Solving Mixed-Integer Semidefinite Programs

Marc Pfetsch, TU Darmstadt

based on joint work with
Tristan Gally, Stefan Ulbrich, Frederic Matter, and Christopher Hojny

ICERM Workshop “Trends in Computational Discrete Optimization”

MISDP

TECHNISCHE UNIVERSITÄT DARMSTADT

We consider solving Mixed-Integer Semidefinite Programs (MISDPs):

$$\begin{aligned} \inf \quad & b^\top y \\ \text{s.t.} \quad & \sum_{k=1}^{m} A^k y_k - A^0 \succeq 0, \\ & \ell_i \le y_i \le u_i && \forall\, i \in [m], \\ & y_i \in \mathbb{Z} && \forall\, i \in I, \end{aligned}$$

- symmetric matrices $A^k \in \mathbb{R}^{n \times n}$ for $k \in [m]_0 := \{0, \dots, m\}$, $b \in \mathbb{R}^m$
- bounds: $\ell_i \in \mathbb{R} \cup \{-\infty\}$, $u_i \in \mathbb{R} \cup \{\infty\}$ for all $i \in [m] := \{1, \dots, m\}$.
- integer variables: $I \subseteq [m]$

MISDP

We consider solving Mixed-Integer Semidefinite Programs (MISDPs):

$$\begin{aligned} \inf \quad & b^\top y \\ \text{s.t.} \quad & \sum_{k=1}^{m} A^k y_k - A^0 \succeq 0, \\ & \ell_i \leq y_i \leq u_i && \forall\, i \in [m], \\ & y_i \in \mathbb{Z} && \forall\, i \in I, \end{aligned}$$

- symmetric matrices $A^k \in \mathbb{R}^{n \times n}$ for $k \in [m]_0 := \{0, \dots, m\}$, $b \in \mathbb{R}^m$
- bounds: $\ell_i \in \mathbb{R} \cup \{-\infty\}$, $u_i \in \mathbb{R} \cup \{\infty\}$ for all $i \in [m] := \{1, \dots, m\}$.
- integer variables: $I \subseteq [m]$

Linear constraints can be expressed as SDP blocks with diagonal entries only. Thus, Mixed Integer Programs (MIPs) are a special case.

Goals of This Talk

The goals of this talk are:

- Explain how MISDPs can be solved.
- Present several improvement techniques:
  - Dual Fixing
  - Presolving
  - Conflict Analysis
  - Symmetry Handling
- Evaluate performance.
- Discuss similarities and differences to mixed-integer programming.

Following the title of the workshop, this talk will focus on computational aspects.

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Applications
2. Solution Methods
3. Dual Fixing
4. Presolving MISDPs
   - Bound Tightening
   - Computational Results
5. Conflict Analysis
6. Symmetry
7. Conclusions

Applications

TECHNISCHE UNIVERSITÄT DARMSTADT

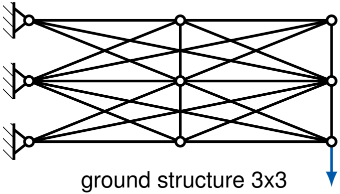
MISDPs have many applications:

- Quadratic TSP [Renata Sotirov $\rightarrow$ earlier ICERM workshop]
- Cardinality Constrained Least Squares
- Minimum k-Partitioning
- Computing restricted isometry constants in compressed sensing
- Optimal transmission switching problem in AC power flow
- Robustification of physical parameters in gas networks
- Subset selection for eliminating multicollinearity
- ...

Application Example: Robust Truss Topology Design

TECHNISCHE UNIVERSITÄT DARMSTADT

- n nodes $V \subset \mathbb{R}^d$
- n_f free nodes $V_f \subset V$
- m possible trusses E
- forces $f \in \mathbb{R}^{d_f}$ for $d_f = d \cdot n_f$



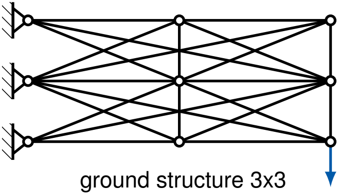
ground structure 3x3

Application Example: Robust Truss Topology Design

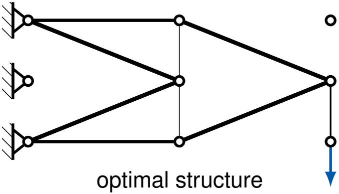
TECHNISCHE UNIVERSITÄT DARMSTADT

- n nodes $V \subset \mathbb{R}^d$
- n_f free nodes $V_f \subset V$
- m possible trusses E
- forces $f \in \mathbb{R}^{d_f}$ for $d_f = d \cdot n_f$

- Choose cross-sectional areas for trusses minimizing volume while creating a “stable” truss.
- Stability is measured by the compliance $\frac{1}{2} f^T u$ with node displacements u.



ground structure 3x3



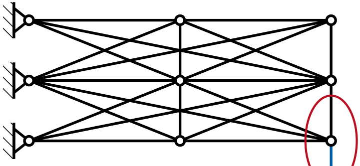
optimal structure

Application Example: Robust Truss Topology Design

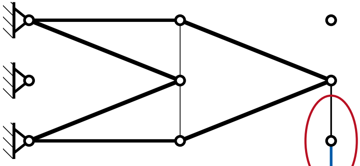
TECHNISCHE UNIVERSITÄT DARMSTADT

- n nodes $V \subset \mathbb{R}^d$
- n_f free nodes $V_f \subset V$
- m possible trusses E
- forces $f \in \mathbb{R}^{d_f}$ for $d_f = d \cdot n_f$

- Choose cross-sectional areas for trusses minimizing volume while creating a “stable” truss.
- Stability is measured by the compliance $\frac{1}{2} f^T u$ with node displacements u.



ground structure 3x3



optimal structure

- Use uncertainty set $\{f \in \mathbb{R}^{d_f} \; : \; f = Qg : \|g\|_2 \leq 1\}$ instead of single force f.
- Restrict cross-sectional areas $x \in \mathbb{R}^m_+$ to a discrete set $\mathcal{A}$.

Application Example: Robust Truss Topology Design

TECHNISCHE UNIVERSITÄT DARMSTADT

Elliptic Robust Discrete TTD [Ben-Tal/Nemirovski 1997; Mars 2013]

$$
\begin{aligned}
\inf \quad & \sum_{e \in E} \ell_e \sum_{a \in \mathcal{A}} a\, x_e^a \\
\text{s.t.} \quad & \begin{pmatrix} 2C_{\max} I & Q^T \\ Q & S(x) \end{pmatrix} \succeq 0, \\
& \sum_{a \in \mathcal{A}} x_e^a \leq 1 && \forall e \in E, \\
& x_e^a \in \{0, 1\} && \forall e \in E, a \in \mathcal{A},
\end{aligned}
$$

with truss lengths ℓ_e, upper bound $C_{\max}$ on compliance and stiffness matrix

$$S(x) = \sum_{e \in E} \sum_{a \in \mathcal{A}} S_e\, a\, x_e^a$$

for positive semidefinite, rank-one single truss stiffness matrices S_e.

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

Solving Methods for MISDPs

TECHNISCHE UNIVERSITÄT DARMSTADT

1. SDP-based branch-and-bound: Solve SDP-relaxations (special case of NLP-based B&B [Dakin 1965])
2. LP-based branch-and-bound: Cutting plane method based on LP-relaxations [Sherali and Fraticelli 2002]; [Krishnan and Mitchell 2006]
3. Outer approximation: Solve MIP-relaxations [Duran and Grossmann 1986].

Implementations:

1. YALMIP [Löfberg 2004] and SCIP-SDP
2. YALMIP and SCIP-SDP
3. Pajarito [Coey, Lubin, and Vielma 2020]

SDP-based Branch-and-Bound

TECHNISCHE UNIVERSITÄT DARMSTADT

- Relax integrality.
- Branch on integral y-variables.
- Need to solve a continuous SDP in each branch-and-bound node.
- Relaxations can be solved by problem-specific approaches (e.g. conic bundle or low-rank methods) or interior-point solvers.
- Convergence assumptions of SDP-solvers should be satisfied.
- Usually much slower than solving LPs and no warmstart.

LP-based Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

For LP-based approach and outer approximation:

- ▷ Usual approach for convex MINLP: gradient cuts

$$g_j(\overline{x}) + \nabla g_j(\overline{x})^\top (x - \overline{x}) \leq 0.$$

- ▷ But function of smallest eigenvalue is not differentiable everywhere.
- ▷ Instead use characterization $X \succeq 0 \quad \Leftrightarrow \quad u^\top X u \geq 0$ for all $u \in \mathbb{R}^n$.
- ▷ If $Z := \sum_{k=1}^m A^k y_k^* - A^0 \not\succeq 0$, compute eigenvector v to smallest eigenvalue. Then

$$v^\top Z v = \sum_{k=1}^m v^\top A^k v\, y_k - v^\top A^0 v \geq 0$$

is valid and cuts off $y^* \quad \rightarrow$ Eigenvector cut.

Cutting Planes: MISOCP vs. MISDP

TECHNISCHE UNIVERSITÄT DARMSTADT

- Cutting planes are often used by solvers for mixed-integer second-order cone problems (MISOCPs).
- Approximation for SOCPs possible with polynomial number of cuts [Ben-Tal/Nemirovski 2001].
- Approximation for SDPs needs exponential number of cuts:

Theorem ([Braun, Fiorini, Pokutta, Steurer 2015])

There are SDPs of dimension $n \times n$ for which any polyhedral approximation is of size $2^{\Omega(n)}$.

SCIP-SDP

TECHNISCHE UNIVERSITÄT DARMSTADT

Our solver: SCIP-SDP

- ▷ Based on SCIP (`www.scipopt.org`)
- ▷ Supports both SDP-based B&B and LP-based branch-and-cut.
- ▷ Introduced by [Mars 2013], continued by [Gally 2019] and Matter [2022], ...
- ▷ Apache 2.0 license.
- ▷ Current version: 4.1: `wwwopt.mathematik.tu-darmstadt.de/scipsdp`
- ▷ Approximately 50 000 lines of C-code
- ▷ SDP-solvers: interfaces to Mosek, DSDP, SDPA
- ▷ Matlab-Interface: `github.com/scipopt/MatlabSCIPInterface`

Computations:

- ▷ Use SCIP developer version (8.0.3).
- ▷ Use Mosek 9.2.40 for solving SDP-relaxations.
- ▷ Linux cluster with 3.5 GHz Intel Xeon E5-1620 Quad-Core CPUs.
- ▷ Nodes and times are shifted geometric means.
- ▷ Time limit 1 h.

Comparison of SDP and LP-based Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

Testset: 185 instances from different sources.

type	# solved	# nodes	time
SDP	167	1066.1	132.2
LP	109	419.2	336.5
all optimal (106):			
SDP		605.0	93.2
LP		507.0	63.2

Conclusions:

- LP-based approach solves significantly less instances.
- On the instances solved by both, it is faster by 32 % and uses less nodes.
- Open question: Predict which method is faster and explain why.

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Fixing

- Extension of reduced-cost fixing to general MINLPs by [Ryoo and Sahinidis 1996] and primal MISDPs by [Helmberg 2000].
- Our approach uses conic duality and only requires feasibility.

Theorem [Gally, P., Ulbrich 2018]

- (X, W, V): Primal feasible solution, where W, V are primal variables corresponding to variable bounds ℓ, u in the dual,
- f: Corresponding primal objective value,
- L: Lower bound on the optimal objective value of the MISDP.

Then for every optimal solution $y^\star$ of the MISDP

$$y_j^\star \leq \ell_j + \frac{f - L}{W_{jj}} \text{ if } \ell_j > -\infty \quad \text{and} \quad y_j^\star \geq u_j - \frac{f - L}{V_{jj}} \text{ if } u_j < \infty.$$

- If $f - L < W_{jj}$ for binary y_j, then $y_j^\star = 0$, if $f - L < V_{jj}$, then $y_j^\star = 1$.
- 9% reduction of B&B-nodes, 23% speedup [Gally et al. 2018].

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

Presolving

TECHNISCHE UNIVERSITÄT DARMSTADT

- Motivated by presolving for mixed-integer (linear) programs (MIPs): [Bixby and Rothberg 2007]: slow-down of 10.8 when turning off presolving.
- Previous work for MISDPs: [Mars 2013], [Gally 2019], [Gally et al. 2017].
- We introduced several new methods and generalizations from MIP methods.
- Speed-up of presolving depends very strongly on instances.
  One reason: Instances are usually "hand-crafted".
  $\rightsquigarrow$ Cannot expect similar speed-ups as for MIPs.
- We also apply most methods in each node (node presolving).
- Standard presolving is applied as well: linear constraints, aggregations, ...
- Concentrate here on one particular method: bound tightening.

Bound Tightening I

TECHNISCHE UNIVERSITÄT DARMSTADT

For an index $k \in [m]$, define

$$P_k := \{i \in [m] \setminus \{k\} \;:\; A^i \succeq 0\}, \qquad N_k := \{i \in [m] \setminus \{k\} \;:\; A^i \preceq 0\},$$

as well as

$$\underline{\mu}_k := \inf \Big\{\mu \;:\; A^k \mu + \sum_{i \in P_k} A^i u_i + \sum_{j \in N_k} A^j \ell_j - A^0 \succeq 0\Big\},$$

$$\overline{\mu}_k := \sup \Big\{\mu \;:\; A^k \mu + \sum_{i \in P_k} A^i u_i + \sum_{j \in N_k} A^j \ell_j - A^0 \succeq 0\Big\}$$

or $\pm\infty$ if $\pm\infty$ occurs in bounds (ℓ, u).

Bound Tightening II

TECHNISCHE UNIVERSITÄT DARMSTADT

Lemma (Tighten Bounds (TB))

Let all matrices be (positive or negative) semidefinite. Then, $\underline{\mu}_k \leq y_k \leq \overline{\mu}_k$ *is valid for all* $k \in [m]$*. We can round bounds for integral variables.*

Proof.

Suppose that $y_k < \underline{\mu}_k$ or $y_k > \overline{\mu}_k$. Then there exists $x \in \mathbb{R}^n$ with

$$\begin{aligned}
0 &> x^\top \Big(A^k\, y_k + \sum_{i \in P_k} A^i\, u_i + \sum_{i \in N_k} A^i\, \ell_i - A^0 \Big) x \\
&= x^\top A^k x\, y_k + \sum_{i \in P_k} \underbrace{x^\top A^i x}_{\geq 0}\, u_i + \sum_{i \in N_k} \underbrace{x^\top A^i x}_{\leq 0}\, \ell_i - x^\top A^0 x \\
&\geq x^\top A^k x\, y_k + \sum_{i \in P_k} x^\top A^i x\, y_i + \sum_{i \in N_k} x^\top A^i x\, y_i - x^\top A^0 x. \ ↯
\end{aligned}$$

□

One-Variable SDPs

TECHNISCHE UNIVERSITÄT DARMSTADT

- For computing bound tightenings, need to solve one-variable SDPs.

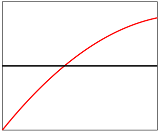
$$\inf \{\mu \,:\, \mu A - B \succeq 0,\ \ell \leq \mu \leq u\}.$$

  for symmetric $A, B \in \mathbb{R}^{n\times n}$.
- Can easily see: $\mu \mapsto \lambda_{\min}(\mu A - B)$ is concave.
- Let $\hat{v}$ be a unit eigenvector for $\lambda_{\min}(\hat{\mu} A - B)$ for $\hat{\mu} \in \mathbb{R}$. Then $\hat{v}^\top A \hat{v}$ is a supergradient, i.e.,

$$\lambda_{\min}(\mu A - B) \leq \lambda_{\min}(\hat{\mu} A - B) + (\mu - \hat{\mu})\, \hat{v}^\top A \hat{v}$$

  for all $\mu \in \mathbb{R}$.

- Goal: Want increase μ from ℓ until $\lambda_{\min}(\mu A - B) = 0$.
- Yields semismooth Newton algorithm …

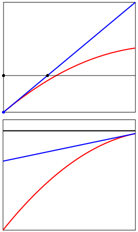


One-Variable SDPs

TECHNISCHE UNIVERSITÄT DARMSTADT

v_k = eigenvector for $\lambda_k := \lambda_{\min}(A\mu_k - B)$

$\mu_{k+1} = \mu_k - \frac{\lambda_k}{(v^k)^\top A v^k}$



Handle easy cases, e.g., infeasible if $\lambda_{\min}(A\,u - B) < 0$, supergradient positive.

- Always converges.
- Converges Q-superlinearly to a zero $\mu^\star$ of $f(\mu) = \lambda_{\min}(\mu\,A - B)$, given that $\partial f(\mu^\star)$ is nonsingular and the starting point lies near $\mu^\star$ [Qi and Sun, 1993].
- Very fast in practice; bottleneck: eigenvector computation …

Condensed Computational Results

TECHNISCHE UNIVERSITÄT DARMSTADT

Testset with 185 instances, results from [Matter and P. 2023]:

Setting	# solved	# nodes	time
nopresol	168	1405.3	180.23
bound tightening	167	1297.6	152.43
MIX	167	1085.2	139.52

- Bound tightening applied in every node produces a speed-up of about 7 %.
- MIX includes bound tightening and several other methods.
  It produces a speed-up of about 22 %.

- Some techniques do not do anything on some instances.
- The methods are effective if they can be applied and induce a small time overhead.

Overview

Conflict Analysis I

TECHNISCHE UNIVERSITÄT DARMSTADT

- The original idea is to learn from infeasible nodes in a branch-and-bound-tree.
- Idea transferred from SAT-solving to MIPs by [Achterberg 2007].
- More generally, can be seen as a way to learn cuts from solutions of the duals $\rightarrow$ similar to “dual ray/solution analysis” [Witzig et al. 2017, Witzig 2021]

Conflict Analysis II

Consider

$$\inf\left\{b^\top y \;:\; A(y) := \sum_{k=1}^{m} A^k y_k - A^0 \succeq 0,\; Dy \geq d,\; \ell \leq y \leq u\right\}$$

and $\hat{X} \succeq 0$, $\hat{z}$, $\hat{r}^\ell$, $\hat{r}^u \geq 0$. Aggregation yields:

$$\langle A(y), \hat{X}\rangle + \hat{z}^\top Dy + (\hat{r}^\ell)^\top y - (\hat{r}^u)^\top y \;\geq\; \hat{z}^\top d + (\hat{r}^\ell)^\top \ell - (\hat{r}^u)^\top u.$$

Idea: Do not add this (redundant) inequality, but perform bound propagation, taking integrality conditions into account.

Could also use CMIR on generated inequality as alternative to generalization of Gomory cuts [Sotirov and de Meijer 2022]. This seems not to be effective.

Conflict Analysis III

TECHNISCHE UNIVERSITÄT DARMSTADT

The dual can provide $(\hat{X}, \hat{z}, \hat{r}^\ell, \hat{r}^u)$:

$$\begin{aligned} \sup \quad & \langle A^0, X\rangle + z^\top d + \ell^\top r^\ell - u^\top r^u \\ \text{s.t.} \quad & \langle A^j, X\rangle + (D^\top z)_j + r^\ell_j - r^u_j = b_j \quad \forall\, j \in [m], \\ & X \succeq 0,\ z,\ r^\ell,\ r^u \geq 0. \end{aligned}$$

Similarly for a primal ray satisfying:

$$\begin{aligned} & \langle A^j, X\rangle + (D^\top z)_j + r^\ell_j - r^u_j = 0 \qquad \forall\, j \in [m], \\ & \langle A^0, X\rangle + d^\top z + \ell^\top r^\ell - u^\top r^u > 0, \\ & X \succeq 0,\ z,\ r^\ell,\ r^u \geq 0. \end{aligned}$$

Lemma

Let $(\hat{X}, \hat{z}, \hat{r}^\ell, \hat{r}^u)$ be a primal ray. Then the aggregated inequality is infeasible with respect to the local bounds ℓ and u.

Conflict Analysis – Computations

TECHNISCHE UNIVERSITÄT DARMSTADT

Generate a conflict constraint for each feasible or infeasible node. Store them as constraints and perform bound propagation.

type	# solved	# nodes	time
default	167	1066.1	132.2
conflicts	168	989.6	122.2
all optimal (167):			
default		788.7	94.2
conflicts		726.3	86.4

- Using conflicts provides a speed-up and node-reduction of about 8 %.
- On average 12792.0 conflict constraints are generated per instance.
- Average number of conflict constraints per node: 1.25.

Overview

Symmetry Detection

TECHNISCHE UNIVERSITÄT DARMSTADT

Goal: apply known symmetry handling methods.

For a permutation σ of $[n]$:

$$\sigma(A)_{ij} = A_{\sigma^{-1}(i),\sigma^{-1}(j)} \quad \forall i, j \in [n].$$

Definition

A permutation $\pi \in \mathcal{S}_m$ of variables is a formulation symmetry if there exists a permutation $\sigma \in \mathcal{S}_n$ such that

1. $\pi(I) = I$, $\pi(\ell) = \ell$, $\pi(u) = u$, and $\pi(b) = b$
   (π leaves integer variables, variable bounds, and the objective coefficients invariant),
2. $\sigma(A^0) = A^0$ and, for all $i \in [m]$, $\sigma(A^i) = A^{\pi^{-1}(i)}$.

Such symmetries can be detected by using graph automorphism algorithms.

Symmetry: Computed Symmetries

TECHNISCHE UNIVERSITÄT DARMSTADT

instance	symmetry group
0+-115305C_MISDPld000010	$\mathcal{S}_2$
0+-115305C_MISDPrd000010	$\mathcal{S}_2$
band60605D_MISDPld000010	$\mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_{10} \times \mathcal{S}_3 \times \mathcal{S}_4$
band60605D_MISDPrd000010	$\mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_{10} \times \mathcal{S}_3 \times \mathcal{S}_4$
band70704A_MISDPld000010	$\mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_3 \times \mathcal{S}_3$
band70704A_MISDPrd000010	$\mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_3 \times \mathcal{S}_3$
clique_60_k10_6_6, clique_60_k15_4_4, clique_60_k20_3_3, clique_60_k4_15_15, clique_60_k5_12_12, clique_60_k6_10_10, clique_60_k7_8_9, clique_60_k8_7_8, clique_60_k9_6_7, clique_70_k3_23_24	$\mathcal{S}_2$
diw_34	$\mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{D}_4 \times \mathcal{S}_4 \times \mathcal{S}_4$
diw_37	$\mathcal{S}_2 \times \mathcal{S}_4 \times \mathcal{S}_3 \times \mathcal{S}_4$
diw_38	$\mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_2 \times \mathcal{S}_3$
diw_43	$\mathcal{S}_3$
diw_44	$\mathcal{S}_3$

$\mathcal{S}_k$ = full symmetric group on k elements; $\mathcal{D}_k$ = dihedral group.

Symmetry: Computational Results

TECHNISCHE UNIVERSITÄT DARMSTADT

Results from [Hojny and P. 2023]:

	all (184)			all optimal (168)		only symmetric (21)
	time (s)	symtime (s)	# gens	time (s)	#nodes	time (s)
without	130.6	–	–	95.0	778.3	45.07
with	125.3	0.44	99	90.8	760.6	29.84

- Speed-up of about 4 % for all instances;
- Speed-up of about 34 % for the 21 instances that contain symmetry.
- Number of generators is quite small.
- Note that we do not exploit symmetries in the solutions of the SDPs (yet).

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

Conclusions and Outlook

TECHNISCHE UNIVERSITÄT DARMSTADT

Take away messages:

- Several methods help to improve performance.
- Most of the methods are effective only on certain instances, but do not incur a significant overhead.
- Solving LP-relaxations is only helpful for certain instance types.
- Solving SDPs is still one bottleneck, but often yields strong bounds.
- Generic MISDP-solver SCIP-SDP is available.

Future work:

- More methods are likely to be helpful – probably motivated by particular structures.
- Investigate SOCP relaxations for propagation.
- We are always interested in new instances.

Conclusions and Outlook

Take away messages:

- ▷ Several methods help to improve performance.
- ▷ Most of the methods are effective only on certain instances, but do not incur a significant overhead.
- ▷ Solving LP-relaxations is only helpful for certain instance types.
- ▷ Solving SDPs is still one bottleneck, but often yields strong bounds.
- ▷ Generic MISDP-solver SCIP-SDP is available.

Future work:

- ▷ More methods are likely to be helpful – probably motivated by particular structures.
- ▷ Investigate SOCP relaxations for propagation.
- ▷ We are always interested in new instances.

Thank you for your attention!

References

- T. Gally.
  Computational Mixed-Integer Semidefinite Programming.
  PhD thesis, TU Darmstadt, 2019.
- T. Gally, M. E. Pfetsch, and S. Ulbrich.
  A framework for solving mixed-integer semidefinite programs.
  Optimization Methods and Software, 33(3):594–632, 2018.
- C. Hojny and M. E. Pfetsch.
  Handling Symmetries in Mixed-Integer Semidefinite Programs.
  Technical report, Optimization Online, CPAIOR 2023, 2023.
- F. Matter and M. E. Pfetsch.
  Presolving for mixed-integer semidefinite optimization.
  INFORMS Journal on Optimization, 2023, to appear.