Technical report CEDRIC RC391-2002 updated in 2010
http://cedric.cnam.fr

Géraud Delaporte, Sébastien Jouteau, Frédéric Roupin[1]

SDP_S:
A TOOL TO FORMULATE AND SOLVE SEMIDEFINITE RELAXATIONS FOR BIVALENT QUADRATIC PROBLEMS

Version 1.1

[1] *Institut d'Informatique d'Entreprise-CEDRIC*
18 allée Jean Rostand 91025 Evry cedex France
e-mail : roupin@iie.cnam.fr

Contents

SDP_S 1.1 MANUAL

Géraud Delaporte, Sébastien Jouteau, Frédéric Roupin

1. Introduction

SDP_S is a stand-alone program which formulates semidefinite programming (SDP) relaxations for any $0-1$ quadratic problem starting from a linear relaxation of the initial problem. Moreover, it can solve the SDP by using the Spectral Bundle Method of C. Helmberg [2], [3]. SDP_S is an implementation of the algorithm proposed by F. Roupin in [4]. One of the major advantages of SDP_S is that it requires no knowledge about semidefinite programming. Indeed, the input problem has just to be stated as a natural $0-1$ quadratic program:

$$(P\{0,1\}) \begin{cases} \text{Min or Max} & x^T A_0 x + b^T x + val & \\ \text{s.t.} & x^T A_i x + c_i^T x = (\text{or} \leq)\, d_i & i = 1, ..., m_1 \\ & c_i^T x = d_i & i = m_1, ..., m_2 \\ & d_i \leq c_i^T x \leq d'_i & i = m_2, ..., m \\ & x \in \{0,1\}^n & \end{cases}$$

where $1 \leq m_1 \leq m_2 \leq m$ and val is a real number. Note that some A_i $i \in \{0, ..., m\}$ can be equal to 0, and if it is true for all i in $\{0, ..., m\}$ then $(P\{0,1\})$ is a $0-1$ linear program. Moreover, some d_i or d'_i can be missing. More precisely, the semidefinite relaxation is obtained by following the rules (see [4] for details) in Table 1 starting from a linear relaxation P_L of $(P\{0,1\})$ (obviously some of the constraints in the first and second columns can be missing). Hence, the problem is seen as a linear relaxation of $(P\{0,1\})$.

2. How to use SDP_S

2.1. Installing and compiling SDP_S.

Install all the files by using the command "tar xvfz SDP_S.1.1.tar.gz". A directory SDP_S will be created. It contains: the source code directories "matrix", tools", "spectral" and "SDP_SOLVER", the documentation directory "docs", and a directory "examples" which contains several ready-to-be-used examples for SDP_S.

Second, to compile SDP_S, simply type "make" in the SDP_S directory. This will produce the stand-alone executable "SDP_S" located in SDP_S/, and a directory "Object" which contains all the object files. "Make clean" will erase all the object and executable files. Caution: the Makefile and SDP_S have been only tested on several Linux distributions. For other unix systems, you may have to modify the Makefile and/or the code. Please, read carefully all the documention files located in "SDP_S/docs".

2.2. Running SDP_S.

To obtain and solve a semidefinite relaxation (obtained by the recipe presented in [4]) of any bivalent quadratic problem, simply run SDP_S like this:

```
SDP_S <filename> [options]
```

where "filename" is the problem file (see Section 3). Options are:

- -fSB <filename> : The file "filename" will contain the problem in the SB format (for a latter use with the SB solver alone for instance).
- -fopt <filename> : Options contained in "filename" will be given to the SB solver. (See the manual "sbmanual.ps" for more details).
- -fres <filename> : The file "filename" will contain an output of the solving.
- -nosolve : the semidefinite program will not be solved, but a file written in the SB format will be created. This option is useful to test the syntax of the input files, or to use the SDP solver SB alone.
- -resume <filename> : resumes a old process using the resume file "filename".

P_L	$(SDP\{0,1\})$
$0 \leq x_i \leq 1 \, \forall i \in \{1,...,n\}$	$\begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \succeq 0$ $\mathbf{d}(X) = x$
$A \bullet X + c^T x = (\leq) d$	$A \bullet X + c^T x = (\leq) d$
$\forall i < j < k < n \in \{1,...,n\}$ $0 \leq X_{ij}$ $X_{ij} \leq x_i$ $X_{ij} \leq x_j$ $x_i + x_j - 1 \leq X_{ij}$ $X_{ij} + X_{ik} + X_{jk} + 1 \geq X_{ij} + X_{ik} + X_{jk} + 1$	$\forall i < j < k < n \in \{1,...,n\}$ $0 \leq X_{ij}$ $X_{ij} \leq x_i$ $X_{ij} \leq x_j$ $x_i + x_j - 1 \leq X_{ij}$ $X_{ij} + X_{ik} + X_{jk} + 1 \geq X_{ij} + X_{ik} + X_{jk} + 1$
$c^T x = d$	$cc^T \bullet X - 2dc^T x + d^2 = 0$ $\Leftrightarrow \left\vert \begin{array}{l} cc^T \bullet X = d^2 \\ c^T x = d \end{array} \right.$ or $c^T x = d$ $\sum_{j=1}^n c_j X_{ij} = d x_i \, i = 1,...,n$
$d' \leq c^T x \leq d$	$cc^T \bullet X - (d + d') \, c^T x + dd' \leq 0$ or $\sum_{j=1}^n c_j X_{ij} \leq d x_i \quad i = 1,...,n$ $d' x_i \leq \sum_{j=1}^n c_j X_{ij} \quad i = 1,...,n$ $\sum_{j=1}^n c_j \, (x_j - X_{ij}) \leq d \, (1 - x_i) \quad i = 1,...,n$ $d' \, (1 - x_i) \leq \sum_{j=1}^n c_j \, (x_j - X_{ij}) \quad i = 1,...,n$
$c^T x \leq d$	$cc^T \bullet X - \left(\sum_{i \, s.t. \, c_i < 0} c_i + d\right) c^T x + d \sum_{i \, s.t. \, c_i < 0} c_i \leq 0$

TABLE 1. *Rules of the scheme*

If the -fopt option is not used, or if there is no "filename.opt" file in the current directory, then a default option file is created with the options "-si 1 -sh 1". This file has the extension ".opt". You can modify it to add options for SB without using the -fopt option, since SDP_S tests if an option file exists in the current directory.

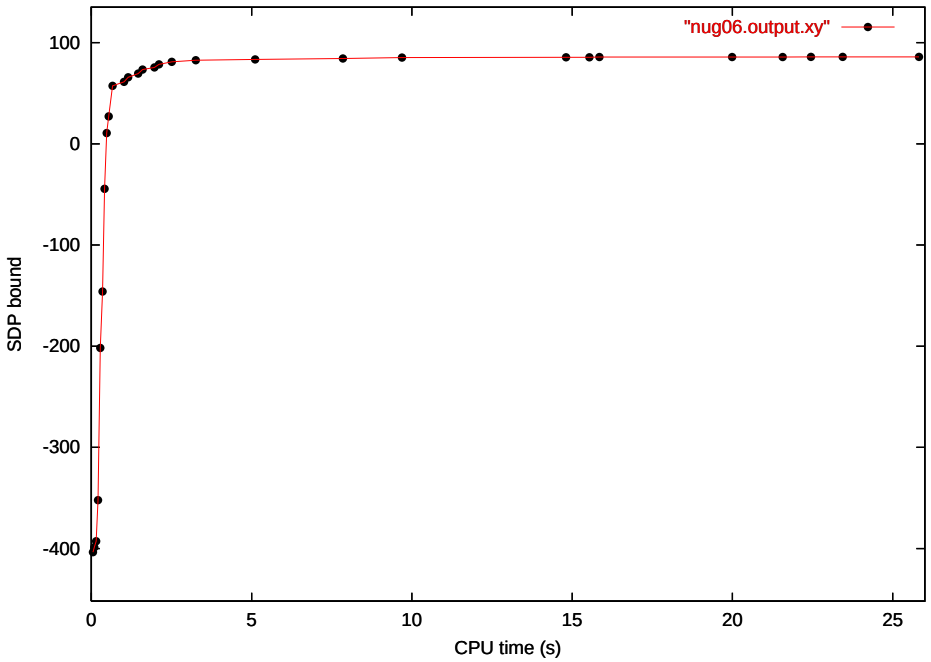
2.3. **Stopping SDP_S before normal termination.** The kill signals of SB can be used with SDP_S. These signals are :

(1) SIGTERM (15) : Save resume info, output summary, exit.
(2) SIGUSR2 (12) : Output summary and continue.
(3) SIGUSR1 (10) : Output summary, exit.
(4) SIGKILL (9) : Kill without any further output.

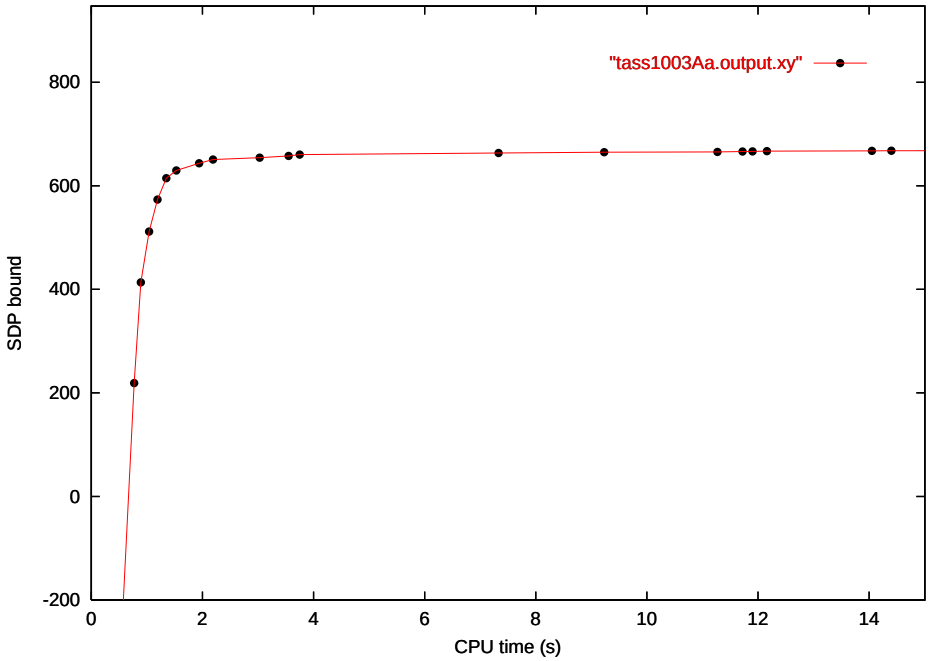
When the "SIGTERM (15)" is used, a resume file "SDPS_resume.dat" is created in the current directory (caution: it can be very large). This file can be used latter with the "-resume" option of SDP_S to continue the solving process. We do not recommend to use a "SIGKILL (9)" signal with SDP_S, especially if one has set the "-fres" option. Indeed, the output file may be incomplete. In all cases, the "SIGUSR1 (10)" and the "SIGTERM (15)" signals are the best ways to quit the program before normal termination.

2.4. **The res2plot utility.** Gnuplot and ghostview must be installed on your system to use the "res2plot" utility. The source code "res2plot.c" is located in SDP_S/tools. This program produces a postscript file from an output file produced by SDP_S (option "-fres"). The executable "res2Plot" is created with SDP_S in the "SDP_S/" directory when the "Make" command is used. res2Plot allows to visualize the convergence of the SDP solver. Several files are created in the current directory in order to output the graphical result. To obtain a better output it may be useful to change the ".plot" file to modify the ranges (x and y). Then gnuplot must be run again to create a new postscript file by using the command: "gnuplot filename.plot".

Example 2.1. This is the output obtained when running "res2plot nug06.output" in the SDP_S/examples/QAP directory:



Example 2.2. This is the output obtained when running "res2plot tass1003Aa.output" in the SDP_S/examples/QAP directory, then modifying the "tass1003Aa.output.plot" file by setting "set xrange [0:15]" "set yrange[-200:947]" and "set key 13,837" and running again gnuplot:



3. Input format

In this section we describe the syntax of the input files. The problem or "master" input file contains four main area:

(1) A commentary area.
(2) A quadratic or a linear function to maximize (or minimize).
(3) A general description of the problem (dimension, trace of the matrice if available, values of the bivalent variables, filenames describing the objective function).
(4) The constraints.

3.1. The commentary area.

It is possible to put some commentaries at the begin of the file. The commentaries must be written between two tokens "//". An example of commentary will be more explicit:

```
// This is a valid commentary //
```

Commentary can also be written on several lines:

```
// This is a //
// commentary //
// on several lines //
```

Or like that:

```
// This is
 a commentary
 on several lines //
```

But make sure that there is a space between the commentary and the tokens "//".

3.2. The objective function.

The objective function is the function to maximize or minimize. This area contains the following data:

(1) Maximization or Minimization.
(2) Does exist a quadratic part ?
(3) Does exist a linear part ?
(4) The constant value (which is not mandatory).

Caution: even if it is not mandatory to have a linear part *and* a quadratic part, there must be at least one of them.

Format

The general format of the objective function is:

$$< Min|Max >< (CX\ [+lx]\ [+'val']) \,|\, lx\ [+'val'] >$$

First, choose if you want to maximize or minimize. Keywords are Min or Max. Second, you must precise (in that order) if there are a quadratic part, a linear part and a constant. The quadratic part is precised by the presence of the character C followed by any character. The linear part is precised by the presence of the character l followed by any character. The constant is given by its value. Quadratic part, linear part and constant must be separated by the token "+".

Note that it must be the first data written after the commentary area. If the file do not begin with "Min or Max", an error will occur.

Example 3.1. One can write one of these objective functions :

```
* Max CX + 3
* Min lx + -5
* Max CX + lx and I can write what I want which does not begin with "+".
```

But not:

* `Max 3 :` *There is neither a quadratic or a linear part.*
* `Maximize CX :` *Maximize is not the same string as Max.*
* `Max CX - 5 :` *The character "+" must be written before the constant value, even if it is negative.*
* `Max 1x + CX :` *The quadratic part must be written before the linear part.*
* `# Max CX :` *The first word must be Min or Max.*

3.3. **Description of the problem.** In this part, the following informations are given:

- Dimension n of the problem (size of $x \in \{0,1\}^n$).
- The trace of the matrix $X = xx^T$ if it is a constant for the considered problem (it represents $\sum_{i=1}^n x_i = \sum_{i=1}^n X_{ii}$ because $\mathbf{d}(X) = x$). Since we work with $\{0,1\}$ variables, the trace of X is always bounded by n, but give a value to Trace(X) speeds up the convergence of the SDP solver SB [2]. This part is NOT mandatory.
- Values of the bivalent variables : $\{0,1\}$ or $\{-1,1\}$.
- Filenames for 'C' or/and 'l' (the quadratic and the linear parts of the objective function).

It is possible to put some commentaries between these informations. Caution : even if it is possible to use $\{-1,1\}$ variables in SDP_S, we *strongly* recommend to use $\{0,1\}$ variables instead. Indeed, several options are not available with the $\{-1,1\}$ model (for instance add triangle inequalities with the keyword "AddMoreConstr"). This is not restrictive since semidefinite relaxations obtained from the two bivalent models are equivalent (see for instance [4] for details).

Format

Each of the 6 labels that are possible has its own arguments:

(1) C : file < file type > : < filename >
(2) l : file < file type > : < filename >
(3) Dim : < Dimension of the problem >
(4) Trace : <Trace of the matrix X>
(5) Type : <$\{0,1\}$ | $\{-1,1\}$>
(6) AddMoreConstr : < integer in $\{0,...,31\}$>

Two different file types are possible to describe C and l : "Sparse" and "Dense":

- In the "Dense" format, "filename" must begin with the dimension of C followed by all the values of the matrix.
- In the "Sparse" format, "filename" begins by the dimension followed by the number of non-zero elements. Then, for each non-zero element, one has to write the 2 indices (row and column) followed by the value corresponding to these indices (indices start at one).

The "AddMoreConstr" option can be only used with the $\{0,1\}$ problems. Five kinds of constraints can be added:

$$\begin{cases} X_{ij} \geq 0 & 0 \leq i < j < n & (0) \\ X_{ij} \leq X_{ii} & 0 \leq i < j < n & (1) \\ X_{ij} \leq X_{jj} & 0 \leq i < j < n & \\ X_{ii} + X_{jj} \leq 1 + X_{ij} & 0 \leq i < j < n & (2) \\ X_{ik} + X_{jk} \leq X_{kk} + X_{ij} & 0 \leq i < j < n \text{ and } 0 \leq k < n & (3) \\ X_{ij} + X_{ik} + X_{jk} + 1 \geq X_{ii} + X_{jj} + X_{kk} & 0 \leq i < j < k < n & (4) \end{cases}$$

There is not yet such triangle inequalities implemented for the $\{-1,1\}$ problems. To choose the constraints, the "bit" of the corresponding constraints must be set to one. Then, the resulting integer number (belonging to $\{0,...,31\}$) must be written after "AddMoreConstr". For example, to add the constraints (0) and (2), the number must be equal to: $2^0 + 2^2 = 5$.

Example 3.2. Assuming that "FileC.dat" and "FileL.dat" exist, the following problem file is valid:

```
Max CX + lx + -4
C : File Sparse : FileC.dat
l : File Dense : FileL.dat
Dim : 23
Trace : 10
Type : {0,1}
AddMoreConstr : 5
```

3.4. **The constraints.** For this part, several labels can be used. In order to test quickly several different relaxations, one can disable constraints by putting a "_" at the beginning of the corresponding lines. The beginning of the constraints area is indicated by the label "Constraints:". For now, five labels of constraints are available:

- "File :"
- "Single :"
- "Sum :"
- "ProductConstrVect"
- "ProductConstrBarVect"
- "Simple"

3.4.1. *The "File" label.* This label indicates that the constraints are described in an external file. For instance : "File Sparse : constraints1.dat" or "File Dense : constraints1.dat" means that the constraints are described in the file "constraints1.dat".

Constraint File format:

```
< "Q" | "l" > < "=" | "<" | ">" | "<>" >
< dimension >
< number of constraints in this file >
< constant value of the first equality | inequality >
< vector/matrix of the first equality | inequality >
< constant value of the second equality | inequality >
< vector/matrix of the second equality | inequality >
...
< constant value of the last equality | inequality >
< vector/matrix of the last equality | inequality >
```

Description

First, one must precise if the constraints described in the file are linear or quadratic: the first character of the file will be a 'Q' for quadratic constraints or a 'l' for linear ones. Second, one must precise if the constraints are equality constraints or inequality constraints. In this last case only an upper bound can be written ('<' or '>'), or a lower and an upper bounds ('<>'). Hence, if m is the number of constraints, one must use:

(1) 'Q=' if the constraints that can be stated as "$B_i \bullet X + c_i^T x = d_i$" for $i \in \{1, ..., m\}$.
(2) 'l=' if the file contains constraints that can be stated as "$c_i^T x = d_i$" for $i \in \{1, ..., m\}$.
(3) 'Q<' if the constraints can be stated as "$B_i \bullet X + c_i^T x \leq d_i$" for $i \in \{1, ..., m\}$.
(4) 'l>' if the constraints can be stated as: "$c^T x \geq d_i$" for $i \in \{1, ..., m\}$.
(5) "l<>" if the constraints can be stated as: "$d_i' \leq c_i^T x \leq d_i$" for $i \in \{1, ..., m\}$.

Second, the dimension of x and the number m of constraints described in the file must be indicated. Third, for each constraint, the value of d and/or d' must be written:

(1) In the case '=', write d.
(2) In the cases '<' and '>', write d' or d.
(3) In the case "<>", write d' and d.

Finally, if the constraint is linear, then one has to write the vector (using the "Dense" or "Sparse" format according to the keyword written after the "File" label), else the constraint is quadratic, and the vector and the matrix have to be written (using the dense or spare format according to the keyword written after the "File" label). See the Section 4 for an example.

3.4.2. *The "Single :" label.* This label allows to write some constraints that will be treated like singletons. Here, no external file is needed.
Format:

```
Single : < i > < j > <operator> < value > [ range i ][ range j ]
```

Description:
Without using range variables, this label allows to write a constraint involving only one variable: "Single : 1 2 = 3.0" is to write $X_{12} = 3$. But if one wants to write many of constraints of this kind, range variables 'i' and 'j' can be used. The format of the range is: $\{a..b\}$ where 'a' and 'b' are integer or equal to 'n' (the size of $x \in \{0,1\}^n$). If the two indices have the same name, only one range have to be given. Caution: spaces are always forbidden between braces.

Example 3.3. The following examples are valid:

* `Single : 1 1 = 0 :` $X_{11} = x_1 = 0$.
* `Single : 1 j > 0` $\{1..2\}$: $X_{11} \geq 0$ *and* $X_{12} \geq 0$.
* `Single : i j > 0` $\{1..n\}$ $\{1..n\}$ *: All variables are non-negative.*
* `Single : i i = 1` $\{1..n\}$ *: All the diagonal elements are set to 1.*

But these examples are not valid:

* `Single : 1 1 < N :` *the value must be a number.*
* `Single : i j = 0` $\{1..n\}$ *: each variable must have a range.*
* `Single : 0 0 = 0 :` *indices start at one and not at zero.*
* `Single : i j = 0` $\{1..n\}$ $\{n..\ \ n\}$ *: spaces are forbidden between braces.*
* `Single : 1 2 ? 3 :` *The operator must be '=', '<' or '>'.*

3.4.3. *The "Sum :" label.* This label allows to write that the sum of some variables is equal to a constant number. To precise which kind of sum must be written, six keywords are allowed: "DiagSum", "VarSum", "RowSum", "SquareSum" ,"TriangSum" and "StrictTriangSum".

Format

```
DiagSum <operator> <value of the sum of the diagonal elements>
VarSum <operator> <value> {i..j}
RowSum <number of the line> <operator> <value of the row sum>
SquareSum {i1..i2} {j1..j2}<operator> <value of the sum>
TriangSum {i..j} <operator> <value of the sum >
StrictTriangSum {i..j} <operator> <value of the sum>
```

Examples

* `DiagSum = 2.0 :` $\sum_{i=1}^{n} X_{ii} = 2$
* `VarSum = 10` $\{1..n\}$ `:` $\sum_{i=1}^{n} x_i = 10 \Leftrightarrow \sum_{i=1}^{n} X_{n+1,i} = 10$
* `RowSum 2 < 3.0 :` $\sum_{j=1}^{n} X_{2j} \leq 3$
* `SquareSum` $\{1..2\}$ $\{3..n\}$ `> 5 :` $\sum_{i=1}^{2} \sum_{j=3}^{n} X_{ij} \geq 5$
* `TriangSum` $\{1..3\}$ `= 5 :` $\sum_{i=1}^{3} \sum_{j=i}^{3} X_{ij} = 5$
* `StrictTriangSum` $\{1..3\}$ `= 5 :` $\sum_{i=1}^{3} \sum_{j=i+1}^{3} X_{ij} = 5$

3.4.4. *The "ProductConstrVect" and "ProductConstrBarVect" labels.* These two labels allow to make the product of linear constraints with the components x_i or $1 - x_i$ ($i \in \{1, ..., n\}$) of the vector x in order to build new "quadratic" constraints. The keywords are: "ProductConstrVect" to multiply some linear constraints by x_i, and ProductConstrBarVect to multiply by $1 - x_i$ ($i \in \{1, ..., n\}$) . These constraints have been introduced by Adams and Sherali [1].

Format of ProductConstrVect

```
ProductConstrVect[Right | Left]< filename >
```

Format of ProductConstrBarVect

```
ProductConstrBarVect[Right | Left]< filename >
```

Here, the file "filename" is read (it must contains only linear constraints !), and for each constraint in this file, the quadratic constraints are equal to the product of the constraint by each component x_i $i \in \{1, ..., n\}$ (or $1 - x_i$ for the *ProductConstrBarVect* function) of the vector x (and will be added to the semidefinite relaxation). The filename indicated between the parentheses must be declared before in the list of all the constraints, in order to know if the file is written in the "Sparse" or "Dense" format. But one may comment it by putting a "_" at the beginning of the line where it is declared (if one wants to use only the product constraints). The option 'Right' or 'left' is useful only for linear constraints that are of the "<>" type. If 'Right' is precised, only the right inequality will be quadratized. If 'Left' is precised, only the left inequality will be quadratized. The default value is 'Right'. Hence, to obtain "quadratized" constraints by multiplying all the inequalities of a "l<>" file "constraints.lin" by x_i $i \in \{1, ..., n\}$, one must write:

```
ProductConstrVectRight(constraints.lin)
ProductConstrVectLeft(constraints.lin)
```

Example 3.4. In this example, the SDP relaxation will contains the linear constraints *and* the constraints that are quadratized (by multiplying by x_i $i \in \{1, ..., n\}$) from the file "constraints_lin.dat":

```
File Sparse : constraints_lin.dat
ProductConstrVectRight(constraints_lin.dat)
```

Example 3.5. Here, only the constraints that are quadratized from the file "constraints_lin.dat" will be considered in the SDP relaxation (here we mupltiply them by both x_i and $1 - x_i$ $i \in \{1, ..., n\}$):

```
_File Sparse : constraints_lin.dat
ProductConstrVectRight(constraints_lin.dat)
ProductConstrBarVectRight(constraints_lin.dat)
```

3.4.5. *The Simple label.* In some cases, one may simply wants to copy the linear constraints of $(P\{0, 1\})$, i.e. without replacing it by on of the two sets of constraints described in Table 1. It is possible by using the **simple** label as follows.

```
_File Sparse : constraints_lin.dat
Simple(constraints_lin.dat)
```

4. A small example

All the files corresponding to this example can be found in the directory SDP_S/examples/PROBLEM. Assume that we want to obtain a lower bound by semidefinite programming (by using SDP_S)

for the following problem :

$$\begin{cases} \text{Maximize} & 2x_1x_2 + x_2x_3 + x_1 + x_2 + x_3 + 2 \\ \text{Subject to} & x_1x_2 + x_3 \leq 1 \\ & x_1 + x_2 + x_3 \leq 2 \\ & 2x_1 + 3x_2 + 5x_3 \leq 6 \\ & x \in \{0,1\}^n \end{cases}$$

Five files have to be written: the problem file, the "C" file, the "l" file, the linear constraints file, and the quadratic constraints file. Here, in addition we decide to use the function "ProdConstrVect" with all the linear constraints, and we add the constraints $X_{ij} \geq 0$ for all $i < j \in \{1, 2, 3\}$.

4.1. **The problem or "master" file: "problem.pb".** This file contains the general description of the problem.

```
Max CX + lx + 2
C : file Sparse : problem.C
l : file Sparse : problem.l
Dim : 3
Type : {0,1}
AddMoreConstr : 1
Constraints :
File Sparse : problem.lin
File Sparse : problem.quad
ProductConstrVect(problem.lin)
```

4.2. **The "C" file: "problem.C".** This file contains the values of all the quadratic part of the objective function:

```
3  //dimension
2  //number of quadratic terms
1 2 1.0
2 3 0.5
```

(since the matrix described is symmetric).

4.3. **The "l" file: "problem.l".** This file contains the values of all the linear part of the objective function:

```
3  //dimension
3  //number of linear terms
1 1.0
2 1.0
3 1.0
```

4.4. **The linear constraints file: "problem.lin".** This file contains the two linear constraints:

```
l<
3  //dimension
2  //number of the constraints
2  //upper bound
3  //number of the non-zero terms of the vector
1 1.0
2 1.0
3 1.0
6
3
1 2.0
2 3.0
3 5.0
```

4.5. **The quadratic constraints file : "problem.quad".** This file contains the quadratic constraint:

```
Q<
3  //dimension
1  //number of constraints
1  //upper bound
1  //linear part
3 1.0
1  //quadratic part
1 2 0.5
```

4.6. **The option file : "problem.opt".** This file is created by SDP_S if it is missing in the current directory. But here, we have chosen to write our own:

```
-si 1 -sh 1 -te 1e-3
```

"-te 1e-e3" is an option of the SB solver which specifies the relative precision for termination. The default value (equal to 1e-5) is too small for our small combinatorial problem.

4.7. **Output.** Running this small example, you should obtain this output (the file can be found in the directory SDP_S/examples/PROBLEM):

```
SDP_S version 1.0, Copyright (C) 2002 G.DELAPORTE, S.JOUTEAU, F.ROUPIN
SDP_S comes with ABSOLUTELY NO WARRANTY
Maximizing
Dim : 3
Filename C : problem.C
Filename l : problem.l
Reading Linear constraints in file "problem.lin".
2 linear constraints read.
Reading Quadratic constraints in file "problem.quad".
1 quadratic constraints read.
Adding 6 product constraints made from file problem.lin
Adding constraints : Xij >= 0 ; 1<=i<j<=n
SBmethod version 1.1.1, Copyright (C) 2000 Christoph Helmberg
SBmethod comes with ABSOLUTELY NO WARRANTY
elapsed time: 00:00:00.00 ---- Fri Sep 20 15:39:46 2002
Dimension = 5 Number of constraints = 16
00:00:00.00 Primal : 0 Bound : (7.7308752)
00:00:00.00 Primal : 6.9162863 Bound : (7.1941383)
00:00:00.00 Primal : 7.0532889 Bound : (7.1012386)
00:00:00.00 Primal : 6.9580565 Bound : (7.0363676)
00:00:00.01 Primal : 6.9093106 Bound : (6.9968169)
00:00:00.02 Primal : 6.8362214 Bound : (6.9593700)
00:00:00.02 Primal : 6.7742534 Bound : (6.8696101)
00:00:00.02 Primal : 6.6361816 Bound : (6.7666645)
00:00:00.02 Primal : 6.4939071 Bound : (6.5743581)
00:00:00.02 Primal : 6.2397256 Bound : (6.2868662)
00:00:00.02 Primal : 5.8958654 Bound : (6.1128962)
00:00:00.03 Primal : 5.9091241 Bound : (6.0623258)
00:00:00.04 Primal : 5.9129459 Bound : (6.0332024)
00:00:00.04 Primal : 5.9007371 Bound : (6.0176165)
00:00:00.05 Primal : 5.9322197 Bound : (6.0086182)
00:00:00.12 Primal : 5.9663304 Bound : (6.0039070)
00:00:00.25 Primal : 5.9839651 Bound : (6.0018570)
00:00:00.58 Primal : 5.9925850 Bound : (6.0009266)
00:00:00.69 Primal : 5.9939318 Bound : (6.0009266)
>>>elapsed time: 00:00:00.69 ---- Fri Sep 20 15:39:46 2002
-------- SDP bound : 6.000927
Vector x:
0.999920 0.999521 0.000156
```

We obtain here an integral solution $(1, 1, 0)$ and the optimal value 6. Look the other examples in SDP_S/examples. In particular, there are the source codes to generate the input files for

SDP_S for the Quadratic Assignment and the Memory-Constrained Allocation problems. Details are given in the corresponding "README" files in /examples/QAP and /examples/MCAP.

References

[1] W.P. Adams and H.D. Sherali, *A tight linearization and an algorithm for zero-one quadratic programming problems*. Management Science, 32(10), 1274-1290, 1986.

[2] C. Helmberg and F. Rendl, *A spectral bundle method for semidefinite programming*, ZIB Prepint SC 97-37, Konrad-Zuse-Zentrum fuer Informationtechnick Berlin, Takustrasee 7, D-14195 Berlin, Germany, 1997.

[3] C. Hemberg, *A C++ implementation of the Spectral Bundle Method*, Manual version 1.1. http://www-user.tu-chemnitz.de/ helmberg/SBmethod/.

[4] F. Roupin. *From Linear to Semidefinite Programming : an Algorithm to obtain Semidefinite relaxations for bivalent Quadratic problems*. Journal Of Combinatorial Optimization 8(4):469-493, 2004.

5. Gnu General Public Licence

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as

distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted

by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items–whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.