The DIMACS library of semidefinite-quadratic-linear programs

Gábor Pataki and Stefan Schmieta

Computational Optimization Research Center
Columbia University
PRELIMINARY DRAFT

July 29, 2002

1 Introduction

The primal semidefinite-quadratic-linear program is

$$\begin{array}{llllll} \min & \sum_{j=1}^{n_S} C_j^S \bullet X_j^S & + \quad \sum_{j=1}^{n_Q} C_j^Q * X_j^Q & + \quad C^N * X^N & & \\ \text{s.t.} & \sum_{j=1}^{n_S} A_{ij}^S \bullet X_j^S & + \quad \sum_{j=1}^{n_Q} A_{ij}^Q * X_j^Q & + \quad A_i^N * X^N & = b_i \quad i=1,\ldots,m & \quad (P) \\ & X_j^S \succeq 0 & X_j^Q \geq_Q 0 & X^N \geq 0 & & \\ & (j=1,\ldots,n_S) & (j=1,\ldots,n_Q) & & & \end{array}$$

The description of the variables, conic inequalities, inner products and cost and constraint matrices in (P) is as follows.

The variables in (P):

- $X_1^S, \ldots, X_{n_S}^S$ are symmetric matrix variables.
- $X_1^Q, \ldots, X_{n_Q}^Q$ are vector variables.
- X^N is a vector variable.

The conic inequalities in (P):

- $X_j^S \succeq 0$ means that the matrix X_j^S is positive semidefinite.
- $X_j^Q \geq_Q 0$ means that the vector X_j^Q is in a quadratic cone (also known as the second-order cone, Lorentz cone or ice cream cone) of appropriate size.
- $X^N \geq 0$ means that each component of X^N is nonnegative.

The inner products in (P):

- If A and B are symmetric matrices, then $A \bullet B = \text{trace}\ (AB)$.
- If A and B are vectors, then $A * B = A^T B$.

The cost and constraint matrices in (P):

- C_j^S, and A_{ij}^S $(i = 1, \ldots, m;\ j = 1, \ldots, n_S)$ are symmetric matrices.
- C_j^Q, and A_{ij}^Q $(i = 1, \ldots, m;\ j = 1, \ldots, n_S)$ are ordinary vectors.
- C^N and A_i^N $(i = 1, \ldots, m)$ are ordinary vectors.
- b is an m-vector.

Thus the feasible set is a product of semidefinite, quadratic and nonnegative orthant cones, intersected with an affine subspace. It is possible that one or more of the three parts of the problem is absent, i.e., any of n_S, n_Q, or the length of X^N may be zero. We assembled a library of test problems of semidefinite-quadratic-linear programs, mainly for the purposes of the 7^{th} DIMACS Implementation Challenge. We attempted to include instances that

(1) arise from the widest possible range of sources, and applications.

(2) are as realistic as possible.

(3) represent all levels of difficulty.

(4) have their origin and the formulation used clearly documented.

The library is divided into problem sets; some of the problem sets are grouped into problem families. The problem families, and sets are listed below.

- The cut problems family:
  - The *torus* set: *toruspm3-8-50*, *toruspm3-15-50*, *torusg3-8*, *torusg3-15*.
  - The *fap* set: *fap09*, *fap25*, *fap36*.
  - The *bisection* set: *bm1*, *biomedP*, *industry2*.
- The *hamming* set: *hamming_9_8*, *hamming_10_2*, *hamming_11_2*, *hamming_7_5_6*, *hamming_8_3_4*, *hamming_9_5_6*.
- The sum of norms family:
  - The *nql* set: *nql30*, *nql60*, *nql180*.
  - The *qssp* set: *qssp30*, *qssp60*, *qssp180*.
- The *antenna* set: *nb*, *nb_L1*, *nb_L2*, *nb_L2_bessel*.
- The *sched* set: *sched_50_50*, *sched_100_50*, *sched_100_100*, *sched_200_100*.

- The *copos* set: *copo14*, *copo23*, *copo68*.
- The *filter* set: *filter48*, *filtinf1*, *minphase*.
- The *truss* set: *truss5*, *truss8*.
- The *hinf* set: *hinf12*, *hinf13*.

A few remarks on points (1) through (4). As to (1), we tried not to overrrepresent any problem type; each problem set contains at most 4 (usually less) instances of varying difficulty. As to (2), we preferred problems arising from true applications as opposed to randomly generated ones. To create some instances, randomness *was* used - but only embedded within a careful modeling process. Hence the optimal solutions of these instances yield useful information about real world problems. A case in point is the *torus* problem set, described in subsection 2.1.
As to (3), there are instances that are routinely solvable; difficult, and at, or beyond the levels of solvability by current methods. The emphasis is on the second two levels; "easy" problems were still included, when they represent an interesting application. We do not expect any current general purpose method to be able to solve all the instances.
As to (4), in computational optimization advances frequently arise from better formulation techniques as well as from improved algorithms, and computing machinery. The art of "well formulating" a problem is not nearly as advanced in semidefinite and conic quadratic programming, as in linear, or mixed integer programming. Therefore, we documented the formulation technique used in each case, and pointed out possible alternatives.
The problems are given in compressed matlab files, using the format of *Sedumi* [Stu99], a Matlab based toolbox for solving problems of the form (P). The reason is, that this format is probably the easiest to convert to any other, assuming that the user has Matlab at his/her disposal.

2 The cut problems family

For each instance in this family a weighted graph $G = (V, E, w)$ is given. We call $\mathcal{V} = (V_1, \ldots, V_k)$ a k-cut, if

$$\bigcup_{i=1}^{k} V_i = V \quad \text{and} \quad V_i \cap V_j = \emptyset \;\; \forall i \neq j.$$

For $\mathcal{V}$, a k-cut of V, we write

$$\text{cut}\,(\mathcal{V}), \quad [\text{uncut}\,(\mathcal{V})]$$

for the set of edges with their endpoints in different [identical] subsets of $\mathcal{V}$, respectively. The goal in each set of instances is to find a k-cut for a given k, that maximizes/minimizes the total weight of the cut, or uncut edges. The peculiarity of each set - *torus*, *fap* and *bisection* - comes from the special structure of the graph, of the required k-cut, or both. Let

$$W \;=\; (w_{ij})_{i,j=1}^{n}, \qquad L(G,w) \;=\; \text{Diag}\,(We) - W$$

The basic formulation to find the k-cut that maximizes the weight of edges in cut $(\mathcal{V})$ is

$$\begin{array}{ll} \max & \frac{k-1}{2k} L(G,w) \bullet X \\ st. & X \succeq 0 \\ & \text{diag } (X) = e \\ & X_{ij} \geq -1/(k-1) \;\; \forall (i,j) \in E \\ & \text{rank } X = k \end{array} \tag{1}$$

Its correctness can be seen as follows. For $j_1 \geq 1, \ldots, j_k \geq 1$, with $j_1 + \ldots j_k = n$, define

$$M(j_1, \ldots, j_k) = \begin{pmatrix} J_1 & \frac{-1}{k-1} & \frac{-1}{k-1} \\ \frac{-1}{k-1} & \ddots & \frac{-1}{k-1} \\ \frac{-1}{k-1} & \frac{-1}{k-1} & J_k \end{pmatrix}$$

with $J_1, \ldots, J_k$ being matrices of all ones, with order $j_1, \ldots, j_k$, respectively. The feasible set of (1) (modulo simultaneous row and column permutations) is then

$$\{\, M(j_1, \ldots, j_k) \in \mathcal{S}^n \mid j_1 \geq 1, \ldots, j_k \geq 1, \, j_1 + \ldots j_k = n \,\}.$$

For $M(j_1, \ldots, j_k)$ define $\mathcal{V}$ to be the k-cut whose subsets correspond to the $J_1, \ldots, J_k$ submatrices. Then a simple calculation shows dropping the last (that is, the only non-convex) constraint from (1) yields the semidefinite relaxation for max k-cut, proposed by Frieze and Jerrum [FriJe95].

2.1 The *torus* set: max cut problems from the Ising model of spin glasses

A *spin glass* is an alloy of n magnetic particles diluted in a non-magnetic material. Each particle will assume a *spin*, so as to minimize the total energy of the system. In the *Ising model* of spin glasses, the spin is represented by ± 1, and the total energy depends on

$$\sum_{i \neq j} w_{ij} s_i s_j$$

with w_{ij} being a given fixed weight for particles i and j, and s_i the the spin assumed by particle i. The Ising model is an approximation of the true behaviour of a spin glass; its advantage is that – given the w_{ij}'s – the optimal assignment of the s_i's can be computed by solving a maximum cut (i.e. 2-cut) problem on the graph $G = (V, E, w)$ with V representing the particles, and E the pairs with nonzero w_{ij} (see e.g. [DeSietal955]).
Thus, Ising spin glasses can be studied via computational experiments (instead of time consuming physical ones). In a typical experiment, the particles are placed on some "regular" grid, and the w_{ij}'s randomly generated according to some distribution for the node pairs closest to each other in the grid. Then the max-cut problem is solved to find the optimal assignment of the spins. To infer useful information on the expected behaviour of a real spin glass, a large number of such runs is needed.

NAME	$\|V\|$	$\|E\|$	$w(E)$	$w_+(E)$	Best cut	SDP value
toruspm3-8-50	512	1,536	0	768	458	527.808663
toruspm3-15-50	3,375	10,125	1	5063	3016	3474.8
torusg3-8	512	1,536	-26.05128	594.24679	391.11654	457.358179
torusg3-15	3,375	10,125	-53.52809	4032.29292	2602.02525	3134.6

Table 1: Statistics of the *torus* problems

The *toruspm3-x-50* and *torusg3-x* instances are SDP relaxations of maximum cut problems arising from such computations. The grid is a 3 dimensional torus-grid, with dimension x, thus the number of nodes in the graph is x^3. The *toruspm3-x-50* instances were generated with edge-weights being ± 1 with probability 50%. The *torusg3-x* instances were generated with edge-weights according to a Gaussian distribution. The library contains the SDP relaxations of these problems, namely

$$\begin{array}{rl} \max & \frac{1}{4}L(G,w) \bullet X \\ st. & \text{diag}\,(X) = e \end{array}$$

the specialization of (1) in section 2, after the $X_{ij} \geq -1/(k-1)$ constraints (which are redundant when $k = 1$) have been dropped. Some statistics of the graphs are in table 1; $w(E)$ denotes the sum of all edge-weights, $w_+(E)$ the sum of all positive edge-weights, and the best upper bound (when available) was computed by solving the SDP relaxation. The best cuts were obtained by Andrea Franceschini, and Carlo Mannino (personal communication) using simulated annealing.

2.2 The *fap* set: k-cut problems with large k from frequency assignment

The *fap-xx* instances arise from frequency assignment for E-Plus Mobilfunk GmbH, a German GSM operator. We are seeeking a k-cut $\mathcal{V}$ for some fairly large k (see Table 2), in which certain node pairs end up in different sets, and

$$w(\text{uncut}\,(\mathcal{V}))$$

is minimized. Let E_U denote the collection of the edges which must be in uncut$\,(\mathcal{V})$. Then the same argument as used to derive (1) shows that

$$\begin{array}{ll} \min & \left[\frac{1}{2}\text{Diag}\,(We) - \frac{k-1}{2k}L(G,w)\right] \bullet X \\ st. & X \succeq 0 \\ & \text{diag}\,(X) = e \\ & X_{ij} \geq -1/(k-1) \;\; \forall (i,j) \\ & X_{ij} = -1/(k-1) \;\; \forall (i,j) \in E_U \\ & \text{rank}\; X = k \end{array} \tag{2}$$

is a correct formulation of our problem, and dropping the rank constraint yields a valid lower bound. Of course, minimizing $w(\text{uncut}\,(\mathcal{V}))$ is equivalent to maximizing

NAME	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert E_U\rvert$	k	$w(E)$	Best k-uncut	SDP value
fap-sup-09	174	15,051	1,026	50	2862.861778500	12.132	10.7
fap-sup-25	2,118	320,806	9,762	50	27213.053924500	32.4010	12.5 (lb, not opt)
fap-sup-36	4,110	1,150,357	38,064	75	96383.723082807	133.0630	63.7 (lb, not opt)

Table 2: Statistics of the *fap* problems

$w(\text{cut}\,(\mathcal{V}))$; when seeking approximate solutions, the difficulty of the 2 problems is quite different though. For max k-cut, it is easy to find solutions with value provably within 2% of the optimum, even without using SDP (see e.g. Frieze and Jerrum [FriJe95]). For min k-uncut, no approximation guarantee is known.

The *fap* problems are extremely difficult, hence in most cases one can be satisfied even solving a somewhat weaker relaxation. The *fap-sup-...* instances arise from the *fap* instances by keeping only those $X_{ij} \geq -1/(k-1)$ inequalities that correspond to an edge with nonzero weight. There is no *fap-sup-09* instance, since in the corresponding graph every edge has a positive weight.

In fact, to our knowledge, no one has ever attempted to solve the *fap25* and *fap36* instances. The statistics are in Table 2. The letter E denotes the set of edges with nonzero weight, and we consider $E_U \subseteq E$ (by say, setting the weight of an edge in E_U to some large constant). The value of the best partition that minimizes the value of the uncut edges was found by simulated annealing, and the value of the SDP-relaxation by solving the SDP relaxation to approximate optimality. These values were taken from ???.

For the *fap* instances $m = n(n+1)/2$, as there is a constraint (either equality, or inequality) for every entry of the matrix, and the number of nonnegative slacks is $n(n+1)/2 - n - |E_U|$. For the *fap-sup* instances $m = |E| + n$, and the number of nonnegative slacks is $|E| - |E_U|$.

2.3 The *bisection* set: graph bisection problems from circuit design

These instances are graph bisection problems arising from VLSI circuit design. Now the goal is to partition V into 2 sets of equal size to *minimize* the total weight of the cut edges. The formulation for these problems is

$$\begin{array}{rl} \min & \frac{1}{4}L(G,w) \bullet X \\ st. & X_{ii} = 1 \qquad (i = 1, \ldots, n) \\ & J \bullet X = 1 \\ & \text{rank } X = 1 \end{array}$$

The challenge here is twofold: solving the relaxations to find valid lower bounds, and finding good feasible solutions by "rounding" the SDP solution. Some statistics of the graphs are given in Table 3.

Remark 2.1. A few words about the origin of these instances. For each graph $G = (V, E, w)$, the original problem is to bisect a corresponding weighted *hypergraph* $G' =$

NAME	$\|V\|$	$\|E\|$	$w(E)$	SDP value
bm1	882	4,711	1092.593480000	23.43982
biomed	6,514	629,839	6580.853243982	33.6
industry2	12,637	798,219	15460.018520953	65.6

Table 3: Statistics of the bisection problems

(V, E', w'), where E' is a collection of hyperedges (i.e. of subsets of V with cardinality more than 2). If (V_1, V_2) is a bisection of V, then its weight in G' is

$$\sum \{ w'(e) \mid e \in E',\ |e \cap V_1| < |e| \}$$

For each hyperedge $e \in E'$, and for all $u, v \in e$, there will be a corresponding edge $(u, v) \in E$, with $w'(u, v)$ appropriately defined. See [ChoYe99] for more details, and also for more references on partitioning techniques in VLSI design. Their computational experiments show that the SDP solution can be used to generate a high quality bisection in the corresponding *hypergraph* (although for the *minimum* weight bisection there is no performance guarantee of an SDP based approximation algorithm). We also think, that these SDP's are good for testing algorithms that round an SDP solution to find a near minimal bisection in the corresponding *graph*, in the spirit of the algorithms in [FriJe95].

3 The *hamming* problem set: computing the ϑ function for Hamming graphs

The ϑ-function is one of the earliest, and most interesting applications of semidefinite programming. Given a graph $G = (V, E)$, we define

$$\begin{aligned} \vartheta(G) \quad = \quad \max \quad & ee^T \bullet X \\ \text{st.} \quad & X \succeq 0 \\ & X_{ij} = 0 \text{ for all } (i, j) \in E \end{aligned}$$

If G is a *Hamming-graph*, to be defined below, then there is a closed form expression for $\vartheta(G)$; for a detailed description, see [Schil99]. Hence, these graphs are well suited to test how well solvers can handle large-scale SDP's with sparse constraints.

The Hamming graph $H_{n,\{d_1,\ldots,d_k\}}$ is defined as

$$\begin{aligned} V_{n,\{d_1,\ldots,d_k\}} &= \{0,1\}^n \\ E_{n,\{d_1,\ldots,d_k\}} &= \{(u, v) \in V \mid |u - v| = d_1, \text{ or} \ldots, |u - v| = d_k \}. \end{aligned}$$

The library contains 6 instances, with parameters listed in Table 4.

4 The sum of norms family

The origin of these problems is described in the papers [ChAn98] and [AnChOv98]. They model the following situation: on a plastic plate, a load is placed. As this load is

NAME	$\|V\|$	$\|E\|$	$\vartheta(G)$
hamming_9_8	512	2,305	224
hamming_10_2	1,024	23,041	102.4
hamming_11_2	2,048	56,321	170 2/3
hamming_7_5_6	128	1,793	42 2/3
hamming_8_3_4	256	16,129	25.6
hamming_9_5_6	512	53,761	85 1/3

Table 4: Statistics of the *hamming* problems

increased, the plate will collapse; at this point in time, one would like to take a snapshot of the collapse state. Instead of doing a physical experiment, the collapse state can be computed by solving a quadratically constrained quadratic program.
Both the *nql* and the *qssp* problem sets arise this way; their difference lies in the underlying model.

5 The *antenna* problem set: SOCP's for sidelobe level minimization in antenna arrays

Given a direction u, and a set of antennae located on the plane, these problems seek to choose weights for their outputs such that the result mimics an antenna that primarily listens towards u. Such problems arise for example in the navy, where choosing the listening direction aims to pick up an expected transmission, while suppressing all transmissions coming from elsewhere as noise.

A more detailed description of the problem is as follows: An isotropic, omnidirectional antenna at location x produces an output

$$\gamma e^{2\pi i(\ell^T \mathbf{x} + ||\ell|| ct)},$$

where

- t is the time at which a single-frequency plane wave, with velocity c, and wavenumber vector $-\ell$, reaches it.
- γ is its sensitivity.

An antenna array consists of many such antennae distributed in the plane. Denote by $\mathcal{X}$ the finite set of antennae locations. The output of the array, or beamformer output is then the complex linear combination

$$\sum_{x\in\mathcal{X}} w_x \gamma e^{2\pi i(\ell^T x + ||\ell|| ct)} = \gamma e^{2\pi i(||\ell|| ct)} \sum_{x\in\mathcal{X}} w_x e^{2\pi i \ell^T x} \tag{3}$$

The second factor in (3) is called the *beamformer gain*, which is the ratio of the beamformer output to the output of a hypothetical isotropic antenna at location $x = 0$.

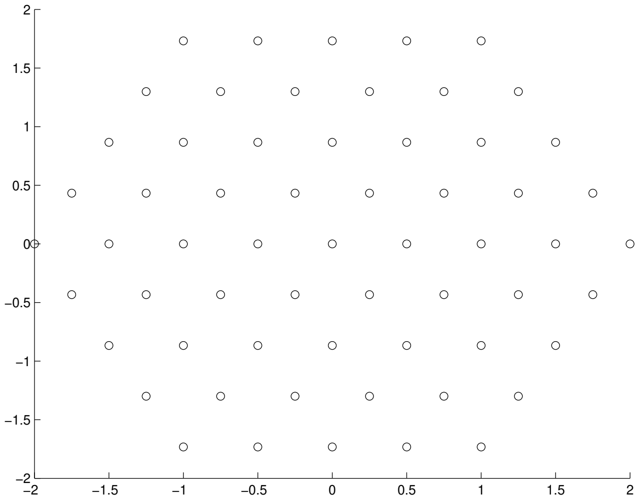

Figure 1: Antenna locations

Now suppose that the array can receive signals only from above the plane, i.e. the possible look directions can be chosen from a hemisphere. We, however, are only interested in listening to signals coming from direction u and not more than θ degrees from it. This region is called the *main beam* and all other directions the *sidelobes*. Sidelobe level minimization seeks appropriate weights w_x to minimize some measure of the sidelobe beamformer gain, while keeping the main beam's gain fixed.

This problem becomes a QCQP if we discretize the set of look directions. The problems in the library assume that the signal frequency is c/λ and that the array contains 61 elements located in the horizontal plane within an origin-centered hexagonally shaped region at the $\lambda/2$ spaced points of triangular lattice (see figure 1). As the coordinates are normalized with respect to λ, the model is independent of the actual values of c and λ. The hemisphere is discretized into 889 look directions. Their projections into the plane are shown in Figure 2.

All four problems require the sidelobe gains for each discretization point to be below -15dB. The main beam's gain is fixed by requiring the gain for direction u to be 0dB. They differ in their overall measure of sidelobe gain.

nb: Minimize the largest sidelobe gain. In this problem the -15dB requirement for all sidelobe gains is omitted since it is implied by the objective function. The look direction is $u = (0.33, 0.19, 0.92)$ and $\theta = 20°$.

nb_L1: Minimize the L_1-norm of the vector of sidelobe gains. The look direction is $\mathbf{u} = (0.33, 0.19, 0.92)$ and $\theta = 20°$.

nb_L2: Minimize the L_2-norm of the vector of sidelobe gains. The look direction is $u = (0.866, 0, 0.5)$ and $\theta = 20°$.

nb_L2_bessel: Minimize the L_2-norm of the vector of sidelobe gains. In this model, the incoming signal is assumed to be a single plane wave, but with random amplitude and phase. Since the signal is random, calculating the gains involves the

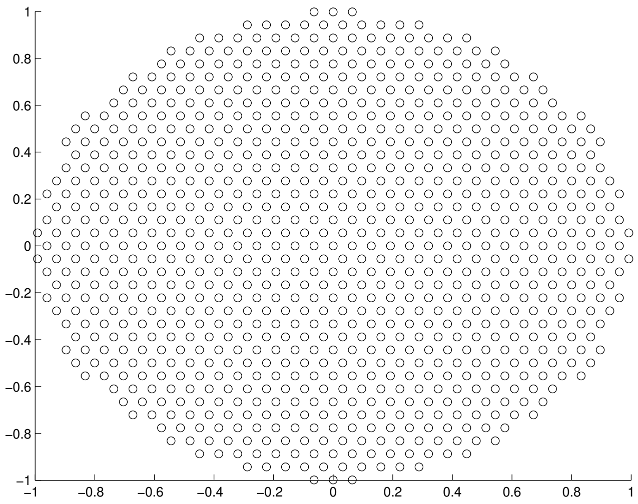

Figure 2: Projections of look directions into the plane

Name	m	SOCP	LP	optimal obj.	Time SeDuMi	Time LOQO
nb	123	$[793; 793 \times 3]$	4	-0.05070309	52s	1m24s
nb_L1	915	$[793; 793 \times 3]$	797	-13.012337	9m14s	diverges
nb_L2	123	$[839; 1 \times 1677, 838 \times 3]$	4	-1.62897198	1m31s	2m24s
nb_L2_bessel	123	$[839; 1 \times 123, 838 \times 3]$	4	-0.102569511	38s	1m06s

Table 5: Summary of problem statistics

autocorrelation of the random field. The autocorrelation can be computed with the help of the Bessel function, hence the name (for details see [1]). The look direction is $u = (0.866, 0, 0.5)$ and $\theta = 20°$.

6 The *sched* problem set: SOCP formulations of machine scheduling problems

These instances arise from scheduling jobs on unrelated, parallel machines. We are given m jobs, n machines, p_{ij}, the processing time of job j on machine i, and a weight w_j corresponding to job j. Each job must be processed on exactly one machine, in order to minimized the sum of the weighted completion times.

Finding the optimal schedule breaks down into 2 parts:

(1) Assigning each job to a machine.

(2) For each machine, sequencing the jobs assigned to it.

After (1) has been done, (2) is straightforward: we process job j before job k, iff

$$\frac{w_j}{p_{ij}} > \frac{w_k}{p_{ik}} \quad \text{or} \quad \frac{w_j}{p_{ij}} = \frac{w_k}{p_{ik}} \text{ and } j < k. \tag{4}$$

We will write $j \prec_i k$, if (4) holds.

The quadratic IP formulation uses 0–1 variables x_{ij}, to be set to 1, iff job j is assigned to machine i. It reads as

$$\begin{array}{rlr} \min_{x,C} & \sum_{j=1}^n w_j C_j & \\ st. & \sum_{i=1}^m x_{ij} = 1 & \text{for all } j \\ & C_j = \sum_{i=1}^m x_{ij} \left(p_{ij} + \sum_{k \prec_i j} x_{ik} p_{ik} \right) & \text{for all } j \\ & x_{ij} \in \{0,1\} & \text{for all } i,j \end{array} \tag{5}$$

Defining $x = (x_{11}, \ldots, x_{1n}, \ldots, x_{m1}, \ldots, x_{mn})$, and substituting for the C_j's, (5) becomes

$$\begin{array}{rlr} \min_x & c^T x + \frac{1}{2} x^T D x & \\ st. & \sum_{i=1}^m x_{ij} = 1 & \text{for all } j \\ & x_{ij} \in \{0,1\} & \text{for all } i,j \end{array} \tag{6}$$

for an appropriate c and D. In particular, D is block-diagonal, with zero diagonal, blocks corresponding to machines, and

$$d_{(ij)(ik)} = \begin{cases} w_j p_{ik}, \text{if } k \prec_i j. \\ w_k p_{ij}, \text{if } j \prec_i k. \end{cases} \tag{7}$$

An equivalent formulation of (6) is

$$\begin{array}{rlr} \min_{z,x} & z & \\ st. & z \geq \frac{1}{2} c^T x + \frac{1}{2} x^T (D + \text{Diag }(c)) x & \\ & z \geq c^T x & \\ & \sum_{i=1}^m x_{ij} = 1 & \text{for all } j \\ & x_{ij} \in \{0,1\} & \text{for all } i,j \end{array} \tag{8}$$

since for $x_{ij} \in \{0,1\}$,

$$\frac{1}{2} c^T x + \frac{1}{2} x^T (D + \text{Diag }(c)) x = c^T x + \frac{1}{2} x^T D x \text{ and } c^T x + \frac{1}{2} x^T D x \geq c^T x$$

However, as shown by Skutella, in (8) all constraints are convex, since $(D + \text{Diag }(c)$ is positive semidefinite. The constraint $z \geq c^T x$ is redundant, when the x_{ij}s are 0–1, but it strengthens the relaxation.

One can formulate (8) in several ways. In all of them, one uses the factorization $\frac{1}{2}(D + \text{Diag }(c)) = GG^T$.

The original formulation first rewrites (8) as

$$\begin{array}{rlr} \min_{z,x,y} & z & \\ st. & y = G^T x & \\ & y_0^2 \geq \|y\|_2^2 & \\ & z \geq \frac{1}{2} c^T x + y_0^2 & \\ & z \geq c^T x & \\ & \sum_{i=1}^m x_{ij} = 1 & \forall\, j \\ & x_{ij} \in \{0,1\} & \forall\, i,j \end{array} \tag{9}$$

which results in the 0–1 SOCP formulation

$$
\begin{array}{rll}
\min_{z,x,y,s_0,s_1,s_2,s_3} & z & \\
st. & (y_0, y) \in SO & \\
& (s_0, s_1, s_2) \in SO & \\
& z, x, s_3 \geq 0 & \\
& y - \sqrt{2} G^T x = 0 & \\
& s_0 + s_1 = 1 & \\
& c^T x - 2z + s_0 - s_1 = 0 & \\
& s_2 - y_0 = 0 & \\
& c^T x - z + s_3 = 0 & \\
& \sum_{i=1}^m x_{ij} = 1 & \text{for all } j \\
& x_{ij} \in \{0,1\} & \text{for all } i,j
\end{array}
\qquad (10)
$$

These formulations are the `.orig` instances in the library.

Another formulation is

$$
\begin{array}{rlll}
\min_{z,x,s_1,s_2,u,s_3} & z & & \\
st. & (s_1, s_2, u) \in SO & & \\
& z, x, s_3 \geq 0 & & \\
& z - \frac{1}{2\gamma} c^T x - s_1 & = -1 & \\
& -z + \frac{1}{2\gamma} c^T x - s_2 & = -1 & \\
& 2G^T x - \sqrt{\gamma} u & = 0 & \\
& \gamma z - c^T x - s_3 & = 0 & \\
& \sum_{i=1}^m x_{ij} = 1 & \text{for all } j & \\
& x_{ij} \in \{0,1\} & \text{for all } i,j &
\end{array}
\qquad (11)
$$

The `.scaled` instances in the library use this formulation with the scaling factor set to $\gamma = \|c\|/n$.

7 The *copos* problem set: checking copositivity of multivariate polynomials

The *copos* set contains mixed semidefinite-linear programs that arise in checking the copositivity of homogeneous, multivariate polynomials. Copositive matrices and polynomials have numerous already recognized and many potential applications in pure mathematics and optimization. Also, the the resulting SDP's are ideally suited to test how well a solver can handle sparse constraints, and block structured semidefinite matrix variables. Our review is based on the detailed description in [Par00].

Semidefinite programming can be used to check the *nonegativity* of such a polynomial as follows: given

$$F(x) = F(x_1, \ldots, x_n) \;=\; \sum_\alpha \bigl(c_\alpha \prod_{i=1}^n x_i^{\alpha_i}\bigr)$$

the degree of a monomial $c_\alpha \prod_{i=1}^n x_i^{\alpha_i}$ is defined as $\sum_{i=1}^n \alpha_i$, and the degree of $F(x)$ as the maximum degree of its monomials. If the degree of $F(x)$ is odd, then it is negative for some x. If its degree is even, then

$$F(x) \equiv v(x)^T Q v(x) \tag{12}$$

for some matrix Q. Here $v(x)$ is a vector containing all monomials of degree $\deg F/2$. Due to identities of the type

$$\begin{aligned} (x_i x_j)^2 &= x_i^2 x_j^2, \\ (x_i x_j)(x_i x_k) &= x_i^2 x_j x_k, \end{aligned}$$

the Q in (12) is not unique. If there is a positive semidefinite Q, then clearly $F(x) \geq 0$ for all x. For example,

$$\begin{aligned} F(x_1, x_2) &= 2x_1^4 + 2x_1^3 x_2 - x_1^2 x_2^2 + 5x_2^4 \\ &= \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}^T \begin{bmatrix} 2 & 0 & 1 \\ 0 & 5 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix} \\ &= \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}^T \begin{bmatrix} 2 & -\lambda & 1 \\ -\lambda & 5 & 0 \\ 1 & 0 & -1+2\lambda \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}. \end{aligned}$$

Here λ can be arbitrarily chosen to satisfy the equalities. For $\lambda = 3$, however, the second matrix is psd, proving the nonnegativity of $F(x)$.

More generally, we may check, whether for some $r \geq 0$

$$F_r(x) := \left(\sum_{i=1}^n x_i^2 \right)^r F(x)$$

has a decomposition of the type (12). If yes, then $F_r(x)$ is nonnegative for all x, hence so is $F(x)$. If $F_r(x)$ has a decomposition of the type (12), so does $F_{r+1}(x)$, but not vice versa; moreover, if $F(x)$ is nonnegative for all x, then with a large enough r we can always prove this, see [Par00].

Checking whether a polynomial $G(x) = G(x_1, \ldots, x_n)$ is *copositive*, that is

$$G(x) \geq 0 \text{ for all } x \geq 0 \tag{13}$$

can be done by checking whether

$$G(x_1^2, \ldots, x_n^2) \geq 0 \text{ for all } x \tag{14}$$

The three instances in the library arise from quadratic cyclic polynomials (that is, $n + r = r$ for $r \geq 1$)

$$G^n(x) = \left(\sum_{i=1}^n x_i \right)^2 - 2 \sum_{i=1}^n x_i \sum_{j=0}^m x_{i+3j+1}$$

with $n = 3m + 2$, and $m = 4, 7, 22$ ($n = 14, 23, 68$). The SDP's seek to find a decomposition as in (12) for

$$\left(\sum_{i=1}^n x_i^2 \right) G^n(x_1^2, \ldots, x_n^2)$$

As the degree of G^n is 2, we must list monomials in $v(x)$, hence this is the order of the matrix in the corresponding SDP. Simplifications are possible though, hence, if M_n is the matrix that defines the quadratic polynomial G^n, then G^n is copositive, iff the semidefinite system

$$\begin{array}{rcll} G^n - \Lambda^i & \succeq & 0 & i = 1, \ldots, n \\ \Lambda^i_{ii} & = & 0 & i = 1, \ldots, n \\ \Lambda^i_{jj} + 2\Lambda^j_{ij} & = & 0 & i, j = 1, \ldots, n; \quad i \neq j \\ \Lambda^i_{jk} + \Lambda^j_{ki} + \Lambda^k_{ij} & = & 0 & i, j, k = 1, \ldots, n; \quad i \neq j, i \neq k, j \neq k \end{array} \tag{15}$$

The instances implement an optimization variant of (15) by replacing $G^n - \Lambda^i \succeq 0$ with $G^n - \Lambda^i + zI \succeq 0$ and minimizing z. Hence, a nonpositive optimal value of z proves the copositivity of G^n.

8 The *filter* problem set: PAM filter design

This set consists of PAM (pulse amplitude modulation) filter design problems from the field of signal processing. The application area is digital communication. One of the fundamental operations in that area is the representation of a message symbol by an analog waveform that is then transmitted. The choice of such waveforms is important for the performance of the resulting communication scheme, and usually involves a compromise between many technical and design constraints. With the help of digital signal processors many such waveforms can be easily implemented and are therefore of interest to practitioners. In this setting, the design of the waveform can be transformed to the design of a multi-rate discrete-time finite impulse response filter. The problems in this set are part of such a design process.

For further details see the papers [Dav00, DLW99] and [DLS00].

8.1 `filter48`

This problem is example 1 from [Dav00]. Here the goal is to design a filter, that minimizes the interference over a given class of distorting channels. In particular, the objective is to minimize the sensitivity coefficient to the worst-case channel subject to a bound on the sensitivity in an ideal channel. In addition, the problem has normalization and "bandwidth" constraints. The original formulation in [Dav00] is

$$\begin{aligned} &\min_{\mu,\gamma,\zeta,r_g} \gamma \\ &\text{subject to} \\ &\qquad -\mu_i \leq r_g[Ni] \leq \mu_i, \qquad i = 1, \ldots, k \\ &\qquad \sum_{i=1}^{k} \mu_i \leq \epsilon \\ &\qquad \|r_g\|^2 \leq \gamma, \quad r_g[0] = 1 \\ &\qquad \zeta 10^{p_l(\omega)/10} \leq R_g(e^{j\omega}) \leq \zeta 10^{p_u(\omega)/10}, \qquad \text{for all } \omega \in [0, \pi] \\ &\qquad \zeta > 0, \mu \geq 0, \gamma \geq 0 \end{aligned} \tag{16}$$

where L is the length of the filter, N and ϵ are given parameters, and $k = \lceil (L-1)/N \rceil$. The design variables r_g are the autocorrelation coefficients of the desired filter. To ensure that the r_g are indeed the autocorrelation of a filter, a semidefinite constraint, stemming from the Positive Real Lemma, is added The functions $p_l(\omega)$ and $p_u(\omega)$ describe a spectral mask on the power spectrum R_g of the filter.

The formulation used in the donated `filter48` problem applies theorem 4.1 from [DLS00] to replace the original semidefinite constraint. Thus we obtain the model:

$$
\begin{aligned}
&\min_{\mu,\gamma,\zeta,r_g,X} \gamma \\
&\text{subject to} \\
&\qquad -\mu_i \le r_g[Ni] \le \mu_i, \qquad i = 1,\dots,k \\
&\qquad \sum_{i=1}^{k} \mu_i \le \epsilon \\
&\qquad \|r_g\|^2 \le \gamma, \quad r_g[0] = 1 \\
&\qquad \zeta 10^{p_l(\omega)/10} \le R_g(e^{j\omega}) \le \zeta 10^{p_u(\omega)/10}, \qquad \text{for all } \omega \in [0,\pi] \\
&\qquad X \bullet I = 1, \quad \frac{1}{2} X \bullet O_i = r_g[i], \qquad i = 1,\dots,L-1 \\
&\qquad \zeta > 0, \mu \ge 0, \gamma \ge 0, \quad X \succeq 0
\end{aligned}
\tag{17}
$$

Here I is the identity matrix, and O_i is the symmetric matrix with all ones in the i-th off-diagonals.

Note that the spectral mask constraint is a semi-infinite constraint. When discretized, it becomes a set of linear inequalities in the r_g variables. Finally, to convert the problem into our standard form, we note that the strict inequality $\zeta > 0$ can be weakened to $\zeta \ge 0$ without adverse effect. Introducing appropriate slacks, the formulation then becomes:

$$
\begin{aligned}
&\min_{\mu,\gamma,\gamma_u,\zeta,r_g,X} \gamma \\
&\text{subject to} \\
&\qquad r_g[Ni] + \mu_i - s_1 = 0, \qquad i = 1,\dots,k \\
&\qquad r_g[Ni] - \mu_i + s_2 = 0, \qquad i = 1,\dots,k \\
&\qquad \sum_{i=1}^{k} \mu_i + s_3 = \epsilon \quad r_g[0] = 1, \quad \gamma_u = \frac{1}{2} \\
&\qquad \zeta p_l - A_l r_g + s_4 = 0, \quad -\zeta p_u + A_l r_g + s_5 = 0 \\
&\qquad X \bullet I = 1, \quad \frac{1}{2} X \bullet O_i = r_g[i], \qquad i = 1,\dots,L-1 \\
&\qquad \zeta, \mu, \gamma, s_1, s_2, s_3, s_4, s_5 \ge 0, \quad X \succeq 0 \\
&\qquad \begin{pmatrix} \gamma \\ \gamma_u \\ r_g \end{pmatrix} \in R-\text{cone}
\end{aligned}
\tag{18}
$$

Here p_l, p_u, A_l, and A_u are the discretization of the spectral mask constraint.

The rotated second-order cone can be converted to a standard second-order cone:

$$\begin{pmatrix} \gamma \\ \gamma_u \\ r_g \end{pmatrix} \in R - \text{cone} \iff \begin{pmatrix} \hat{\gamma} \\ \hat{\gamma}_u \\ r_g \end{pmatrix} \in SO, \quad \gamma = \frac{1}{\sqrt{2}}(\hat{\gamma} - \hat{\gamma}_u), \gamma_u = \frac{1}{\sqrt{2}}(\hat{\gamma} + \hat{\gamma}_u) \tag{19}$$

For our final formulation we also note that in the `filter48` problem, the inequality involving ϵ is replaced with an equality and $r_g[0]$ is used in place of 1. We now have:

$$\begin{array}{ll} \min\limits_{\mu,\hat{\gamma},\hat{\gamma}_u,\zeta,r_g,s_1,s_2,s_3,s_4,X} & \frac{1}{\sqrt{2}}(\hat{\gamma} - \hat{\gamma}_u) \\ \text{subject to} & \\ & r_g[Ni] + \mu_i - s_1 = 0, \qquad i = 1,\ldots,k \\ & r_g[Ni] - \mu_i + s_2 = 0, \qquad i = 1,\ldots,k \\ & \sum_{i=1}^{k} \mu_i - \epsilon = 0, \quad r_g[0] = 1, \quad \frac{1}{2} r_g[0] - \frac{1}{\sqrt{2}}(\hat{\gamma} + \hat{\gamma}_u) = 0 \\ & \zeta p_l - A_l r_g + s_3 = 0, \quad -\zeta p_u + A_l r_g + s_4 = 0 \\ & X \bullet I - r_g[0] = 0, \quad \frac{1}{2} X \bullet O_i - r_g[i] = 0, \qquad i = 1,\ldots,L-1 \\ & \zeta, \mu, s_1, s_2, s_3, s_4 \geq 0, \quad X \succeq 0 \\ & \begin{pmatrix} \hat{\gamma} \\ \hat{\gamma}_u \\ r_g \end{pmatrix} \in SO \end{array} \tag{20}$$

The data for `filter48` comes from example 1 in [Dav00]. The filter to be designed is a robust variant of a filter specified in the IS95 standard. The parameters are $N = 4$, $L = 48$, and $\epsilon = 0.0245$. The spectral mask is shown figure 3 of [Dav00]. `filter48` uses 896 grid points for the discretization.

8.2 `filtinf1`

The `filtinf1` problem originates in example 2 of [Dav00]. The objective of the design process in this example is to improve upon a filter specified in the European UMTS proposal. The design parameter is the spectral mask of the filter. The sensitivities are set to those attained by the proposed filter. The optimization involves a bisection search over the design parameters. At each step, a feasibility problem is solved to determine if a filter with the specified parameters exists. The model is close to (17), i.e. the objective is to decide feasibility of the system:

$$
\begin{gathered}
-\mu_i \leq r_g[Ni] \leq \mu_i, \qquad i = 1, \ldots, k \\
\sum_{i=1}^{k} \mu_i = \epsilon_{\mathrm{UMTS}} \\
\|r_g\| \leq \gamma_{\mathrm{UMTS}}, \quad r_g[0] = 1 \\
A_l r_g \geq \zeta p_l, \quad A_l r_g \leq \zeta p_u \\
X \bullet I = 1, \quad \frac{1}{2} X \bullet O_i = r_g[i], \qquad i = 1, \ldots, L-1 \\
\zeta \geq 0, \mu \geq 0, \quad X \succeq 0
\end{gathered}
$$

where ϵ_{UMTS} and γ_{UMTS} are the sensitivities of the original filter from the UMTS proposal, and p_l, A_l, p_u, A_u are the discretization of the chosen spectral mask. In standard form, this becomes the following problem:

$$
\begin{aligned}
\min_{\mu,\zeta,r_g,s_1,s_2,s_3,s_4,s_q,X} \quad & 0 \\
\text{subject to} \quad & \\
& r_g[Ni] + \mu_i - s_1 = 0, \qquad i = 1, \ldots, k \\
& r_g[Ni] - \mu_i + s_2 = 0, \qquad i = 1, \ldots, k \\
& \sum_{i=1}^{k} \mu_i - \epsilon_{\mathrm{UMTS}} = 0, \quad r_g[0] = 1, \quad s_q = \gamma_{\mathrm{UMTS}} \\
& \zeta p_l - A_l r_g + s_3 = 0, \quad -\zeta p_u + A_l r_g + s_4 = 0 \\
& X \bullet I - r_g[0] = 0, \quad \frac{1}{2} X \bullet O_i - r_g[i] = 0, \qquad i = 1, \ldots, L-1 \\
& \zeta, \mu, s_1, s_2, s_3, s_4 \geq 0, \quad X \succeq 0, \quad \begin{pmatrix} s_q \\ r_g \end{pmatrix} \in SO
\end{aligned}
\tag{21}
$$

In `filtinf1` the parameters are $L = 49$, $N = 4$, $k = \lceil (L-1)/N \rceil$, $\epsilon_{\mathrm{UMTS}} = 0.0163$, and $\gamma_{\mathrm{UMTS}} = 1.1795$. The spectral mask uses 907 grid points and is shown in figure 6 in [Dav00]. The problem is infeasible, but very close to feasibility.

8.3 `minphase`

In the previous two problems, the design variables are the autocorrelation coefficients. Since a filter is not uniquely defined by its autocorrelation, there is still some freedom in choosing a filter once a solution to the previous problems has been obtained. A filter can be obtained from the autocorrelation by computing a spectral factor. One desirable such factor is the minimum phase spectral factor. It can be computed with the following SDP, again using theorem 4.1 from [DLS00]:

$$
\begin{aligned}
&\min_X \sum_{i=1}^{L} (L-i) X_{i,i} \\
&\text{subject to} \\
&\qquad X \bullet i = r_g[0] \\
&\qquad \frac{1}{2} X \bullet O_k = r_g[k] \qquad k = 1, \ldots, L-1 \\
&\qquad X \succeq \mathbf{0}
\end{aligned}
\tag{22}
$$

In the `minphase` problem, $L = 48$ and r_g is taken from a solution of `filter48`.

9 The rest of the problems

The *truss* and *hinf* sets were taken from SDPLIB [Bor98].

References

[Dav00] T. N. Davidson. Efficient evaluation of trade-offs in waveform design for robust pulse amplitude modulation. In *Proceedings of the 2000 International Conference on Acoustics, Speech, and Signal Processing*, volume V, pages 2789–2792, Istanbul, Turkey, June 2000.

[DLS00] T. N. Davidson, Z.-Q. Luo, and J. F. Sturm. Linear matrix inequality formulation of spectral mask constraints, September 2000. 30 pages. Submitted to the *IEEE Transactions on Sigmal Processing.* Also: `http://www.crl.mcmaster.ca/ASPC/people/davidson/LMImasks.ps`.

[DLW99] T.N. Davidson, Z.-Q. Luo, and K.M. Wong. Spectrally-Efficient Orthogonal Pulse Shape Design via Semidefinite Programming. Technical report, Communications Research Laboratory, McMaster University, Hamilton, Ontario, Canada, 1999. Submitted to IEEE Transactions on Signal Processing.

[Stu99] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.*, 11–12:625–653, 1999. Special issue on Interior Point Methods (CD supplement with software).

[AlKa95] C. J. Alpert and A. B. Kahng: Recent Directions in Netlist Partitioning: A Survey, *Integration: the VLSI Journal*, 19(1-2), 1995, pp. 1-81.

[AnChOv98] Knud D. Andersen, Edmund Christiansen and Michael L. Overton: Computing limit loads by minimizing a sum of norms, *SIAM J. Sci. Comput.* 19(8), 1998, pp. 1046-1062.

[Bor98] B. Borchers: SDPLIB 1.2, a library of semidefinite programming test problems, *Optimization Methods and Software*, volume 12, pages 683-690, 1999.

[ChAn98] Edmund Christiansen and Knud D. Andersen: Computation of collapse states with von Mises type yield condition, preprint 1998.

[ChoYe99] Cris Choi and Yinyu Ye: Application of Semidefinite Programming to Circuit Partitioning, Technical report, Department of Management Sciences ,The University of Iowa

[Circ] The Circuit Partitioning Page, *http://vlsicad.cs.ucla.edu/ cheese/benchmarks.html*

[1] Jeffrey O. Coleman and Robert J. Vanderbei, "Random-Process Formulation of Computationally Efficient Performance Measures for Wideband Arrays in the Far Field", *Proc. of The 1999 Midwest Symposium on Circuits and Systems*, New Mexico State University, Aug. 19999

[DeSietal955] C. DeSimone, M. Diehl, M. Junger, P. Mutzel, G. Reinelt, and G. Rinaldi : Exact ground states of Ising spin glasses: new experimental results with a branch-and-cut algorithm, *Journal of Statistical Physics*, 80, 1995, pp. 487-496.

NAME	ROWS	SDP	QUADR	LIN	OPT VALUE
toruspm3-8-50	512	[1; 512]	-	-	527.808663
toruspm3-15-50	3,375	[1; 3,375]	-	-	3474.8
torusg3-8	512	[1; 512]	-	-	457.358179
torusg3-15	3,375	[1; 3,375]	-	-	3134.6
bm1	883	[1; 882]	-	-	23.43982
biomedP	6,515	[1; 6,514]	-	-	33.6
industry2	12,638	[1; 12,637]	-	-	65.6
fap09	15,225	[1; 174]	-	14,025	10.8
fap25	2,244,021	[1; 2,118]	-	2,232,141	12.5 (lb, not opt)
fap36	8,448,105	[1; 4,110]	-	8,405,931	63.7 (lb, not opt)
fap-sup-25	322,924	[1; 2,118]	-	311,044	12.5 (lb, not opt)
fap-sup-36	1,154,467	[1; 4,110]	-	1,112,293	63.7 (lb, not opt)
hamming_9_8	2,305	[1; 512]	-	-	224
hamming_10_2	23,041	[1; 1,024]	-	-	102.4
hamming_11_2	56,321	[1; 2,048]	-	-	170 2/3
hamming_7_5_6	1,793	[1; 2,048]	-	-	42 2/3
hamming_8_3_4	16,129	[1; 256]	-	-	25.6
hamming_9_5_6	53,761	[1; 256]	-	-	85 1/3
copo14	1,275	$[14; 14 \times 14]$	-	364	0
copo23	5,820	$[23; 23 \times 23]$	-	1771	0
copo68	154,905	$[68; 68 \times 68]$	-	50,116	0
truss5	208	$[34; 33 \times 10, 1]$	-	-	132.6356779
truss8	496	$[34; 33 \times 19, 1]$	-	-	133.1145891
hinf12	43	$[3; 6, 6, 12]$	-	-	- 0.0231 (?)
hinf13	57	$[3; 7, 9, 14]$	-	-	-44.38 (?)

Table 6: Problem Statistics: Pure SDPs

NAME	ROWS	SDP	QUADR	LIN	OPT VALUE
filter48	969	$[1;48]$	$[1;49]$	931	1.41612901
filtinf1	983	$[1;49]$	$[1;49]$	945	primal inf.
minphase	48	$[1;48]$	-	-	5.98
nql30	3,680	-	$[900;900 \times 3]$	3,602	-0.9460
nql60	14,560	-	$[3600;3600 \times 3]$	14,402	-0.935423
nql180	130,080	-	$[32400;32400 \times 3]$	129,602	-0.927717
nql30old	3,601	-	$[900;900 \times 3]$	5,560	-0.9460
nql60old	14,401	-	$[3600;3600 \times 3]$	21,920	-0.935423
nql180old	129,601	-	$[32400;32400 \times 3]$	195,360	-0.927717
qssp30	3,691	-	$[1891;1891 \times 4]$	2	-6.496675
qssp60	14,581	-	$[7381;7381 \times 4]$	2	-6.562696
qssp180	130,141	-	$[65341;65341 \times 4]$	2	-6.639527
qssp30	5,674	-	$[1891;1891 \times 4]$	3,600	-6.496675
qssp60	22,144	-	$[7381;7381 \times 4]$	14,400	-6.562696
qssp180	196,024	-	$[65341;65341 \times 4]$	129,600	-6.639527
sched_50_50_orig	2527	-	[2;2474,3]	2,502	26,673
sched_100_50_orig	4844	-	[2;4741, 3]	5,002	181,889
sched_100_100_orig	8338	-	[2;8235, 3]	10,002	717,367
sched_200_100_orig	18087	-	[2;17884, 3]	20,002	141,360
sched_50_50_scaled	2526	-	2475	2,502	7.852038
sched_100_50_scaled	4843	-	4742	5,002	67.166281
sched_100_100_scaled	8337	-	8236	10,002	27.331457
sched_200_100_scaled	18086	-	17885	20002	51.812471
nb	123	-	$[793;793 \times 3]$	4	- 0.05070309
nb_L1	915	-	$[793;793 \times 3]$	797	- 13.01227
nb_L2	123	-	$[839;1 \times 1677, 838 \times 3]$	4	-1.628972
nb_L2_bessel	123	-	$[839;1 \times 123, 838 \times 3]$	4	-0.102571

Table 7: Problem Statistics: Mixed and Pure SOCPs

NAME	ORIGINATOR	FORMULATOR	DONATOR	REF
toruspm3-8	Michael Junger	Michael Junger	Michael Junger	2.1
toruspm3-15				
torusg3-8				
torusg3-15				
fap09	E-Plus Mobilfunk GmbH,	Christoph Helmberg	Christoph Helmberg	2.2
fap25	Andreas Eisenblatter			
fap36				
bm1		Cris Choi, Yinyu Ye	Cris Choi, Yinyu Ye	2.3
biomedP				
industry2				
nql30	Edmund Christiansen	Edmund Christiansen,	Erling Andersen	
nql60		Knud Andersen		
nql180				
qssp30	Edmund Christiansen	Edmund Christiansen	Erling Andersen	
qssp60		Knud Andersen		
qssp180				
truss5	SDPLIB problem set			
truss8				
hinf12	SDPLIB problem set			
hinf13				
filter48			Jos Sturm	
filtinf1				
minphase				

Table 8: Problem Origins

[FriJe95] A. Frieze and M. Jerrum: Improved approximation algorithms for max k-cut and max bisection, Proceedings of the 4th IPCO Conference, ppq. 1-13, Lecture Notes in Comput. Sci., Springer, 1995.

[Helm99] C. Helmberg, private communication.

[Par00] P. A. Parrilo: Semidefinite programmming based tests for matrix copositivity, *CDC 2000*

[Schil99] F. Schiller: Doctoral Dissertation, Technical University of Berlin, 1999

[Sku99] M. Skutella: Convex Quadratic and Semidefinite Programming Relaxations in Scheduling, technical report, Technical University Berlin

[Stu99] J. F. Sturm: Using Sedumi 1.02, a Matlab toolbox for optimization over symmetric cones, technical report, McMaster University, 1998