------------------

**Details**

For a symmetric matrix A, determines the closest Toeplitz matrix. Mathematical and implementation details can be found in the vignette

**Value**

X	A list containing the solution matrix to the primal problem
y	A list containing the solution vector to the dual problem
Z	A list containing the solution matrix to the dual problem
pobj	The achieved value of the primary objective function
dobj	The achieved value of the dual objective function

**Examples**

```
data(Ftoep)

#Not Run
#out <- toep(Ftoep)
```

---

Vminelips

---

*Configuration Matrix for Minimum Ellipse Problem*


---

**Description**

Configuration Matrix for Minimum Ellipse Problem

**Usage**

```
data(Vminelips)
```

**Format**

A matrix with 2 rows and 2 columns

# Index

## \* datasets

- Andwd, [2](#)
  - Apdwd, [3](#)
  - Betp, [3](#)
  - Bgpp, [3](#)
  - Blogcheby, [4](#)
  - Bmaxcut, [4](#)
  - Bmaxkcut, [4](#)
  - DoptDesign, [6](#)
  - flogcheby, [9](#)
  - Ftoep, [9](#)
  - Glovasz, [9](#)
  - Hnearcorr, [11](#)
  - Vminelips, [22](#)
- 
- Andwd, [2](#)
  - Apdwd, [3](#)
- 
- Betp, [3](#)
  - Bgpp, [3](#)
  - Blogcheby, [4](#)
  - Bmaxcut, [4](#)
  - Bmaxkcut, [4](#)
- 
- control\_theory, [5](#)
- 
- DoptDesign, [6](#)
  - doptimal, [6](#)
  - dwd, [7](#)
- 
- etp, [8](#)
- 
- flogcheby, [9](#)
  - Ftoep, [9](#)
- 
- Glovasz, [9](#)
  - gpp, [10](#)
- 
- Hnearcorr, [11](#)
- 
- lmi1, [11](#)
  - lmi2, [12](#)
  - lmi3, [13](#)
  - logcheby, [14](#)
  - lovasz, [15](#)
  - maxcut, [15](#)
  - maxkcut, [16](#)
  - minelips, [17](#)
  - nearcorr, [18](#)
  - smat, [19](#)
  - sqlp, [19](#)
  - svec, [21](#)
  - toep, [21](#)
  - Vminelips, [22](#)