Scoring positive semidefinite cutting planes for quadratic optimization via trained neural networks

Radu Baltean-Lugojan · Pierre Bonami · Ruth Misener · Andrea Tramontani

November 19, 2019

Abstract Semidefinite programming relaxations complement polyhedral relaxations for quadratic optimization, but global optimization solvers built on polyhedral relaxations cannot fully exploit this advantage. This paper develops linear outer-approximations of semidefinite constraints that can be effectively integrated into global solvers. The difference from previous work is that our proposed cuts are (i) sparser with respect to the number of nonzeros in the row and (ii) explicitly selected to improve the objective. We select which cuts to generate using objective structure and explore engineering trade-offs for sparsity patterns, e.g. cut dimensionality and chordal extensions. A neural network estimator is key to our cut selection strategy: ranking each cut based on objective improvement involves solving a semidefinite optimization problem, but this is an expensive proposition at each Branch&Cut node. The neural network estimator, trained a priori of any instance to solve, takes the computation offline by predicting the objective improvement for any cut. Most of this paper discusses quadratic programming with box constraints as a test bed for the relevant engineering trade-offs. We also show how the engineering decisions may immediately extend to quadratically constrained instances, particularly ones with rich quadratic objective structure.

Keywords mixed-integer nonlinear programming · quadratic programming · cutting planes · convex and semidefinite relaxations · neural networks · cut selection

Mathematics Subject Classification (2000) 90C26 · 90C57 · 90C20 · 90C22

1 Introduction

Nonconvex quadratic programming is fundamental to optimization and pervasive in diverse applications [79, 108], but solving such NP-hard [112] problems at large scale to global optimality

This work was funded by an Engineering & Physical Sciences Research Council (EPSRC) DTP (award ref. 1675949) and an IBM PhD Fellowship to RBL and EPSRC Research Fellowship to RM (grant number EP/P016871/1). This manuscript significantly benefited from discussions at Dagstuhl Seminar 18081.

{radu.baltean-lugojan09, r.misener}@imperial.ac.uk
Department of Computing, Imperial College London, 180 Queen's Gate, London, SW7 2AZ, UK
pierre.bonami@es.ibm.com
CPLEX Optimization, IBM Spain, Madrid, Spain
andrea.tramontani@it.ibm.com
CPLEX Optimization, IBM Italy, Bologna, Italy

is challenging. From an MINLP perspective [22], mixed-integer quadratic instances, e.g. QPLib [45], are typically solved using finite Branch&Cut frameworks mixing different relaxation classes. State-of-the-art solvers tackle continuous nonconvexity through computationally light relaxations (see Section 2 for details) that fuse well with the powerful and expansive cutting plane machinery for mixed-integer programming (MIP). A longstanding research effort integrates strong continuous quadratic relaxations into Branch&Cut, e.g. copositive [31] or semidefinite [11, 40]. These cutting surfaces include: (i) directly semidefinite [26, 28, 46], (ii) convex outer-approximations [39, 42, 100, 101], and (iii) linear outer-approximations [94, 105]. But the trade-off between bound strength and computational tractability in these convex nonlinear relaxations is not trivial. Copositive and semidefinite relaxations typically offer strong bounds for a heavier computational cost, although binary quadratic programs are a notable exception [17, 25, 65, 72, 95, 117].

Our paradigm, driven by our goal of effectively integrating semidefinite cutting surfaces into state-of-the-art global solvers, incorporates the following desiderata:

1. **Easy integration into current technology**, i.e. the cutting surface should integrate well into state-of-the-art MINLP Branch&Cut technology and complement existing relaxations.
2. **Computationally light relaxations**, i.e. neither the cutting surface generation nor its incorporation into the Branch&Cut node should dominate solver execution time. An expensive approximation may be useful for a specific application, but general-purpose solvers cannot invest too much time in possibly unproductive cutting surfaces.
3. **Relatively few cuts** should approximate the relaxation, i.e. the *computationally light relaxations* should not become computationally heavy via an excessive number of cuts.
4. **Cutting surface generation should converge** to the desired relaxation, i.e. the relaxation should not miss critical parts of the semidefinite relaxation.

These desiderata are familiar algorithmic considerations in [35], but we require a significantly augmented framework for quadratic optimization. In the spirit of efficient second-order cone programming (SOCP) polyhedral relaxations [113], we chose cheap, linear outer-approximations of semidefinite relaxations that mix well with the cutting planes in solver software [18, 20, 82]. MINLP solvers use extended formulations [14, 59, 80, 81, 97, 110, 114], i.e. replace nonlinear term $x_i x_j$ with auxiliary variable X_{ij}, which we also adopt to integrate into current technology. Moreover, such extended formulations also allow us to fulfill the desiderata of using computationally light relaxations via low-dimensional or sparse cutting planes. Using sparse cuts drives MIP cut selection [35, 36, 115] and sparse cuts in extended formulations dominate cuts in the original polyhedron [36].

Developments in the semidefinite programming literature balance the conflicting desiderata of (3) using relatively few cuts versus (4) converging to the desired relaxation. This paper relaxes the semidefinite relaxation [6] into a collection of low-dimensional positive semidefinite (PSD) cones, based on submatrix partitioning. These low-dimensional cones are inspired by the d-exponential complexity of approximating d-dimensional convex bodies via polytopes [41, 52], empirically confirmed in PSD cone linearizations [94, 105]. A low-dimensional relaxation enables relatively few cuts, but we also want to converge, e.g. that a possibly greedy method does not miss important cuts. The sparse semidefinite literature [43, 44, 87] motivates cut selection with variables in chordal extensions for sparse/near-chordal instances and offers convergence guarantees.

MIP cut selection [35], i.e. avoiding to add all generated cuts, is an active research area. Cut selection leads to speedups [5] and enables early branching [21, 68]. Prior work explores difficult-to-find [92] cut selection measures, e.g. [23, 35, 116]. These contributions motivate selecting cuts maximizing objective improvement, e.g. [33], rather than those cutting random parts of feasible space. Most existing techniques focus on *feasibility cuts* selected based on feasibility violation

[23]. These feasibility techniques attempt heuristically to improve the objective via parallelism to it or cut orthogonality [5] while ignoring explicit structure. But our goal of developing relatively few cuts motivates directly our using the objective structure for *optimality(-based) selection* of localized *deep objective cuts* [23]. For quadratically-constrained quadratic optimization (QCQP), we use not only objective but also constraint structure, see Section 6. However, we find using objective information for cut selection leads to many small generic SDP subproblems that have to be solved repeatedly, a prohibitive proposition. To get around solving these small SDP repeatedly, we follow the current trend of applying machine learning in Branch&Cut, e.g. to select nodes [54, 96, 107], develop primal heuristics [56, 62, 106], or chose branching variables [2, 61, 74]. This paper takes cut selection complexity offline from the solver process via a trained (a priori and instance independent) neural network estimator, answering a hint to enlist machine learning in cut selection [35].

This paper focuses on outer-approximating the semidefinite relaxation by selecting sparse, linear cutting planes based on objective improvement via trained neural networks. The rest of the paper proceeds as follows: Section 2 introduces the background/notation for semidefinite (SDP) and other linear (base) relaxations the outer-approximation scheme relies on; Section 3 gives the theoretical framework for low-dimensional SDP relaxations and extracting optimality-based sparse cutting planes; Section 4 implements the neural network estimators to make optimality cut selection tractable, discussing their training and data sampling; Section 5 compares implemented cutting plane strategies on BoxQP instances, confirming in practice that: (i) neural networks are effective estimators, (ii) optimality cut selection identifies strong but not all feasibility-based cuts, and (iii) instance sparsity structure and cut sparsity influence cut selection strategies; Section 6 extends the selection of strong optimality cuts to QCQP and Section 7 concludes with further extensions. Source code reproducing our results is on Github [9].

2 Nonconvex quadratic formulations and relaxations

For the discussion in Sections 3 and 5, we first consider nonconvex quadratic problems that are box and linearly constrained, i.e.

$$z_{qp} = \min_x \left\{ x^T Q x + c^T x \mid Ax \leq b,\ x \in [0,1]^N \right\}, \tag{QP}$$

with an N-variable vector x, $A \in \mathbb{R}^{p \times N}$ and $Q \in \mathbb{R}^{N \times N}$ assumed to be an indefinite (symmetric) matrix. This class of problems encompasses many relevant instances, since the box constraints can be obtained by simple variable scaling from any closed and bounded domains [18, Appendix A.1]. We lift each quadratic term $x_i x_j$ by replacing it with a new variable X_{ij}. Let lifted variables $X_{ij}\ \forall i, j$ form symmetric matrix $X = xx^T$ and let $Q \bullet X = Tr(Q^T X) = \sum_{i,j} Q_{ij} X_{ij}$, representing the Frobenius inner product (applied to pairs of either matrices or vectors with the same dimensions). Then z_{qp} is lower-bounded by

$$z_{qp}(\mathcal{B}) := \min_{x,X} \left\{ Q \bullet X + c^T x \;\middle|\; Ax \leq b,\ x \in [0,1]^N \text{ and } (x, X) \in \mathcal{B} \right\},$$

parametric on any base convex set $\mathcal{B}$ so that $\left\{(x, X) \in \mathbb{R}^N \times \mathbb{R}^{N \times N} \middle| X = xx^T\right\} \subseteq \mathcal{B}$ and $z_{qp}(\mathcal{B})$ represents a QP relaxation. For notation convenience, to denote the intersection of any two relaxations $\mathcal{B}, \mathcal{B}'$, we interchange $\mathcal{B} \cap \mathcal{B}'$ with $\mathcal{B} + \mathcal{B}'$.

Let $G(V, E)$ denote the sparsity pattern graph induced by matrix Q (linking the original x variables via the extended space) where vertex set V and edge set E are defined as

$$V = \{1, 2, \ldots, N\}, \qquad E = \{\{i, j\} \in V \times V \mid i > j,\ Q_{ij} \neq 0\}.$$

We assume throughout that $G(V, E)$ has no disconnected vertex i; i.e. $\forall i \in V, \exists j \in V, j \neq i$ s.t. $Q_{ij} \neq 0$. Otherwise, the term $Q_{ii}x_i^2$ for a disconnected $i \in V$ is separable and its convex envelope is (i) linear if $Q_{ii} < 0$ or (ii) SOCP if $Q_{ii} > 0$. Let $G(V, \overline{E})$ ($E \subseteq \overline{E}$) be a chordal extension of $G(V, E)$, with $\overline{E}$ obtained by adding edges to E so that all cycles of four or more vertices have a chord (an edge not part of the cycle connecting two vertices), e.g. every induced cycle in $\overline{E}$ is a triangle. Additionally, define a full edge set $E^+ = \{\{i, j\} \in V \times V \mid i > j\}$ (as in [18]), with $E \subseteq \overline{E} \subseteq E^+$.

One common choice of $\mathcal{B}$ is to relax the nonconvex $X = xx^T$ to $X \succeq xx^T$, or equivalently (by Schur's complement) to $\begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \succeq 0$, giving the well-known semidefinite relaxation of QP. Furthermore, as [6] argues, the extra constraints $X_{ii} \leq x_i \ \forall i \in V$ (a subset of the reformulation linearization technique or RLT relaxations) are needed to bound the semidefinite relaxation. We denote the resulting relaxed set as $\mathcal{S}$:

$$\mathcal{S} := \left\{ (x, X) \middle| \begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix} \succeq 0, \ X_{ii} \leq x_i \ \forall i \in V \right\}.$$

The cutting plane approach in this paper relies on developing cuts in the variable space of extended formulations (both x and X), in order to be directly pluggable into extended linear relaxations employed at each node by Branch&Cut solvers such as Antigone [80, 81], Baron [97, 110], Couenne [14], CPLEX [59] or SCIP [1, 114]. This direction allows easy integration with other routinely used classes of cuts in the extended formulation space such as (i) the reformulation linearization technique (RLT) or the McCormick [75] hull [103], (ii) zero-half or triangle inequalities [18], and (iii) edge-concave cuts [10, 82]. To illustrate this potential, motivated by the findings in Anstreicher [6], in our implementation we augment any semidefinite relaxation such as $\mathcal{S}$ by the McCormick inequalities $\mathcal{M}$, given in our case by

$$\mathcal{M} := \left\{ (x, X) \middle| \begin{array}{cc} \forall i, j \in V \\ \{i, j\} \in E \end{array} : \begin{array}{r} X_{ij} \geq l_i x_j + l_j x_i - l_i l_j = 0, \\ X_{ij} \geq u_i x_j + u_j x_i - u_i u_j = x_i + x_j - 1, \\ X_{ij} \leq l_i x_j + u_j x_i - l_i u_j = x_i, \\ X_{ij} \leq u_i x_j + l_j x_i - u_i l_j = x_j. \end{array} \right\}.$$

A projection of the generated semidefinite cuts together with the McCormick bounds as in [39, 101] can lead to an equivalent projected formulation. However, we choose the intuitive results and easy integration of an extended formulation with any classes of cuts over any potential speedup a projection might bring in this case.

The cutting plane approach can be strengthened with any other class of valid extended inequalities to obtain tighter bounds. Based on [18], we append in our experiments from Section 5 the triangle inequalities [91], i.e.

$$\triangle := \left\{ (x, X) \middle| \ \forall i, j, k \in V : \begin{array}{r} X_{ij} + X_{jk} + X_{ki} - x_i - x_j - x_k \geq -1 \\ -X_{ij} + X_{jk} - X_{ki} + x_i \geq 0, \\ -X_{ij} - X_{jk} + X_{ki} + x_j \geq 0, \\ X_{ij} - X_{jk} - X_{ki} + x_k \geq 0 \end{array} \right\},$$

and in Section 5.4 we also use a class of SOCP constraints to augment the base $\mathcal{M}$ relaxation, i.e.

$$\mathcal{M}^2 := \mathcal{M} \cap \left\{ (x, X) \middle| \ \forall i \in V : \quad X_{ii} \geq x_i^2 \ \Leftrightarrow \ \left\| \begin{matrix} 1 - X_{ii} \\ 2x_i \end{matrix} \right\|_2 \leq 1 + X_{ii} \right\}.$$

Section 6 introduces quadratically constrained programs as an extension of QP, i.e.

$$z_{qc} = \min_x \left\{ x^T Q^{(0)} x + c_0^T x \;\middle|\; \begin{array}{ll} x^T Q^{(k)} x + c_k^T x \leq 0 & k = 1, \dots, m \\ Ax \leq b, \; x \in [0,1]^N \end{array} \right\}, \tag{QCQP}$$

with an N-variable vector x, $A \in \mathbb{R}^{p \times N}$ and $Q^{(k)} \in \mathbb{R}^{N \times N}$ $\forall k = 0, \dots, m$ assumed to be symmetric (potentially indefinite) matrices. This class of problems encompasses any instances bounded by closed domains by the same reasoning as in the case of QP. The extended space lifting to X variables and subsequent relaxations $\mathcal{S}, \mathcal{M}, \triangle$ from QP apply to QCQP as well. We define edge sets E_0 and E_m to capture QCQP structure in the objective-only and all the quadratics, respectively, i.e.,

$$\begin{aligned} E_0 &= \{\{i,j\} \in V \times V \mid i > j, \; Q_{ij}^{(0)} \neq 0\}, \\ E_m &= \{\{i,j\} \in V \times V \mid i > j, \; \exists k \in 0, \dots, m \text{ s.t. } Q_{ij}^{(k)} \neq 0\}, \quad E_0 \subseteq E_m, \end{aligned}$$

assuming no disconnected vertex in graph $G(V, E_m)$ for the same reasons as for $G(V, E)$.

3 Low-dimensional cutting planes from the semidefinite relaxation

This section develops effective outer-approximations for the quadratic semidefinite relaxation, retaining (most of) its tightness in the form of sparse (low-dimensional) linear cuts well-suited to Branch&Cut. Section 3.1 introduces low-dimensional semidefinite relaxations, Section 3.2 develops the theoretical background for outer-approximation via matrix completion and optimality-based cutting planes, while Section 3.3 describes alternative cut selection strategies and introduces an estimator for optimality-based selection that is then developed in Section 4.

3.1 Low-dimensional semidefinite relaxations via (sparse) vertex covers

To obtain lower-dimensional SDP relaxations starting from $\mathcal{S}$, let $\mathcal{V}$ represent any cover of the vertex set V, i.e. $V = \bigcup_{\rho \in \mathcal{V}} \rho$, as an overlapping decomposition [12]. For an arbitrary vertex/index subset $\rho \in \mathcal{V}$, denote:

- $x_\rho \in \mathbb{R}^{|\rho|}$ as the vector slice of x, i.e. for $\rho = (i_1, \dots, i_{|\rho|})$, $x_\rho = (x_{i_1}, \dots, x_{i_{|\rho|}})$;
- $X_\rho \in \mathbb{R}^{|\rho| \times |\rho|}$ as the submatrix slice of X.

We introduce the semidefinite relaxation corresponding to $\mathcal{V}$ consisting of semidefinite constraints of submatrices sliced on all $\mathcal{V}$ elements, e.g.

$$\mathcal{S}(\mathcal{V}) := \left\{ (x, X) \;\middle|\; \forall \rho \in \mathcal{V} : \; \begin{bmatrix} 1 & x_\rho^T \\ x_\rho & X_\rho \end{bmatrix} \succeq 0, \; X_{ii} \leq x_i \; \forall i \in \rho \right\}.$$

Remark 3.1.1 *(n-dimensional semidefinite constraint)*
For any $\rho \subseteq V$, $|\rho| = n$, the constraint $\begin{bmatrix} 1 & x_\rho^T \\ x_\rho & X_\rho \end{bmatrix} \succeq 0$ has $n(n+3)/2$ distinct variables in the lifted/extended space (x, X). However, we refer to the constraint as n-dimensional to reflect the size of the set ρ of vertices from the sparsity pattern graph determining the constraint. □

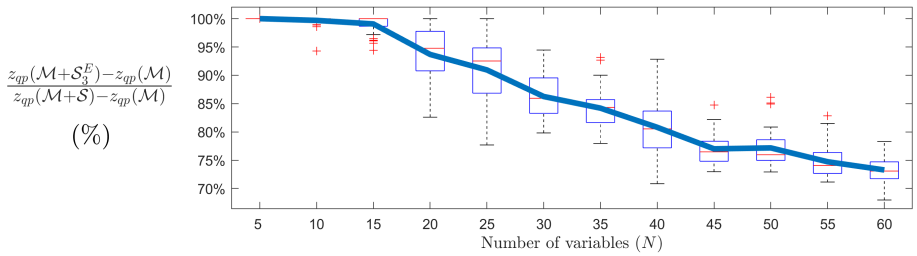


Fig. 1: The blue line shows, for different numbers of variables N (iterates of 5), the average gap percentage between $\mathcal{M}$ and $\mathcal{M}+\mathcal{S}$ bounds closed by $\mathcal{M}+\mathcal{S}_3^E$ on 30 dense ($E^+ = \overline{E} = E$) QP random instances. For each N, the box captures the 25th-75th percentiles, the central red line indicates the median, the whiskers capture non-outliers outside the box and the '+' marks indicate outliers. Each QP instance is sampled via random Q as explained in Table 1, with c assumed 0 and $Ax \leq b$ dropped.

We focus on finding a sparse cover $\mathcal{V}$ with few low-dimensional elements while balancing the tightness of $\mathcal{S}(\mathcal{V})$. A naïve cover set is given by the power set of V, denoted by $\mathcal{P}$, where $\mathcal{S}(\mathcal{P}) = \mathcal{S}$ involves an exponential number $|\mathcal{P}| = 2^N$ of (potentially dominated) semidefinite constraints. This motivates restricting/imposing a chosen cardinality n ($1 < n \leq N$) on elements of $\mathcal{P}$. For any edge set E^* such that $E \subseteq E^* \subseteq E^+$, let $C(V, E^*)$ denote the set of all $G(V, E^*)$ cliques of size at least 2. Depending on E^*, define for any $n \in 2, \ldots, N$,

$$\mathcal{P}_n^{E^*} := \{\rho \in C(V, E^*) \mid \ 2 \leq |\rho| \leq n, \ \nexists \rho_2 \in C(V, E^*) \text{ s.t. } |\rho_2| \leq n, \ \rho \subset \rho_2\} \text{ and } \mathcal{S}_n^{E^*} := \mathcal{S}(\mathcal{P}_n^{E^*}),$$

to represent a generic cover set with cardinality up to n and its associated n-dimensional semidefinite relaxation, respectively.

We explore $\mathcal{S}_n^{E^*}$, $z_{qp}(\mathcal{S}_n^{E^*})$ properties for $E^* \in \{E^+, \overline{E}, E\}$. Choosing $E^* = E^+$ leads to cover

$$\mathcal{P}_n^{E^+} = \{\rho \in \mathcal{P} \mid |\rho| = n\} \text{ with } |\mathcal{P}_n^{E^+}| = \binom{N}{n},$$

with $\mathcal{S}_n^{E^+}$ relaxing $\mathcal{S}$ in exchange for a polynomial number of n-dimensional PSD constraints. Furthermore, any n-dimensional semidefinite relaxation can directly exploit any sparsity pattern in QP instances. Chordal sparsity, exploited by sparse semidefinite solvers such as SDPA-C [43, 44, 87] to exactly decompose $\mathcal{S}$, can also make the $\mathcal{S}_n^{E^+}$ relaxation more lightweight and effective with no loss in bound strength.

Lemma 3.1.2 *(Cover $\mathcal{P}_n^{\overline{E}}$ based on chordal decomposition)*
$\mathcal{S}_n^{\overline{E}} \supseteq \mathcal{S}_n^{E^+}$ and $z_{qp}(\mathcal{S}_n^{\overline{E}}) = z_{qp}(\mathcal{S}_n^{E^+})$. If n is not smaller than the clique number of $G(V, \overline{E})$, then $z_{qp}(\mathcal{S}_n^{\overline{E}}) = z_{qp}(\mathcal{S})$.

Proof $\mathcal{S}_n^{\overline{E}} \supseteq \mathcal{S}_n^{E^+}$ follows immediately, since a positive semidefinite matrix implies any of its principal minors are positive semidefinite. Let the maximal cliques of $G(V, \overline{E})$ be $C_1, \ldots, C_k \in C(V, \overline{E})$ and w.l.o.g. $\cup_{i \in 1,\ldots,k} C_i = V$. The result in [44, th. 2.5] based on [50, th. 7] implies,

$$z_{qp}(\{X \succeq 0\}) = z_{qp}(\{X_{C_i} \succeq 0 \ \forall i \in 1, \ldots, k\}). \tag{1}$$

Each x variable in the QP original space appears in $\mathcal{S}$ because $\forall i \in V$, $X_{ii} \leq x_i$ is required to bound $\mathcal{S}$. To incorporate all x variables into the sparsity pattern, create an extra vertex v and

let $V' = V \cup \{v\}$ and $\overline{E}' = \overline{E} \cup \{(v,i)\ \forall i \in V\}$. Then $G(V', \overline{E}')$ is by definition chordal with maximal cliques $C_i \cup \{v\}\ \forall i \in 1,\dots,k$ and Eq. (1) becomes,

$$z_{qp}\left(\left\{\begin{bmatrix}1 & x^T\\ x & X\end{bmatrix} \succeq 0\right\}\right) = z_{qp}\left(\left\{\begin{bmatrix}1 & x_{C_i}^T\\ x_{C_i} & X_{C_i}\end{bmatrix} \succeq 0\ \forall i \in 1,\dots,k\right\}\right)$$
$$\Rightarrow z_{qp}(\mathcal{S}) = z_{qp}(\mathcal{S}(\{C_1,\dots,C_k\})), \tag{2}$$

where the last implication follows by adding $X_{ii} \le x_i\ \forall i \in V$ to the feasible sets on both sides.

To get $z_{qp}(\mathcal{S}_n^{\overline{E}}) = z_{qp}(\mathcal{S}_n^{E^+})$, since $\mathcal{S}_n^{\overline{E}} \supseteq \mathcal{S}_n^{E^+}$ implies $z_{qp}(\mathcal{S}_n^{\overline{E}}) \le z_{qp}(\mathcal{S}_n^{E^+})$, we try to prove $z_{qp}(\mathcal{S}_n^{\overline{E}}) \ge z_{qp}(\mathcal{S}_n^{E^+})$ by focusing on elements $\rho \in \mathcal{P}_n^{E^+} \setminus \mathcal{P}_n^{\overline{E}}$. It is well-known that any subgraph of a chordal graph is itself chordal.Thus, any $\rho \in \mathcal{P}_n^{E^+} \setminus \mathcal{P}_n^{\overline{E}}$ induces a chordal subgraph $G(\rho, \overline{E}_\rho)$ of $G(V, \overline{E})$, where $\overline{E}_\rho = \{\{i,j\} \in \overline{E} \mid i,j \in \rho\}$. Since the $G(\rho, \overline{E}_\rho)$ maximal cliques, denoted by $C_1^\rho,\dots,C_r^\rho$, are all subsets of $C_1,\dots,C_k$ with $|C_i^\rho| \le n$ then $\mathcal{S}(\{C_1^\rho,\dots,C_r^\rho\}) \supseteq \mathcal{S}(\mathcal{P}_n^{\overline{E}})$. This leads to

$$z_{qp}\left(\mathcal{S}(\mathcal{P}_n^{\overline{E}} \cup \{\rho\})\right) = z_{qp}\left(\mathcal{S}_n^{\overline{E}} \cap \mathcal{S}(\{\rho\})\right) \overset{(2)}{=} z_{qp}\left(\mathcal{S}_n^{\overline{E}} \cap \mathcal{S}(\{C_1^\rho,\dots,C_r^\rho\})\right)$$
$$= z_{qp}\left(\mathcal{S}(\mathcal{P}_n^{\overline{E}} \cup \{C_1^\rho,\dots,C_r^\rho\})\right) = z_{qp}(\mathcal{S}_n^{\overline{E}}).$$

The above result shows that no element $\rho \in \mathcal{P}_n^{E^+} \setminus \mathcal{P}_n^{\overline{E}}$ tightens the bound $z_{qp}(\mathcal{S}_n^{\overline{E}})$, so $z_{qp}(\mathcal{S}_n^{\overline{E}}) \ge z_{qp}(\mathcal{S}_n^{E^+})$, and consequently $z_{qp}(\mathcal{S}_n^{E^+}) = z_{qp}(\mathcal{S}_n^{\overline{E}})$. Additionally, Eq. (2) implies that if $n \ge \max_{1\le i\le k} |C_i|$ (clique number), then all maximal cliques are contained in $\mathcal{P}_n^{\overline{E}}$ and the bound of the original SDP relaxation is achieved, i.e. $z_{qp}(\mathcal{S}_n^{E^+}) = z_{qp}(\mathcal{S}_n^{\overline{E}}) = z_{qp}(\mathcal{S})$. □

Computing the cover $\mathcal{P}_n^{\overline{E}}$ in Lemma 3.1.2 depends on constructing $G(V, \overline{E})$. It is practically possible via heuristic sparse matrix ordering packages (as in SDPA-C) such as Chompack [3]. Since $E \subseteq \overline{E}$, unless $G(V,E)$ is chordal, the relaxation $\mathcal{S}_n^{\overline{E}}$ uses lifted variables X_{ij} with zero coefficients, i.e. $Q_{ij} = 0$, meaning they do not appear directly in the objective and thus contribute to relaxation bounds only indirectly. Intuitively, the farther $G(V,E)$ is from chordal, the more lifted variables with zero coefficients are used in $\mathcal{S}_n^{\overline{E}}$, which make the relaxation heavier for less direct bound improvement than non-zero coefficient variables. To capture this latter trade-off, we relax $\mathcal{S}_n^{\overline{E}}$ further into $\mathcal{S}_n^{E}$ to avoid lifted variables with zero coefficients.

Corollary 3.1.3 *(Cover $\mathcal{P}_n^E$ as an objective-based restriction of chordal decomposition)* $\mathcal{S}_n^E \supseteq \mathcal{S}_n^{\overline{E}}\ (\supseteq \mathcal{S}_n^{E^+})$ and $z_{qp}(\mathcal{S}_n^E) \le z_{qp}(\mathcal{S}_n^{\overline{E}})\ (= z_{qp}(\mathcal{S}_n^{E^+}))$ with equalities for chordal $G(V,E)$. □

The next remarks discuss the choice of dimensionality n, solving n-dimensional SDP relaxations, and their connections with other related conic and linear relaxations from the literature.

Remark 3.1.4 *(Low-dim. SDP relaxation hierarchy - choosing n for tractability & strong bounds)* Any $E^* \subseteq E^+$, e.g. E^+, $\overline{E}$, E, implies

$$\mathcal{S}_2^{E^*} \supseteq \mathcal{S}_3^{E^*} \supseteq \cdots \supseteq \mathcal{S}_n^{E^*} \supseteq \cdots \supseteq \mathcal{S}_N^{E^*} \supseteq \mathcal{S} \quad \text{(PSD matrix} \Rightarrow \text{PSD principal minors)}.$$

Relaxations $\mathcal{S}_n^{E^*}$ get tighter with increasing n, but a low n is both tractable and can capture strong bounds:

- More tractable $\mathcal{S}_n^{E^*}$ are obtained for small n values, in terms of (i) number of constraints and (ii) constraint dimensionality. The binomial coefficient $\binom{N}{n} = |\mathcal{P}_n^{E^+}| \ge |\mathcal{P}_n^{E}|, |\mathcal{P}_n^{\overline{E}}|$, bounding the number of constraints, is a concave symmetric function of $n \in 2,\dots,N-1$ and lowest around extremes, i.e. $n \in \{2, N-1\}$. But small n additionally limits constraint dimensionality.

– Relaxations $\mathcal{S}_n^{E^*}$ can lead to strong bounds even for small n values. For example, consider $n = 3$ with fully dense QP instances, e.g. $E^+ = \overline{E} = E$, of size $N \leq 60$ and no linear constraints $Ax \leq b$. Starting with a base McCormick $\mathcal{M}$ relaxation, Fig. 1 shows that $\mathcal{M}+\mathcal{S}_3^{E^+}$ closes a significant proportion of the gap between $\mathcal{M}$ and $\mathcal{M} + \mathcal{S}$.

□

Remark 3.1.5 *($\mathcal{S}_n^{E^*}$ - linear outer-approximation versus interior point methods)*
The number of semidefinite constraints scales polynomially with the number N of variables ($\binom{N}{n}$ constraints for $E^* = E^+$ or fully dense instances), creating a heavy semidefinite formulation even for small n. Prior work finds that using an interior point solver combined with linear cuts, e.g. the widely used $\mathcal{M}$, leads to impractical computational times [18, 94, 100, 101]. As explained in Section 1, linear formulations are well-suited to the Branch&Cut framework and exploit a large body of existing work. Thus, we aim to construct strong linear outer-approximations rather than solving relaxations $\mathcal{S}_n^{E^*}$ $\forall n \in 2, \ldots, N$ using interior point methods. □

Remark 3.1.6 *($\mathcal{S}_n^{E^*}$ extends SOCP relaxations & sparsifies SDP outer-approximations)*
• Considering Remark 3.1.4, for increasing $n \geq 2$, by design the relaxations $\{\mathcal{S}_n^{E^*}\}$ dominate and progressively extend an equivalent SOCP Type 2 relaxation [63, 64] which contains SOCP constraints for variable matrices of dimension 2.
• Other semidefinite outer-approximations consider high-dimensional convex cutting surfaces in all (x, X) variables [39, 100, 101]. Prior work separates linear outer-approximations using a high-dimension [94] or medium-dimension [104, 105] feasibility approach. Convex nonlinear cutting surfaces are less suitable to a Branch&Cut framework than the established (mixed-integer) linear relaxations which combine advanced cut classes, heuristics, preprocessing and easy warm-starting for reduced solution times at the tree nodes. But high-dimensional linear cuts lead to a computationally heavy LP [94]. Since $\binom{N}{n}$ is concave in n, separating medium-dimensional linear cuts incurs the most separation subproblems. This paper proposes a low-dimensional approach leading to lighter linear relaxations suitable for Branch&Cut. These cuts can be separated first via optimality-based selection (see Section 5). □

Remark 3.1.7 *($\mathcal{S}_n^{E^*}$ complements multiterm linear relaxations)*
The table below summarizes the main differences between outer-approximating the $\mathcal{S}_n^{E^*}$ relaxation and multiterm linear relaxations [10, 12, 18, 78, 82] for quadratic programming:

	Low-dim. semidefinite cuts from $\mathcal{S}_n^{E^*}$	*Multiterm linear relaxations*
Approach	Decompose SDP constraints to select (separate) low-dimensional cuts iteratively	Decompose multiterm expressions to explicitly identify low-dimensional dominant facets/cuts
Expressions captured & bound strength	All possible quadratic expressions (including mixtures of convex and vertex-polyhedral terms), which can improve bounds overall	Only vertex-polyhedral [78, 82] or multilinear expressions having a polyhedral convex envelope [10, 18], restricting bound strength
Cutting plane selection & generation	Outer-approximation cuts selected & generated based on the current LP solution	Explicit facets only selected (if not all added) but not generated based on the current LP solution

Both a $\mathcal{S}_n^E/\mathcal{S}_n^{\overline{E}}$ outer-approximation and multiterm linear relaxations lead to lower-dimensional cutting planes via a decomposition based on problem sparsity. Similar to $\mathcal{S}_n^{E^*}$ (see Remark 3.1.4), the number of facets involved by multiterm relaxations scales exponentially with dimensionality, so these cuts are most common for low n, e.g. $n \in \{3, 4, 5\}$ for vertex-polyhedral cuts.

However, for fixed n, compared to $\mathcal{S}_n^{E^*}$, multiterm linear relaxations specialize only to expressions allowing explicit facets of a polyhedral convex envelope. Multiterm relaxations generate strong facet-defining cuts (independent of the LP solution) at the cost of restricted bound strength since they cannot bound all quadratic expressions. A cutting plane outer-approximation of $\mathcal{S}_n^{E^*}$ may improve multiterm bounds by capturing any quadratic expression, i.e. including convex terms and zero-coefficient terms. But, since we generate new outer-approximating cuts based on the current LP solution at every cut round, careful cut selection becomes important.

A hybrid complementary approach can use multiterm relaxations for relevant expressions and tighten bounds further via cuts selected from $\mathcal{S}_n^{E^*}$ outer-approximations. Section 5.4 tests this idea computationally. □

The Remark 3.1.5 observations motivate the problem of selecting which low-dimensional semidefinite constraints to include in linearized form. The next subsections explore this idea for any relaxation $\mathcal{S}(\mathcal{V})$ by inferring linear cutting planes from semidefinite constraints using an optimality-based scoring function estimator.

3.2 From semidefinite relaxations to optimality-based linear cuts

First, we study more generally outer-approximating a PSD condition with optimality-based linear cuts in a specific direction on a partially-fixed matrix in the presence of other convex constraints. Effective QP specialization relies on the properties emerging from Lemma 3.2.3: (i) optimality-based cuts in the objective direction, (ii) tradeoffs of fixing patterns and (iii) the effect on the directional outer-approximation of more cuts/other cut classes.

Definition 3.2.1 *[57] (Partial symmetric matrix of variables via a fixing)*
Any (symmetric) matrix of variables $M \in \mathbb{R}^{N\times N}$ is made *partial* by partitioning its entries, according to a (symmetric) pattern/mapping, into two complementary variable subsets represented by vectors v, v_C and *fixing* v to a given value.

Definition 3.2.2 *(Projection of a convex set based on a fixing)*
For any fixing $v = \tilde{v} \in \mathbb{R}^{dim(v)}$ associated with a partial M and any convex non-empty set $\mathcal{B}$ in the space of M variables, denote the projection of $\mathcal{B}$ onto the complementary vector v_C by $\mathcal{B}^{\tilde{v}} = \text{proj}_{v_C}\, \mathcal{B} = \mathcal{B} \cap \{(v, v_C) \mid v = \tilde{v}\}$.

Lemma 3.2.3 *(Optimality-based outer-approximation of any PSD matrix completion)*
For any fixed variables v associated with partial M, with $\mathcal{S}$ and any convex set $\mathcal{B}$ both in the space of M variables, assume v is fixed only to values $\tilde{v}$ s.t. $(\mathcal{B} \cap \mathcal{S})^{\tilde{v}} \neq \emptyset$, i.e. M has PSD completion at fixing $\tilde{v}$. Then:

(i) Given a fixing $v = \tilde{v}$, the PSD completion of M can be fully outer-approximated via *linear inequalities* of the form,

$$\mathcal{S}^{\tilde{v}} = \left\{ v_C \mid \forall q \in \mathbb{R}^{|C|} :\ q \cdot v_C \geq q \cdot v_C^* = \min_{v_C \in \mathcal{S}^{\tilde{v}}} \{q \cdot v_C\} \right\}.$$

(ii) Given a fixed $q \in \mathbb{R}^{|C|}$, the outer-approximation of the convex set $(\mathcal{B} \cap \mathcal{S})$ *based on optimality only in the direction of* q (see Fig. 2),

$$(\mathcal{B} \cap \mathcal{S}) \subseteq \mathcal{B} \cap \left\{ M \mid \forall \tilde{v} :\ q \cdot v_C \geq q \cdot v_C^* = \min_{v_C \in \mathcal{S}^{\tilde{v}}} \{q \cdot v_C\} \right\},$$

becomes more restrictive with (a) additional restrictions on $\mathcal{B}$ or (b) a larger fixing size $\dim(v)$.

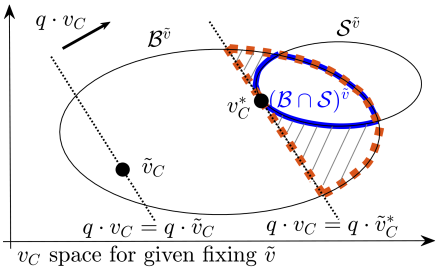


Fig. 2: Illustration of Lemma 3.2.3 convex sets for a fixing $v = \tilde{v}$. The set delimited by the dashed contour outer-approximates $(\mathcal{B} \cap \mathcal{S})^{\tilde{v}}$ by restricting $\mathcal{B}^{\tilde{v}}$ via cutting plane $q \cdot v_C \geq q \cdot v_C^*$ based on fixing a q direction. The hashed region shows q may be a strong direction to cut off set $(\mathcal{B}^{\tilde{v}} \setminus \mathcal{S}^{\tilde{v}})$, but it may not eliminate it entirely.

Proof See Appendix A.1. □

We specialize Lemma 3.2.3 to any low-dimensional SDP relaxation of QP based on a cover $\mathcal{V}$, to get optimality-based cuts only in the QP objective direction.

- $f(X) := Q \bullet X$ and $f(X_\rho) := Q_\rho \bullet X_\rho$ with $Q_\rho \in \mathbb{R}^{|\rho| \times |\rho|}$ the submatrix slice of Q on ρ.
- Given a fixing $x = \tilde{x}$, the ρ subproblem

$$f(X_\rho^*|\tilde{x}_\rho) := \min_{X_\rho} \left\{ Q_\rho \bullet X_\rho \,\middle|\, \begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & X_\rho \end{bmatrix} \succeq 0,\ X_{ii} \leq \tilde{x}_i \ \forall i \in \rho \right\} = \min_{X_\rho} \left\{ f(X_\rho) \middle| \mathcal{S}^{\tilde{x}}(\{\rho\}) \right\}. \tag{$P^{\tilde{x}}(\rho)$}$$

We hereafter refer to $P^{\tilde{x}}(\rho)$ as a ρ *subproblem* since picking index set ρ defines its feasible region.

Corollary 3.2.4 *(Low-dim. SDP relaxations bounds via optimality-based cuts and fixing x)* Given a QP relaxation with base convex set $\mathcal{B}$ and a cover $\mathcal{V}$ associated with a low-dimensional SDP relaxation, then for any fixing $x = \tilde{x}$ s.t. $\tilde{x} \in [0,1]^N,\ A\tilde{x} \leq b,\ \mathcal{B}^{\tilde{x}} \neq \emptyset$,

$$z_{qp}(\mathcal{B} \cap \mathcal{S}^{\tilde{x}}(\mathcal{V})) \geq z_{qp}^L(\mathcal{B} \cap \mathcal{S}^{\tilde{x}}(\mathcal{V})) := \underbrace{\min_{X \in \mathcal{B}^{\tilde{x}}} \{ f(X) \mid \overbrace{\forall \rho \in \mathcal{V}:\ f(X_\rho) \geq f(X_\rho^*|\tilde{x}_\rho)}^{\text{optimality-based cuts}} \}}_{\text{main problem } P^{\tilde{x}}} + c^T \tilde{x},$$

and $z_{qp}(\mathcal{B} \cap \mathcal{S}(\mathcal{V})) \geq z_{qp}^L(\mathcal{B} \cap \mathcal{S}(\mathcal{V}))$.

Proof See Appendix A.2. □

In the context of LP-based Branch&Cut, one practical fixing $x = \tilde{x}$ for Corollary 3.2.4 is the current LP relaxation solution obtained after a cut round. We hereafter refer to $P^{\tilde{x}}$ from Corollary 3.2.4 as the *main problem*, since $P^{\tilde{x}}$ aggregates the objective structure of all subproblems $P^{\tilde{x}}(\rho)$ to bound the full QP objective.

Remark 3.2.5 *(Fixing x to the current LP solution $\tilde{x}$ for optimality-based outer-approximation)* Fixing $x = \tilde{x}$ has the following implications on the optimality-based outer-approximations derived in Corollary 3.2.4:

- Guarantees an unconditional matrix PSD completion at all values $\tilde{x}$ for any lifted QP relaxation, thus ensuring the validity of each linear inequality $(\forall \rho \in \mathcal{V})\ f(X_\rho) \geq f(X_\rho^*|\tilde{x}_\rho)$.
- Strengthens the optimality-based outer-approximation. Lemma 3.2.3(ii) shows that fixing x captures stronger lower bounds than unfixed x (which leads to only one possible optimality-based cut per $\rho \in \mathcal{V}$ subproblem). More generally, across all possible fixing values, a larger fixing size implies tighter optimality-based outer-approximation. Despite better bounds, other augmented fixing patterns do not guarantee a PSD completion at all values, requiring specialized graph-based PSD completion patterns, e.g. [13, 32, 57, 60].

- The independent ρ subproblems $P^{\tilde{x}}(\rho)$ each satisfy coupling constraints $Ax \leq b$.

□

Corollary 3.2.4 linearizes semidefinite constraints (at each $x = \tilde{x}$ fixing) based only on the objective structure/optimality and solutions of ρ subproblems in the low dimensional (sparse) variable space X_ρ. Consequently, the next subsection shows that, despite a weaker final bound z^L_{qp}, the $P^{\tilde{x}}$ relaxed formulation can be used for an optimality-based cut selection measure. We next discuss separation/selection measures of cutting planes from $\mathcal{S}^E_n$, with the optimality approach enabled by $P^{\tilde{x}}$ requiring a parametric estimator (built in Section 4 for small $n \in \{2, 3, 4, 5\}$ using neural networks).

3.3 Cutting plane generation and selection (via estimator)

3.3.1 Cut generation via eigenvalues

Linear cutting planes need to be generated to outer-approximate, $\forall \rho \in \mathcal{V}$, the semidefinite constraint $\mathcal{S}\{(\rho)\}$ by cutting off the current LP solution point $(\tilde{x}_\rho, \tilde{X}_\rho)$.

Following the literature, we cut off $(\tilde{x}_\rho, \tilde{X}_\rho)$ by a hyperplane based on the largest negative eigenvalue [94, 104], defined as

$$\lambda_{\min}(\rho) = \min(\Lambda), \text{ with } \Lambda \text{ the eigenvalues of } \begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & \tilde{X}_\rho \end{bmatrix}. \qquad (\lambda_{\min}(\rho))$$

If $\lambda_{\min}(\rho) < 0$ and $v(\rho)$ is the corresponding eigenvector, generate and add the violated cut,

$$(\lambda_{\min}(\rho) =) \qquad v^T(\rho) \begin{bmatrix} 1 & x_\rho^T \\ x_\rho & X_\rho \end{bmatrix} v(\rho) \geq 0. \qquad (\text{EigCut}(\rho))$$

These eigenvalue cuts are well-known, but they are typically high-dimensional [104]. Rather than sparsifying high-dimensional cuts (see, e.g. [94]), our new contribution allows choosing sparse but effective cuts by combining selection strategies based on cut violation, i.e. feasibility measure and expected objective improvement, i.e. optimality measure.

As an alternative to eigenvalue cuts, tangents to the SDP underestimator surface $\{\forall \tilde{x}_\rho f(X^*_\rho | \tilde{x}_\rho)\}$ can be generated, e.g. Fig. 3 for $|\rho| = n = 2$: (i) a tangent hyperplane at $f(X^*_\rho | \tilde{x}_\rho)$ (shown in transparent red); (ii) the tangent hyperplane at the closest $f(X^*_\rho, x_\rho)$ point via projection. However, generating such tangent cuts requires numerically solving an SDP projection subproblem with interior point solvers, a computational overhead we avoid. Also note that any inexact $f(X^*_\rho | \tilde{x}_\rho)$ estimation can lead to invalid cutting planes.

3.3.2 Cut selection via different measures (including optimality-based)

For a cover $\mathcal{V}$ and the current LP solution point $(\tilde{x}_\rho, \tilde{X}_\rho)$, cut selection involves choosing/ranking (by a measure) which $\rho \in \mathcal{V}$ to generate eigenvalue cut EigCut(ρ) for. Similar to the idea behind *deep objective cuts* [23], we use the objective information to sort the possible low-dimensional cutting planes. Given a $\rho \in \mathcal{V}$ and solution point $(\tilde{x}_\rho, \tilde{X}_\rho)$, Corollary 3.2.4 implies,

$$\underbrace{\left(\begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & \tilde{X}_\rho \end{bmatrix} \succeq 0 \Leftrightarrow P^{\tilde{x}}(\rho) \text{ feasible at } \tilde{X}_\rho \right)}_{\text{feasibility}} \rightarrow \underbrace{\left(f(\tilde{X}_\rho) \geq f(X^*_\rho | \tilde{x}_\rho) \right)}_{\text{optimality}}. \qquad (3)$$

The negation/violation of either side of implication (3) at the current solution $(\tilde{x}_\rho, \tilde{X}_\rho)$ leads to a violated EigCut(ρ) for ρ, which can be ranked and selected to generate based on the measures:

- **Feasibility measure.** The absolute value of the largest negative eigenvalue, $-\lambda_{\min}(\rho)$, found at the current point $(\tilde{x}_\rho, \tilde{X}_\rho)$, represents the violation of the ρ-sliced semidefinite constraint $\begin{bmatrix} 1 & x_\rho^T \\ x_\rho & X_\rho \end{bmatrix} \succeq 0$. The measure is an Euclidian distance in the variable space (x, X) and does not incorporate or correlate explicitly with the QP objective. Feasibility cut selection may not yield the best objective function improvement, but it does identify violated cuts. Prior work uses these negative eigenvalues for individual cut selection [94, 104].
- **Optimality measure.** The expressions:

$$f(X_\rho^*|\tilde{x}_\rho) - f(\tilde{X}_\rho) \approx \qquad (\mathcal{I}_X(\rho))$$
$$\hat{f}_n^*(Q_\rho, \tilde{x}_\rho) - f(\tilde{X}_\rho) = \hat{f}_n^*(Q_\rho, \tilde{x}_\rho) - Q_\rho \bullet \tilde{X}_\rho, \qquad (\hat{\mathcal{I}}_X(\rho))$$

  with estimator $\hat{f}_n^*(Q_\rho, \tilde{x}_\rho)$ defined in Eq. (4), represent both:
  (i) The *estimated objective improvement* on X_ρ variables possible at $\tilde{x}_\rho$ by cutting off the infeasible variable solution $\tilde{X}_\rho$ and its associated objective $f(\tilde{X}_\rho)$ (see Figure 3). By transposing implication (3), a positive $\mathcal{I}_X(\rho)$ corresponds to $P^{\tilde{x}}(\rho)$ infeasible at $\tilde{X}_\rho$. This ensures a cut removing the solution point $(\tilde{x}_\rho, \tilde{X}_\rho)$ exists, with the $\mathcal{I}_X(\rho)$ magnitude indicating the objective improvement possible by adding the cut. For given n, selecting a limited number of ρ subproblems with the most positive $\mathcal{I}_X(\rho)$ for cut generation would likely result in the greatest objective improvement over the entire problem $P^{\tilde{x}}$.
  (ii) The *violation of an optimality-based cut for the main problem* $P^{\tilde{x}}$ at $(\tilde{x}_\rho, \tilde{X}_\rho)$.
- **Combined measure** (optimality and feasibility). A measure combining $\hat{\mathcal{I}}_X(\rho)$ and $\lambda_{\min}(\rho)$ based on their respective advantages and weaknesses is introduced in Definition 5.2.1.

Optimality measures provide an avenue to select cuts likely to improve the objective the most. But for instance when $\mathcal{V} = \mathcal{P}_n^{E^+}$, finding X_ρ^* or $f(X_\rho^*|\tilde{x}_\rho)$ $\forall \rho \in \mathcal{P}_n^{E^+}$ requires solving $\binom{N}{n}$ semidefinite-constrained subproblems $P^{\tilde{x}}(\rho)$, which is prohibitive via algorithmic methods such as interior point when N is large and/or $n \geq 2$, contravening the Section 1 desiderata (2) of inexpensive cut selection.

Alternatively, we introduce a ***fast estimator*** $\hat{f}_n^*(Q_\rho, \tilde{x}_\rho)$ (computed in Section 4 for small n) parametric explicitly only on inputs Q_ρ and $\tilde{x}_\rho$, with X_ρ^* implicit, given by,

$$(\forall \rho \in \mathcal{V},\ |\rho| = n) \quad \hat{f}_n^*(Q_\rho, \tilde{x}_\rho) \ \approx \ f(X_\rho^*|\tilde{x}_\rho) \qquad (4)$$

The estimator $\hat{f}_n^*(Q_\rho, \tilde{x}_\rho)$ approximates $\mathcal{I}_X(\rho)$ via $\hat{\mathcal{I}}_X(\rho)$ for any given ρ within a given error to provide a quick deterministic measure for selecting ρ subproblems $P^{\tilde{x}}(\rho)$ and their associated eigenvalue cuts. Therefore, the neural network estimators developed in Section 4 take the complexity of cut selection offline.

4 Estimating subproblem solutions via trained neural networks

This section builds estimators for $\hat{f}_n^*$ $(n \leq 5)$ and, by extension, for optimality measure $\hat{\mathcal{I}}_X(\rho)$ to select the optimality-based Section 3.3 cuts. We learn $\hat{f}_n^*$ in a supervised manner with the goal of taking offline the computational complexity of optimality-based cut selection.

The data for the supervised learning task relies on inputs $\tilde{x}_\rho, Q_\rho$ and output $f(X_\rho^*|\tilde{x}_\rho)$. The regression problem is nonlinear, consisting of all points $\tilde{x}_\rho$ on a collection of convex surfaces, each

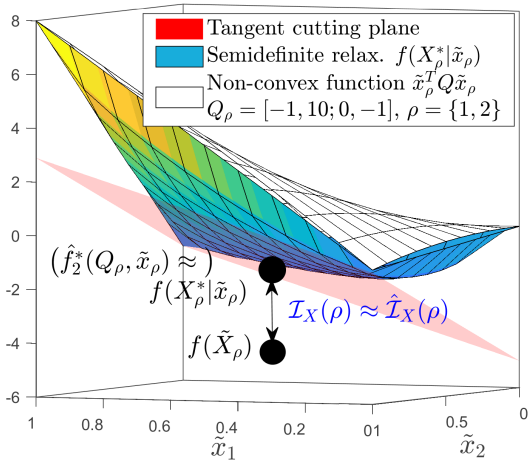


Fig. 3: Illustration, given $|\rho| = 2$, of the semidefinite relaxation (colored) that underestimates a quadratic nonconvex function (white) across $\tilde{x}_\rho \in [0, 1]^2$.

For a given point $(\tilde{x}_\rho, \tilde{X}_\rho)$ with objective $f(\tilde{X}_\rho)$ the $\mathcal{I}_X(\rho) \approx \hat{\mathcal{I}}_X(\rho)$ maximal objective improvement on X_ρ variables possible at $\tilde{x}_\rho$ is found vertically by optimization only over X_ρ variables till touching the semidefinite underestimator at the point $f(X^*_\rho|\tilde{x}_\rho)$.

uniquely determined by Q_ρ. The solution X^*_ρ associated with output $f(X^*_\rho|\tilde{x}_\rho)$ acts as a latent variable. To mitigate potential model bias, we do not explicitly specify a model but rather adapt a flexible model based on data and the latent nonlinear features in that data. Abundant data can be sampled randomly as explained in Section 4.1 and scaled appropriately for training a model. Therefore, deep neural networks provide the appropriate learner: nonlinear features are implicitly learned in hidden layers and low variance and bias are expected given sufficient data and model complexity. Section 4.2 illustrates the engineering choices of neural network architectures and their training.

4.1 Data - sampling and scaling

A trained (supervised) model is only as good as its training data. First, to generalize the trained model, the input-output data sampled/generated for training needs to be uniformly distributed in the important input features. Second, the domain scaling of the sampled training data needs to enable model learning (in this case neural network learning, see Section 4.2). Third, any input evaluated by a trained estimator needs scaling to meet the training data assumptions.

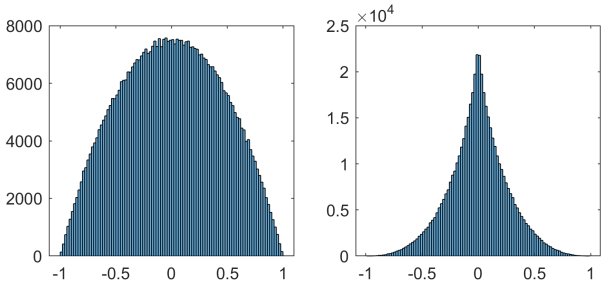


Fig. 4: Histograms, given $|\rho| = 3$, of any on-diagonal (left) and off-diagonal (right) elements for 0.5 million samples of Q_ρ generated through the eigen-decomposition shown in Table 1.

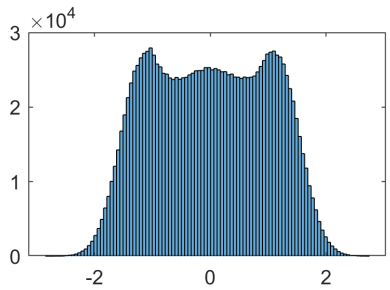


Fig. 5: Histogram, given $|\rho| = 3$, of eigenvalues for 0.5 million samples of Q_ρ generated via matrix entries uniformly distributed in $[0, 1]$.

Table 1: Sampling of inputs used as data to train neural networks for estimating $\hat{\mathcal{I}}_X(\rho)$ for a given ρ.

Inputs	Features to learn from inputs	(Input) Sampling uniformly on features	
		Sampled variables	Sample distributions & domains
$\tilde{x}_\rho$	Point positioning across a convex surface	Point $\tilde{x}_\rho$	$U([0,1]^{\lvert\rho\rvert})$
Q_ρ $(= V(\Lambda I_{\lvert\rho\rvert})V^T)$	Positive/negative semidefiniteness (shape/curvature of convex surface)	Eigenvalue vector Λ	$U([-1,1]^{\lvert\rho\rvert})$
		Orthonormal basis/matrix V	$U(O^{\lvert\rho\rvert\times\lvert\rho\rvert})$ (uniform on orthogonal group)

For sampling, the literature on positive definite and correlation matrices focuses on eigenvalues (rather than matrix entries) as the most important feature [58, 84]. Eigenvalues are also the most important features for sampling indefinite matrix Q_ρ, to obtain samples of inputs $\tilde{x}_\rho$, Q_ρ and outputs $f(X_\rho^*|\tilde{x}_\rho)$ for every relevant ρ:

- ***Input sampling.*** Table 1 shows how independent inputs $\tilde{x}_\rho$ and Q_ρ are sampled in order to distribute the training data uniformly over features. Input $\tilde{x}_\rho$ localizes a point on a convex surface (the convex shell in Fig. 3) and can thus be directly sampled. However, we do not sample the symmetric coefficient matrix Q_ρ uniformly over its elements, even though it determines the properties (curvature) of a convex surface (again, the convex shell in Fig. 3). Fig. 5 shows sampling matrix Q_ρ entries directly results in an unwanted non-uniform eigenvalue distribution. The extensive literature on sampling positive definite [84] and correlation matrices [58] also avoids sampling Q_ρ entries directly. Instead, an easy extension to sampling negative definite matrices can be done if sampling is based on a random spectrum [58] by simply allowing negative eigenvalues. We adopt this sampling technique for any Q_ρ via random eigenvalues and orthonormal bases (sampled using the SciPy [89] function *scipy.stats.ortho_group* implementing [76]) as shown in Table 1. Fig. 4 shows that sampling Q_ρ via its eigen-decomposition does not lead to uniform distributions of Q_ρ elements, which conversely further validates not sampling uniform Q_ρ elements directly.
- ***Output sampling.*** For each $\tilde{x}_\rho$, Q_ρ sampled as described above, $f(X_\rho^*|\tilde{x}_\rho)$ is the exact solution of problem $P^{\tilde{x}}(\rho)$ computed using interior point solver Mosek [86] with default settings.

Appropriate data scaling for training or evaluating a supervised model is essential. Neural networks have a complex architecture of nodes, but the operation at each node, the *activation function*, is relatively simple, e.g. the sigmoid hyperbolic tangent *tanh* or the piecewise-linear *ReLU*. But these activation functions have limited domains of applicability, i.e. ranges where the function has non-zero gradient (see Fig. 6). The training process relies on varying function gradients, so we scale the domain/image ranges to keep the gradients numerically far from 0:

- ***Scaling of training data.*** Table 1 limits sampled eigenvalues Λ to the range $[-1,1]$. This domain choice is driven by (i) the Λ features and (ii) the Λ-dependent domain of Q_ρ. The domain and sign of the Λ elements determines the shape/nonconvexity that Q_ρ induces in the QP objective, i.e. $x_\rho^T Q_\rho x_\rho$. Thus, Λ impacts the distance from $x_\rho^T Q_\rho x_\rho$ to its semidefinite relaxation in the subproblem $P^{\tilde{x}}(\rho)$ with solution $f(X_\rho^*|\tilde{x}_\rho)$, and by extension impacts the magnitude of $\hat{f}^*$. Consequently, eigenvalue signs and their relative magnitudes directly impact $\hat{f}_n^*$ as relevant features and must be maintained. However, since symmetric (sigmoid-like) activation functions for neural networks have non-zero gradients only over limited domains, the domain of Q_ρ elements (as inputs to activation functions) must be bounded as well. A scaled down eigenvalue domain of $[-1,1]$ retains all relevant features and ensures the domain

of Q_ρ elements is also $[-1,1]$-bounded (see Lemma 4.1.1). The $[-1,1]$-bounded Q_ρ elements and $[0,1]$-bounded $\tilde{x}_\rho$ (via the QP formulation) are therefore inputs a sigmoid-like activation function can learn over (see Section 4.2).

- ***Re-scaling of inputs for evaluation.*** The trained model should evaluate estimated solutions for any subproblems $P^{\tilde{x}}(\rho)$ from any QP instance. For the inputs associated with a subproblem $P^{\tilde{x}}(\rho)$ $\forall\rho$ to meet the training data assumptions, they may need re-scaling, i.e. both Q_ρ elements and eigenvalues Λ are re-scaled to $[-1,1]$. Dividing the input Q_ρ by a constant scaling factor appropriately bounds the Q_ρ elements and their implicit eigenvalues Λ. The model output obtained using the scaled Q_ρ can then be multiplied by the scaling factor to estimate the original Q_ρ input. This scaling is possible because $\hat{f}_n^*(Q_\rho, \tilde{x}_\rho) \approx f(X_\rho^*|\tilde{x}_\rho)$ is linear in the Q_ρ input (see $P^{\tilde{x}}(\rho)$). Lemma 4.1.2 finds a Q_ρ scaling factor that bounds implicit Λ within $[-1,1]$, but special cases and more elaborate proofs may enable tighter bounds with a smaller scaling factor, see e.g. [47]. The QP definition guarantees that $\tilde{x}_\rho$ does *not* need re-scaling, and more general QP with arbitrary bounds can also be re-scaled to $[0,1]$ as discussed in Section 2.

Lemma 4.1.1 *(eigenvalue bounds → matrix bounds)*
If all eigenvalues of $M \in \mathbb{R}^{n\times n}$ are in $[-m,m]$ then any element in M is in $[-m,m]$.

Proof Let eigen-pairs of M be $(\lambda_i,\ v_i)$ for $\forall i \in 1,\ldots,n$, and let v_{ij} be the j-th element of eigenvector v_i. Then $|M_{ij}|$ satisfies,

$$|M_{ij}| = \left|\sum_{k=1}^{n} v_{ki}v_{jk}\lambda_k\right| \leq \sum_{k=1}^{n}|v_{ki}v_{jk}|\cdot|\lambda_k| \leq \sum_{k=1}^{n}((v_{ki}^2+v_{jk}^2)/2)\cdot|\lambda_k| \leq \sum_{k=1}^{n}((v_{ki}^2+v_{jk}^2)/2)m = m$$

□

Lemma 4.1.2 *(matrix bounds → eigenvalue bounds) [47], Theorem A1.*
Any eigenvalue λ of $M \in \mathbb{R}^{n\times n}$ satisfies $|\lambda| \leq n\cdot\max_{i,j}|M_{ij}|$. Equivalently, the eigenvalues of the rescaled matrix $\left(1/(n\cdot\max_{i,j}|M_{ij}|)\right)M$ are all bounded by $[-1,1]$. □

4.2 Neural network training and evaluation

Motivated by Section 3.1, we learn an estimate solution $\hat{f}_n^*$ of problem $P^{\tilde{x}}(\rho)$ for limited small $n \in \{2,3,4,5\}$, as warranted by Remark 3.1.4 and Fig. 1. We use sufficient randomly generated data (following Section 4.1) to train a neural network model. The training and architecture of neural nets require several engineering choices [49] (see Appendix A.3). The sequel further expands on activation functions in the input/hidden layers, which are an important modeling choice closely linked to sampling the training data.

Two activation classes for neural network regression are sigmoid-based functions and variants of rectified linear units (*ReLU*) [48, 71]. Of the sigmoid-based functions, the hyperbolic tangent *tanh* is typically fastest and easiest to train [66]. Further, the *tanh* activation (see Fig. 6) matches our data assumptions around learning $P^{\tilde{x}}(\rho)$ solutions:

- **The *tanh* gradient is symmetric around** 0, matching the symmetric, 0-centred domains of the data generated according to Section 4.1.
- **The *tanh* gradient is concave with maximum attained at** 0**.** For all ρ, $Q_\rho \in [-1,1]^{n\times n}$ values close to 0 and their sign are crucial to solving $P^{\tilde{x}}(\rho)$ because a sign change near 0 may provoke a jump in the solution. Since *tanh* has a high smooth gradient around 0, inputs Q_ρ close to 0 will be well-learned.

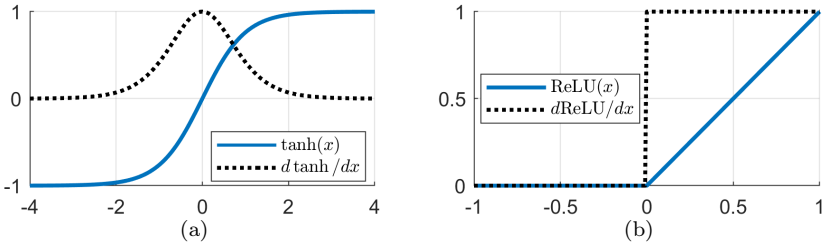


Fig. 6: Functions *tanh* and *ReLU* with their gradients.

- **The *tanh* gradient is significantly positive on the domain $[-2, 2]$ with a bounded *tanh* output range of $[-1, 1]$.** By being well inside the relevant *tanh* domain and co-domain, the bounded inputs $Q_\rho \in [-1, 1]^{n\times n}, \tilde{x}_\rho \in [0, 1]^n$ mitigate saturation, i.e. the activation function does not squeeze the inputs to 0 or 1 and thereby lose important features. Therefore, the common sigmoid activation problem of vanishing gradients [93] is likely to be avoided, as the Fig. 7 computational results show on test data.

Alternatively, we could use simple *ReLU* for learning $\hat{f}_n^*$ as a collection of smooth convex surfaces that convey curvature. However, due to non-smooth 0 gradients for *ReLU* input below 0, this approach likely requires deeper, sparser networks [48]. This increased architecture involves potentially more testing and finding an efficient regularization, issues we seek to avoid. Another possibility to keep architectures dense is a leaky *ReLU* activation [71]. Given the good generalization obtained with *tanh* in Fig. 7, the only distinctive improvements correctly engineered *ReLU* architectures can bring are training speed (less significant for small $n \leq 5$) and faster (trained) neural net evaluation. So *ReLU* activations could potentially gain computational performance at evaluation time for any n and keep training time manageable for increased n ($\gg 5$), both outside the scope of our analysis.

Alongside *tanh* activation, we employ the scaled conjugate gradient (SCG) learning algorithm [85]. SCG is typically used in moderate-sized neural network architectures [30, 98, 99], e.g. those in Fig. 7. Via approximate second-order information, SCG can further prevent *tanh* from under-fitting due to vanishing gradients and achieve faster convergence than either first-order backpropagation or the more accurate Levenberg-Marquardt [53, 83, 102].

Figure 7 illustrates, on independent test data, the good generalization of neural networks trained using our proposed methodology. Observe in Fig. 7 on a test set of appropriately scaled inputs: (i) tight fits, (ii) residuals $\mathcal{I}_X(\rho)$–$\hat{\mathcal{I}}_X(\rho)$ that are normally distributed with no major skew and centered around 0, and (iii) scatter plots of residuals versus fits lacking any patterns. Low residuals coupled with stable results on a large test set imply the trained neural networks offers both low bias and variance. There is a slight deterioration in the fits and residuals variance (as evidenced by the 95% confidence lines) when increasing n from 2 to 5, which is expected when learning progressively more complex models. The $n = 5$ neural network can also be used for $n = \{2, 3, 4\}$, but Fig. 7(d) shows better approximations are obtained for neural networks trained with smaller n. Thus, under our data sampling and model training choices, Fig. 7 shows that neural networks can successfully learn the solution space of ρ subproblems $P^{\tilde{x}}(\rho)$ via $\hat{f}_n^*$, implicitly estimating the optimality measure $\hat{\mathcal{I}}_X(\rho)$.

Table 2 analyzes the average speed of evaluating the trained neural networks tested in Fig. 7. We obtain speedups of 1-2 orders of magnitude with neural nets estimates $\hat{\mathcal{I}}_X(\rho)$ versus computing $\mathcal{I}_X(\rho)$ with interior point solvers for tolerances similar to the neural net errors found in Fig. 7. Moreover, evaluating neural networks is slightly faster than computing $\lambda_{\min}(\rho)$. Thus, the neural network estimator brings the overhead of optimality cut selection at least on par with

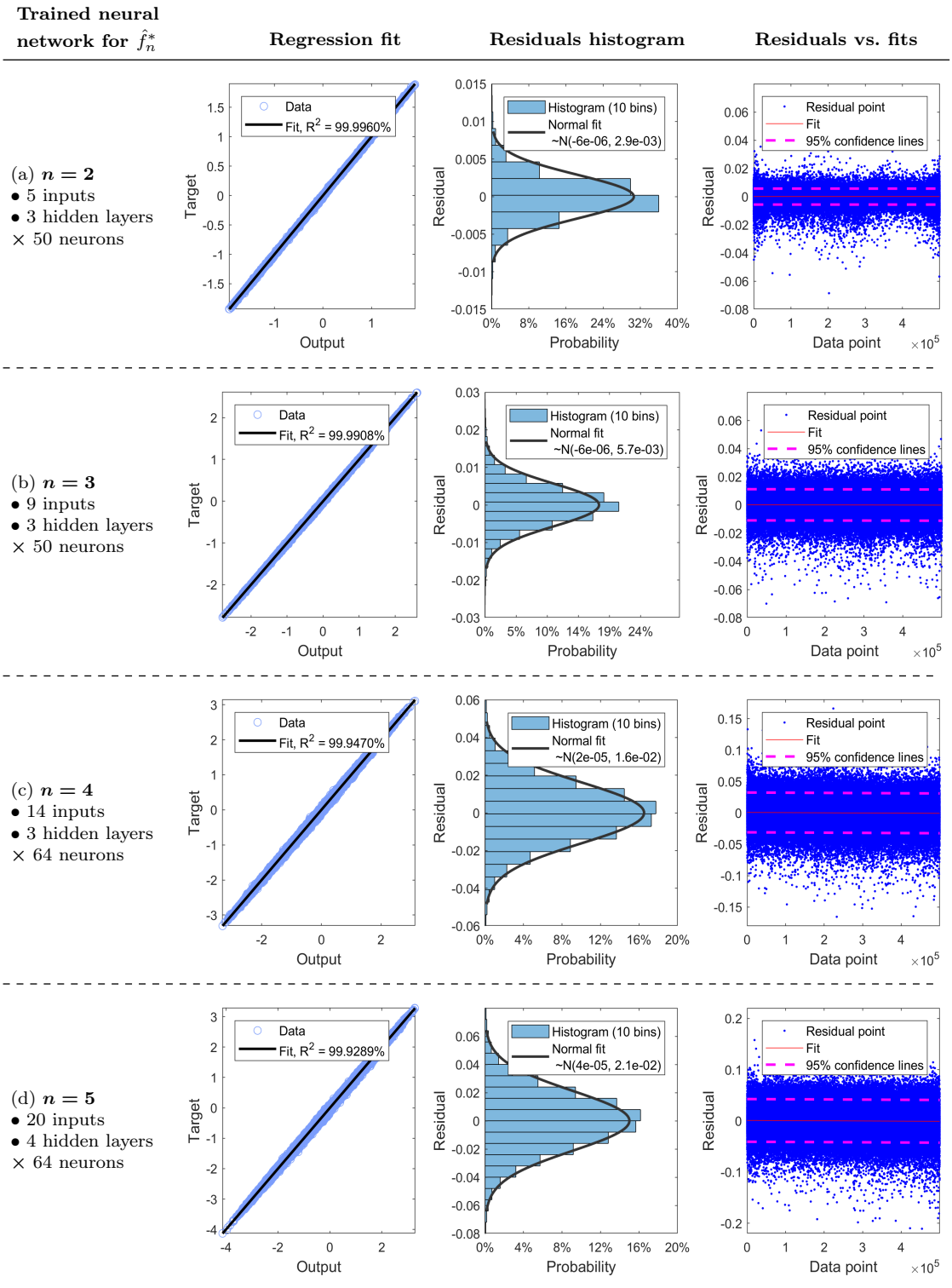


Fig. 7: Performance of trained neural networks for $\hat{f}_n^*, \forall n \in \{2,3,4,5\}$. For each n (and the associated network topology), the performance of the trained estimator in terms of fit and residuals is shown on a test set of 0.5 million data points (not used in training/validation). Each data point is generated as in Table 1 and as such, has appropriately scaled inputs for evaluation. The inputs are $\tilde{x}_\rho, Q_\rho$ where $|\rho| = n$ and Q_ρ is symmetric. Therefore, the number of inputs are $n + \frac{1}{2}(n+1)n$.

Table 2: Speed comparison of evaluating cut selection measures in Section 3.3.2 : (NN) evaluation time for estimated $\hat{\mathcal{I}}_X(\rho)$ via the trained neural nets; (Eigendecomp.) evaluation time for $\lambda_{\min}(\rho)$ via the Lapack [4] *_syevd* eigendecomposition routine; solver times for getting the exact solution $\hat{\mathcal{I}}_X(\rho)$ of $P^{\bar{x}}(\rho)$ by setting all documented tolerances to 0.1/0.01 for: (Mosek) Mosek 8.1 [86]; (SeDuMi) SeDuMi 1.3 [109]. All times reported are averaged across 10000 evaluations of randomly generated inputs and obtained on the hardware/software setup mentioned in the next section.

n	Average times (ms)						Speedups of NN vs.				
	NN	Eigdecomp.	Mosek		SeDuMi		Eigdecomp.	Mosek		SeDuMi	
			tol. 1e-2	tol. 1e-1	tol. 1e-2	tol. 1e-1		tol. 1e-2	tol. 1e-1	tol. 1e-2	tol. 1e-1
2	0.031	0.058	0.96	0.63	5.63	4.65	1.9x	31.3x	20.4x	183.1x	151.2x
3	0.031	0.063	1.23	0.75	5.81	5.33	2.1x	40.2x	24.5x	190.4x	174.7x
4	0.031	0.064	1.33	0.82	6.9	5.87	2.1x	42.7x	26.3x	221.5x	188.4x
5	0.046	0.068	1.43	0.93	7.16	5.75	1.5x	31x	20.3x	155.6x	125.1x

feasibility cut selection. In contrast, the larger overhead of interior point solvers is inadequate for thousands of ρ subproblem evaluations, e.g. Table 7, and disproportionate versus the feasibility cut selection time or the relaxation solve time, e.g. Table 6 and Fig. 10.

The subsequent sections use the specific neural networks trained for $n = \{2, 3, 4, 5\}$ to get the estimated optimality measure $\hat{\mathcal{I}}_X(\rho)$.

5 Optimality-based semidefinite cut selection for QP problems

This section implements and analyzes the optimality-based selection of cutting planes discussed in Sections 3.2–3.3 for outer-approximating the low-dimension semidefinite relaxation of QP problems. Algorithm 1 sets the framework to outer-approximate $\mathcal{B} + \mathcal{S}(\mathcal{V})$, given any $\mathcal{B}$ linear base relaxation and $\mathcal{V} \subseteq \mathcal{P}_n^{E^+}$ for small $n \leq 5$. We compare and analyze cut selection based on different orderings/selections of $\rho \in \mathcal{V} \subseteq \mathcal{P}_n^{E^+}$ subproblems. Testing on all 99 BoxQP instances [27, 29, 111], grouped as in Table 4, we analyze optimality-based, i.e. $\hat{\mathcal{I}}_X(\rho)$-based, cut selection in different setups. The results, summarized in Table 3, illustrate the Corollary 3.2.4 implications and, more importantly, show the Section 1 desiderata is met.

First, Section 5.1 shows that the highly-scalable, deterministic $\hat{\mathcal{I}}_X(\rho)$ evaluated via the Section 4 neural network estimators adequately substitutes $\mathcal{I}_X(\rho)$ in terms of optimality selection convergence. Section 5.2 then compares the convergence properties of selecting, within the set of cuts EigCut(ρ), the feasibility versus optimality versus random cuts. The optimality selection shows stronger early convergence but weaker bounds compared to feasibility selection. As a result, a new combined ordering measure selects cuts based on both optimality and feasibility, showing both strong convergence and strong final bounds. Section 5.3 first analyzes the impact of cut sparsity/dimensionality on cut selection, proposing a heuristic based on instance structure, followed by a discussion on the tradeoffs of incorporating chordal extensions. Finally, Section 5.4 shows even tight/competitive QP relaxations can be cheaply improved via low-dimensional semidefinite cuts, meeting the Section 1 desiderata. All results can be reproduced via the source code available on Github [9]. This manuscript only presents (i) limited, individual examples and (ii) summary statistics aggregated with respect to Table 4. But Github [9] also shows full results for each instance. The results shown throughout the paper are computed on a single thread on an AWS instance [77] with 2.3 GHz Intel Broadwell E5-2686v4 vCPUs and 16 GiB RAM memory.

Table 3: Summary of cut selection/ordering based on different measures, computational needs (algorithmic vs. deterministic), and convergence and bounds obtained employing them. The *Max bound* column illustrates the implications of Corollary 3.2.4.

Ordering/ selection of cuts/ subproblems $\rho \in \mathcal{V} \subseteq \mathcal{P}_n^{E^+}$		Measure evaluation (max #evals $\leq \binom{N}{n}$ for $\mathcal{V} \subseteq \mathcal{P}_n^{E^+}$)		Cuts selected [a]	Convergence (+good, −bad)		Max bound [b]
Type	Measure	Type	Algorithmic/ Deterministic [c]		First iters	Last iters	
Feasibility	$-\lambda_{\min}(\rho)$	Eigenvalue decomp.	Algo.	feas.-based	−	+	$= z_{qp}$
Optimality	$\mathcal{I}_X(\rho)$ $\hat{\mathcal{I}}_X(\rho)$	SDP sol. (interior pt. [86]) Estimator eval. *(neural net, see Section 4)*	Algo. Det.	opt.-based	+	−	$\geq z_{qp}^L$ $\leq z_{qp}$
Combined (feas.+opt.)	$\mathcal{C}(\rho)$ *(see Section 5.2)*	Eigenvalue decomp. & Estimator eval.	Algo.+ Det.	opt.- & feas.-based	+	+	$= z_{qp}$
Random	n/a	Random number	Det.	random	−	−	$= z_{qp}$

[a] The *optimality-based* cuts are a subset of the *feasibility-based* cuts.

[b] The bounds z_{qp}, z_{qp}^L are shorthand for $z_{qp}(\mathcal{B} + \mathcal{S}(\mathcal{V}))$, $z_{qp}^L(\mathcal{B} + \mathcal{S}(\mathcal{V}))$, respectively, assuming any convex base relaxation $\mathcal{B}$.

[c] *Algorithmic* measures rely on an algorithm with several steps, whereas *deterministic* evaluation is a closed-form expression.

Algorithm 1: Iterative SDP outer-approximation with cut selection based on an ordering

Input: - current base LP relaxation $\mathcal{B}$ of QP, either fully added from the start, i.e. $\mathcal{M}$, or (independently) separated iteratively at each cut round ($\triangle$, $0-1/2$, edge-concave);
- decomposed SDP relax. $\mathcal{S}(\mathcal{V})$ to outer-approx., where $\mathcal{V} \subseteq \mathcal{P}_n^{E^+}$ with small n, i.e. $\mathcal{P}_n^{\overline{E}}$, $\mathcal{P}_n^E$ etc.;
- current LP solution $(\tilde{x}, \tilde{X})$;
- selection strategy/ordering metric $M(\rho)\ \forall \rho \in \mathcal{V}$ at $(\tilde{x}, \tilde{X})$ e.g. $\hat{\mathcal{I}}_X(\rho)$, $-\lambda_{\min}(\rho)$, $\mathcal{C}(\rho)$ etc.;
- selection size*, i.e. a fixed % of $|\mathcal{V}|$ (upper capped at 5000);
- number of cut rounds R (set by default to 20);
- termination criteria - if active, terminate on an improvement between two consecutive cut rounds of $< 0.1\%$ of the gap closed overall so far from the $\mathcal{M}$ bound;

Output: Polyhedral outer-approximation that lower-bounds $z_{qp}(\mathcal{B} + \mathcal{S}(\mathcal{V}))$ and SDP relax. $z_{qp}(\mathcal{B} + \mathcal{S})$;

for *R cut rounds if termination criteria not met* **do**
  Sort $\mathcal{V}$ by descending $M(\rho)\ \forall \rho \in \mathcal{V}$ at current $(\tilde{x}, \tilde{X})$;
  for *top ρ subproblems in sorted $\mathcal{V}$ within selection size* **do**
    if $\lambda_{min}(\rho) < 0$ *(violated PSD condition for* $\begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & \tilde{X}_\rho \end{bmatrix}$*)* **then**
      $\mathcal{B} = \mathcal{B} \cap \{\text{new EigCut}(\rho) \text{ based on } \lambda_{\min}(\rho)\}$;
  Resolve (warm-start) new LP relaxation $\mathcal{B}$ that includes added cuts;
  Update current LP solution $(\tilde{x}, \tilde{X})$;
Last obtained $z_{qp}(\mathcal{B})$ lower bounds $z_{qp}(\mathcal{B} + \mathcal{S}(\mathcal{V}))$ and $z_{qp}(\mathcal{B} + \mathcal{S})$;

*For the combined and feasibility selection measures, the selected hyperplanes will always cut off the current LP solution and the number of cuts will match *selection size*. For the optimality and random selection measures, the number of cuts used depends on the number of the selected cuts violating the feasibility measure.

Table 4: All 99 BoxQP instances in [27, 29, 111], grouped by size and density [18].

Size	Low	Medium	Large	Jumbo	Density	Low	Medium	High
# vars	20-40	50-70	80-90	100-125	%	[0-40]	(40-60]	(60,100]

Throughout this section, we will refer to *feasibility-based* cuts, i.e. all cuts EigCut(ρ) violating the feasibility measure, versus *optimality-based* cuts, i.e. the cuts favored by our optimality measure. Our hypothesis, which we will motivate in the following sections, is that an optimization solver should automatically select the subset of *optimality-based* cuts within the set of *feasibility-based* cuts. Feasibility-based cut selection is known to converge slowly [94, 116]. But, as a consequence of Lemma 3.2.3(ii), naïve, greedy cut selection may not converge at all. We claim, and attempt to show in the sequel, that selecting the *optimality-based* cuts within the set of *feasibility-based* cuts is key to good performance.

5.1 Optimality cut selection - estimated versus exact measures

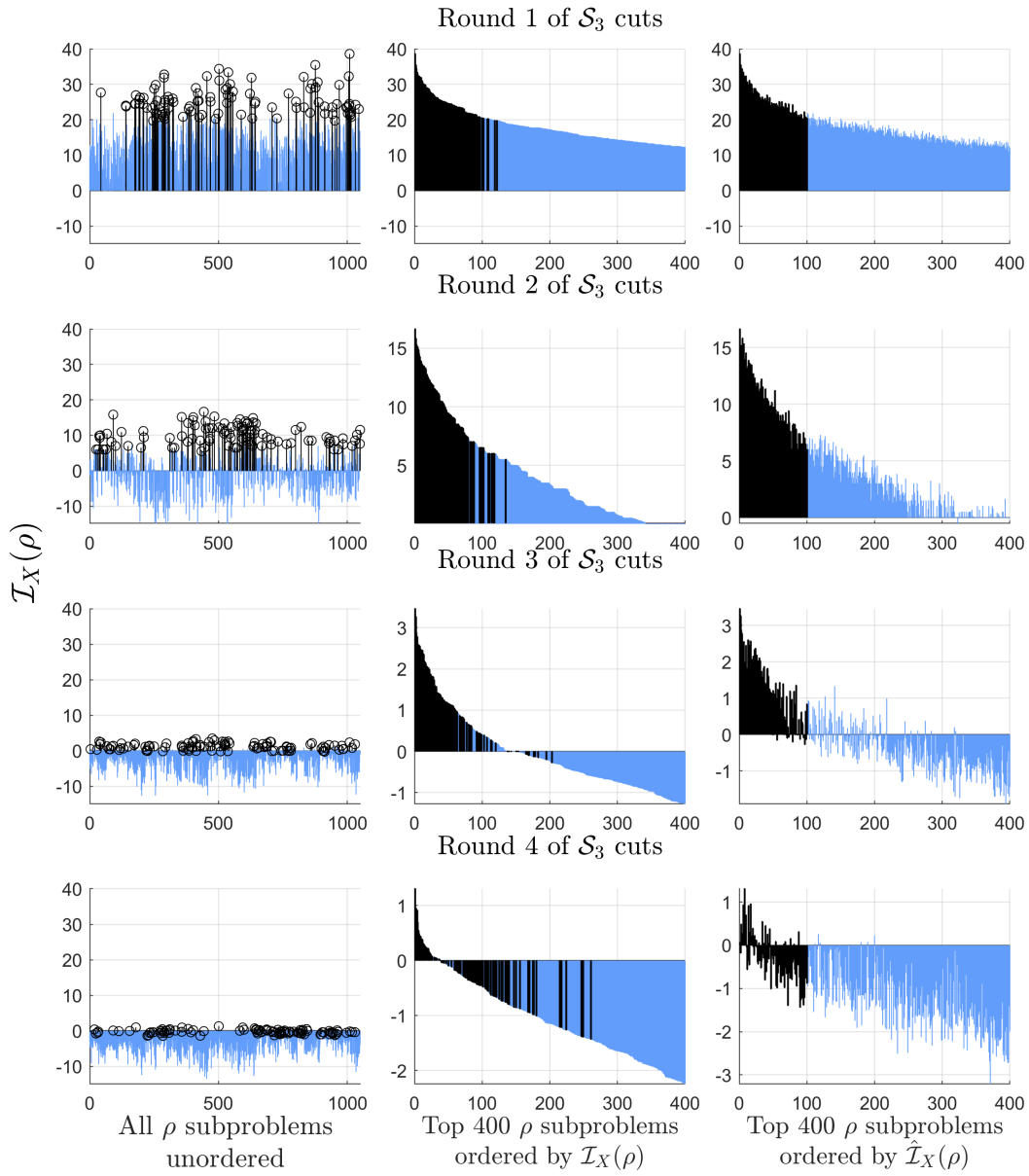
First, we analyze optimality-based cut selection by comparing the convergence properties of ordering on either the estimated $\hat{\mathcal{I}}_X(\rho)$ (deterministic evaluation of a trained neural net) or the exact $\mathcal{I}_X(\rho)$ (algorithmic interior point [86] solution). Fig. 8 illustrates on one empirical example points generally valid on any instance: (i) why bounds based on cut selection via $\mathcal{I}_X(\rho)$ can be replicated via an estimated $\hat{\mathcal{I}}_X(\rho)$; (ii) when and how the error of the estimator influences results.

Fig. 8 illustrates cut selection based on the $\hat{\mathcal{I}}_X(\rho)$ optimality measure to outer-approximate $\mathcal{M} + \mathcal{S}_3^E$ for a small dense BoxQP instance, where ρ subproblems $P^{\tilde{x}}(\rho)$ are selected for cut generation across 4 rounds. Across the rounds of cuts in Fig. 8, the three columns of subfigures plot $\mathcal{I}_X(\rho)$ against three types of ordering (at each round) the ρ subproblems:

- The first column leaves subproblems unordered in the same positions across the rounds. Figure 8, column 1 shows that the magnitude of the cuts selected (black) via $\tilde{\mathcal{I}}_X(\rho)$ at a round generally decreases at the next round. The positive expected objective improvements for all subproblems tend towards 0, i.e. $\mathcal{I}_X(\rho) \searrow 0$ $\forall \rho$ s.t. $\mathcal{I}_X(\rho) > 0$. In turn, this improves overall bounds towards the $\mathcal{M} + \mathcal{S}_3^E$ bound as shown in the Fig. 8 table. Note $\forall \rho$, a negative $\mathcal{I}_X(\rho)$ value in a later round with LP solution $(\tilde{x}, \tilde{X})$ means the objective of the ρ subproblem $P^{\tilde{x}}(\rho)$ is improved above its optimal value due to other cuts external to $P^{\tilde{x}}(\rho)$.
- The second column orders subproblems by decreasing $\mathcal{I}_X(\rho)$ each round. The more $\mathcal{I}_X(\rho)/\hat{\mathcal{I}}_X(\rho)$ discrepancies, the larger x-axis gaps in the black selected area.
- The third column orders subproblems by decreasing $\hat{\mathcal{I}}_X(\rho)$ each round. The more $\mathcal{I}_X(\rho)/\hat{\mathcal{I}}_X(\rho)$ discrepancies, the less ordered $\mathcal{I}_X(\rho)$ is across the $\hat{\mathcal{I}}_X(\rho)$-ordered x-axis.

Across cut rounds (subfigure rows), the $\mathcal{I}_X(\rho)/\hat{\mathcal{I}}_X(\rho)$ discrepancies become more pronounced towards rounds 3-4. Consider the Fig. 8 table fourth column: in the later rounds, the smaller standard deviation among the top $\mathcal{I}_X(\rho)$ values imply smaller differences among their estimates $\hat{\mathcal{I}}_X(\rho)$. Therefore, the selection/ordering of cuts based on estimated $\hat{\mathcal{I}}_X(\rho)$ becomes less accurate. For example, consider ρ_1, ρ_2: an (exact) ordering $\mathcal{I}_X(\rho_1) \gtrapprox \mathcal{I}_X(\rho_2)$ with sufficiently close values can be flipped via estimates $\tilde{\mathcal{I}}_X(\rho_1) \lessapprox \tilde{\mathcal{I}}_X(\rho_2)$ due to the neural net estimator error (see residual plots in Fig. 7). The same observation is supported by column 3 in the Fig. 8 table: in later rounds, a decreasing fraction of subproblems are selected by both the $\mathcal{I}_X(\rho)$ and $\hat{\mathcal{I}}_X(\rho)$ orderings.

More generally, we expect eventual differences in selection results between $\mathcal{I}_X(\rho)$ and $\hat{\mathcal{I}}_X(\rho)$ for any instance, choice of n, or choice of estimator model. After sufficient cut rounds improving the objective bound, any positive expected objective improvements $\mathcal{I}_X(\rho)$ $\forall \rho$ yielding violated



Round #	% of $\mathcal{M}$ to ($\mathcal{M}$+$\mathcal{S}_3^E$) gap closed	% Same selection (top 100) via $\mathcal{I}_X(\rho)$ vs. $\hat{\mathcal{I}}_X(\rho)$	Std. dev. of top 100 $\mathcal{I}_X(\rho)$ values
1	56.26	94	3.81
2	85.82	90	2.65
3	94.19	76	0.72
4	97.19	67	0.34

Fig. 8: Illustration (using Algorithm 1) on BoxQP dense instance *spar020-100-1* of optimality cut selection via neural net estimate $\hat{\mathcal{I}}_X(\rho)$ across 4 cut rounds, to converge from $z_{qp}(\mathcal{M})$ towards $z_{qp}(\mathcal{M}+\mathcal{S}_3^E)$.
For all subfigures, the y-axis represents $\mathcal{I}_X(\rho)$ (calculated using [86]) plotted as a function of subproblems $\rho \in \mathcal{P}_3^E$ on the x-axis. Each subfigure column orders the ρ subproblems on the x-axis in a different way to compare $\mathcal{I}_X(\rho)$ and $\hat{\mathcal{I}}_X(\rho)$. Each subfigure row corresponds to a round of maximum 100 cuts EigCut(ρ) added for 100 selected ρ subproblems $P^{\tilde{x}}(\rho)$. Selected subproblems at each cut round are marked by black lines in all subfigures.

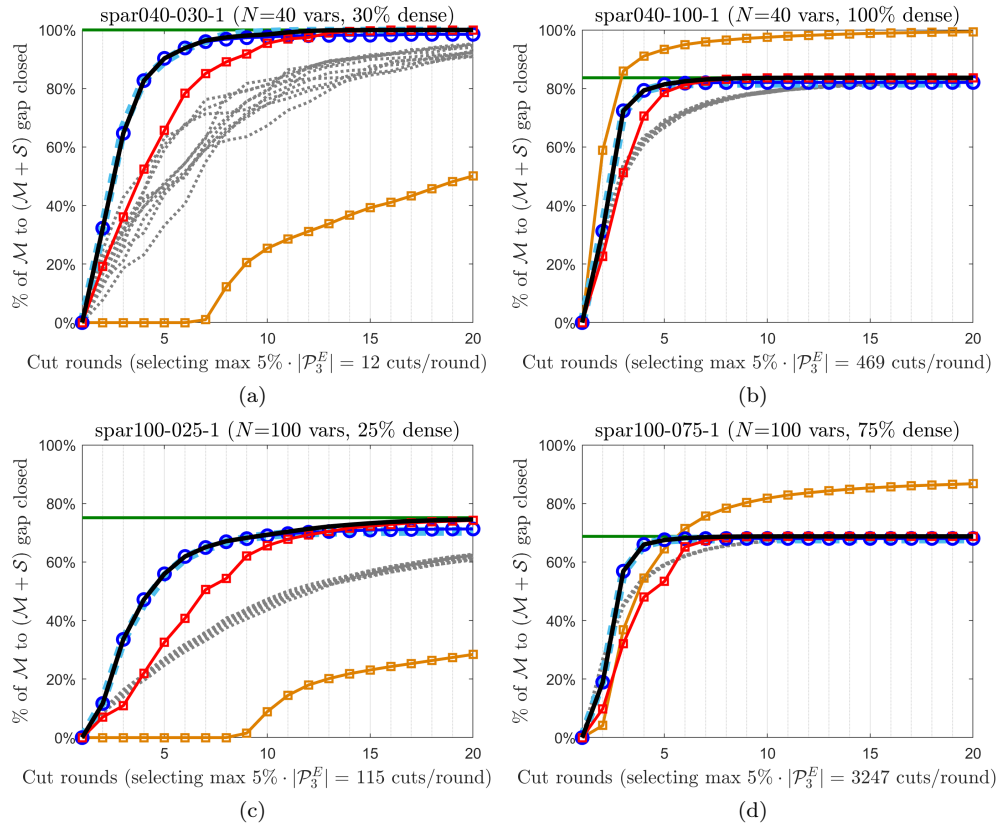


Fig. 9: Bounds comparison, across 20 cut rounds, between Table 3 low-dimension cut selection/ordering strategies to outer-approximate $\mathcal{M}+\mathcal{S}_3^E$ (green bound), for different size/density BoxQP instances. Algorithm 1 adds to $\mathcal{M}$ every round, given the rounds' $\mathcal{P}_3^E$ ordering, any violated low-dimension EigCut(ρ) for ρ within the top 5% · $|\mathcal{P}_3^E|$ selection. The comparison includes adding each round all violated full matrix (high-dim.) eigenvalue cuts to outer-approximate $\mathcal{M}+\mathcal{S}$.

cuts will be closer to 0. The closer to 0 these $\mathcal{I}_X(\rho)$ get, the smaller their differences and the more expected estimator error influences cut selection. Nonetheless, all positive $\mathcal{I}_X(\rho)$ close to 0 implies small expected objective/bound improvements, i.e. similar final bounds are reached both via $\hat{\mathcal{I}}_X(\rho)$ and $\mathcal{I}_X(\rho)$ before the effect of the expected estimator error. When positive $\mathcal{I}_X(\rho) \gg 0$ $\forall\rho$ are more spread out in values, $\hat{\mathcal{I}}_X(\rho)$ should mimic $\mathcal{I}_X(\rho)$ in cut selection.

These points mean well trained estimators are adequate models when used in practice for cut selection, as Fig. 9 illustrates for the neural network estimators built in Section 4 on BoxQP instances across varying sizes/densities. Thus, the fast, vectorizable, parametric computation of $\hat{\mathcal{I}}_X(\rho)$ is an adequate replacement for $\mathcal{I}_X(\rho)$, allowing the effective scaling needed for optimality selection to compete with feasibility selection.

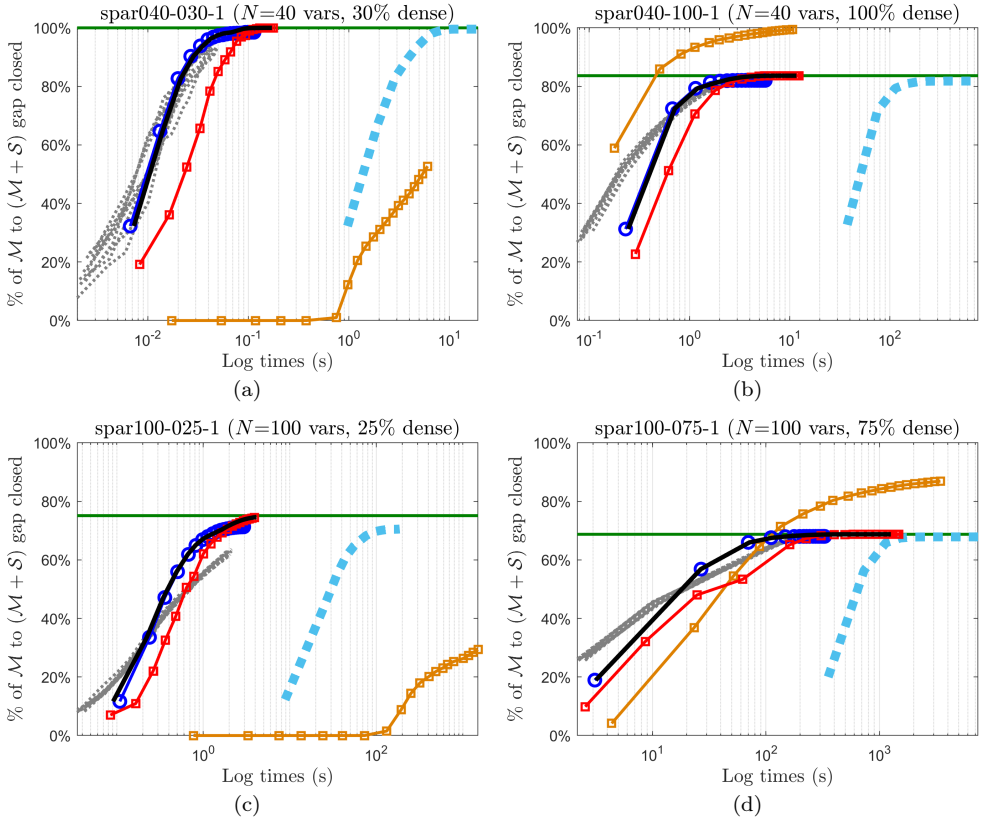


Fig. 10: Bounds comparison, across log time (seconds), between Table 3 low-dimensional cut selection/ordering strategies to outer-approximate $\mathcal{M}+\mathcal{S}_3^E$ (green bound) and adding each round all violated full matrix (high-dimensional) eigenvalue cuts to outer-approximate $\mathcal{M}+\mathcal{S}$. All runs are set up identical to Fig. 9, but visualized across log time.

5.2 Cut selection strategies - optimality versus feasibility versus random

This section compares, with respect to the % gap closed towards the $\mathcal{M}+\mathcal{S}$ relaxation bound across cut rounds, three selection strategies for outer-approximating $\mathcal{M}+\mathcal{S}_3^E$ via Algorithm 1. Figs. 9 and 10 analyze these selection strategies across iterations and time, respectively, for BoxQP instances of different sizes and densities. The results illustrate:

(o1) Low-dimensional SDP vertex cover (Section 3.1) bounds can be closely outer-approximated using fewer linear cuts with the optimality, rather than the feasibility selection measure. This fewer number of cuts, coupled with fast neural net evaluations, enables a computationally lighter relaxation (Fig. 10) as set out in the Section 1 desiderata, e.g. the $\mathcal{M}+\mathcal{S}_3^E$ relaxations in Fig. 1. Fully-dense instances (Fig. 9b, Fig. 10b) are the only exception, with fully dense (unselected) cuts more appropriate (see Table 6 discussion at the end of Section 5.2).

(o2) Optimality-based selection is strongest, especially in early cut rounds. These results match the discussion of deep objective cuts by Boyd and Vandenberghe [23].

(o3) Feasibility selection, which matches the $\mathcal{M} + \mathcal{S}_3^E$ bounds, exhibits stronger final bounds than optimality selection (Fig. 9c). The results are justified by Corollary 3.2.4 which proves that $z_{qp}(\mathcal{S}^{\bar{x}}(\mathcal{V}) \geq z_{qp}^L(\mathcal{S}^{\bar{x}}(\mathcal{V})$, i.e. the feasibility-based bound is tighter than the optimality-based bound. Notably, the $\mathcal{M} + \mathcal{S}_3^E$ bounds can be reached by random selection, but poor convergence leads to large numbers of cuts and/or cut rounds.

A trade-off thus exists between strong performance of early iterations for optimality selection versus slightly better (given enough cut rounds) final bounds for feasibility selection at further cost. Figs. 11 and 12 recast this trade-off into one between selecting fewer optimality-based (that are also feasibility-based) cuts and selecting all feasibility-based cuts, isolating these two competing goals for the Fig. 9c BoxQP instance:

- **Strong (feasibility-based) cuts are found by optimality selection** (Fig. 11).
  - Cut selection based on optimality measures (estimated or not) clearly leads to better bound (gap closed) improvements per cut than feasibility or random selection (with feasibility partially outperforming random). Moreover, Fig. 11b shows across gap closed that cuts selected by optimality are stronger until convergence to the final bound, i.e. until $\nexists \rho$ s.t. $\hat{\mathcal{I}}_X(\rho) > 0$.
- **All violated cuts are found by feasibility selection** (Fig. 12).
  - Feasibility selection based on the violation $\lambda_{\min}(\rho)$ identifies violated cuts for the entire selection at every cut round assuming there are enough violated cuts remaining overall.
  - Random selection finds violated cuts only for a random selection subset, but is expected probabilistically, given sufficiently many cut rounds, to eventually find all violated cuts.
  - Optimality selection finds strong, violated ($\hat{\mathcal{I}}_X(\rho) > 0$) cuts for the entire selection size in the first few cut rounds. But eventually, we find no more cuts that are both optimality- and feasibility-based because, as suggested by Eq. (3), there may be fewer optimality- than feasibility-based cuts. We effectively exhaust the supply of strong, violated cuts before adding all the relevant violated cuts. Figure 12b shows, however, the strong, violated cut percentages drop only after most gap to $\mathcal{M} + \mathcal{S}_3^E$ is already closed.
  - Selecting by optimality measure $\mathcal{I}_X(\rho)$ rather than estimated $\hat{\mathcal{I}}_X(\rho)$ shows minor improvements, due to estimator error when $\mathcal{I}_X(\rho) \searrow 0$ (Section 5.1).

So optimality and feasibility selection are complementary, and a combined (optimality and feasibility) ordering can capture both strong and all violated cuts.

Definition 5.2.1 *(Combined measure for cut selection)*
For a given selection size s, at cut round r define the combined measure $\mathcal{C}(\rho)$ $\forall \rho \in \mathcal{V}$:

i. $\mathcal{C}(\rho) = \hat{\mathcal{I}}_X(\rho)$, if an optimality selection measure picked only feasibility-based cuts at r, i.e.

$$|\{\rho \in \mathcal{V} : \hat{\mathcal{I}}_X(\rho) > 0,\ \lambda_{\min}(\rho) < 0\}| \geq s$$

.

ii. $\mathcal{C}(\rho) = -\lambda_{\min}(\rho)$, if at any round $r_{prev} < r$ an optimality selection measure picked proportionally fewer feasibility-based cuts than would random chance, i.e. if at any round $r_{prev} < r$ we had

$$\frac{|\{\rho \in \mathcal{V} : \hat{\mathcal{I}}_X(\rho) > 0,\ \lambda_{\min}(\rho) < 0\}|}{s} < \frac{|\{\rho \in \mathcal{V} : \lambda_{\min}(\rho) < 0\}|}{|\mathcal{V}|}.$$

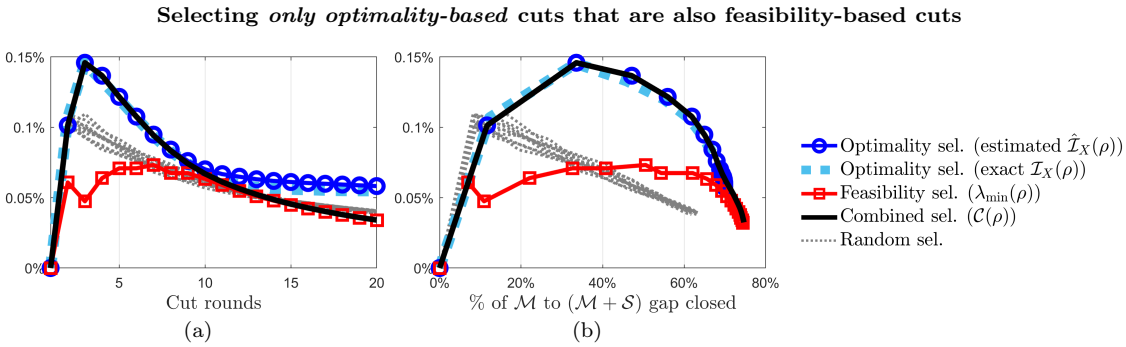


Fig. 11: Comparison, in terms of selecting optimality-based cuts that are also feasibility-based cuts, between Table 3 low-dimension cut selection/ordering strategies to outer-approximate $\mathcal{M}+\mathcal{S}_3^E$. This illustration on BoxQP instance *spar-100-025-1* has the same Algorithm 1 setup as Figs. 9–10, except each round optimality-only strategies add only feasibility-based cuts within the selection size associated with positive optimality measure (that predicts their violation). To compare the strength of cuts selected by each strategy, the y-axis shows (across cut rounds and gaps closed/bounds) the average percent gap closed between $\mathcal{M}$ and $\mathcal{M}+\mathcal{S}$ for each additional feasibility-based cut. This percent is a cumulative ratio between the total gap closed over the total number of feasibility-based cuts added until the current cut round (the effect/strength of added cuts is cumulative).

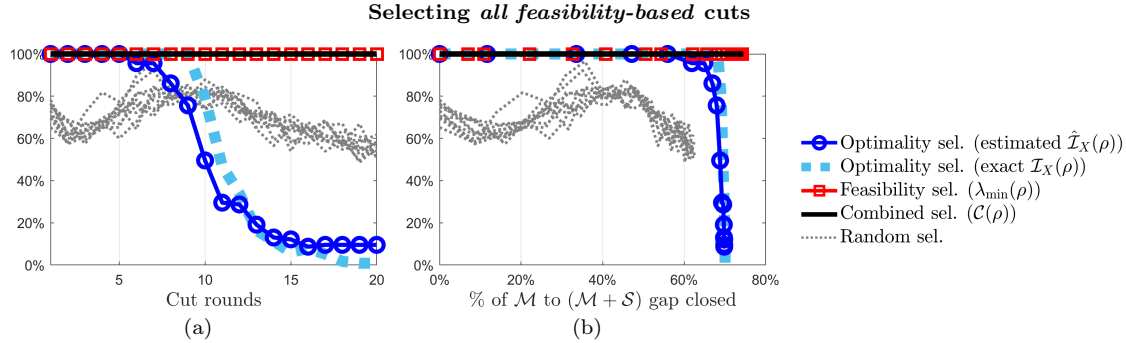


Fig. 12: Comparison, in terms of selecting or finding all feasibility-based cuts, between Table 3 low-dimension cut selection/ordering strategies to outer-approx $\mathcal{M}+\mathcal{S}_3^E$. This illustration on BoxQP instance *spar-100-025-1* has the same Algorithm 1 setup as Figs. 9–10, except, at each round, optimality-only strategies add only feasibility-based cuts within the selection size associated with positive optimality measure (that predicts their violation). The y-axis shows (across cut rounds and gaps closed/bounds) the percent of feasibility-based cuts actually added to the relaxation at each cut round from among the selection of cuts made by each strategy. A low percent of feasibility-based cuts in a selection by a strategy at a cut round can be interpreted as a high rate of false positives.

iii. Otherwise,

$$\mathcal{C}(\rho) = \begin{cases} \hat{\mathcal{I}}_X(\rho) + M, & \text{if } \hat{\mathcal{I}}_X(\rho) > 0 \text{ and } \lambda_{\min}(\rho) < 0, \\ -\lambda_{\min}(\rho) & \text{otherwise,} \end{cases} \qquad (\mathcal{C}(\rho))$$

where M is an arbitrary large positive number.

In worst case, the combined selection computational cost sums both optimality and feasibility measures. But, in early cut rounds, we only have to check the feasibility measure of cuts selected by the optimality measure, and combined cut selection can be as quick as the optimality measure. □

Motivated by the Figs. 11 and 12 observations on optimality- and feasibility-based cuts, $\mathcal{C}(\rho)$ prioritizes strong feasibility-based cut selection via a positive expected objective improvement

$\hat{\mathcal{I}}_X(\rho) > 0$ and uses a feasibility check to ensure violated cuts (the weakness of optimality selection). If optimality-based cut selection does not find a full selection of violated cuts, further feasibility-based cuts are found in a second tier of subproblems that offer no useful objective improvement information via $\hat{\mathcal{I}}_X(\rho)$. When optimality selection becomes worse at finding feasibility-based cuts than random, we switch to only feasibility-based cut selection. Combined cut selection via $\mathcal{C}(\rho)$ is thus expected to outperform optimality-based selection in the sense that the combined measure will always incorporate the additional feasibility-based cuts that prevented the optimality measure from converging. Fig. 12a shows that we may exhaust the set of cuts that are both optimality- and feasibility-based, i.e. violated eigenvalue cuts from subproblems ρ with $\hat{\mathcal{I}}_X(\rho) > 0$, before identifying all feasibility-based cuts. Heuristically, our tests show that combined cut selection outperforms feasibility-based selection because of the strong performance of early iterations. These observations are illustrated in Figs. 9 and 11 and further motivated by Table 6. In terms of the combined selection running time: (i) fast neural network evaluations (Table 2) identify optimality-based cuts (Fig. 11), leading to stronger overall performance than feasibility selection in early iterations; (ii) feasibility selection, when needed, trades overhead (to get $\lambda_{\min}(\rho)$ by eigendecomposition) for more violated cuts (Fig. 12) and stronger final bounds compared to optimality selection.

The results in Tables 5, 7 and 8 show bounds at the termination criteria of Algorithm 1, i.e. two consecutive cut rounds fail to improve the objective by $\geq 0.1\%$. Definition 5.2.1 ensures that the final gaps closed by the feasibility and combined measures will both converge to the same $\mathcal{M} + \mathcal{S}_n^E$ bound. To remove redundancy between the final gaps of the feasibility and combined measures, we exclude the combined ranking from Tables 5, 7 and 8.

Table 5 analyzes the BoxQP final bounds obtained by pure optimality and feasibility selections, showing the (expected) gap in bounds between the two strategies is reduced by:

(o4) Increasing selection size, e.g. 5% to 10%, at each Algorithm 1 cut round. Similar to Grötschel et al. [51] and other classical texts, we observe an engineering trade-off between the number of cuts added in each round, e.g. 5% versus 10% of the set $|\mathcal{P}_3^E|$.
(o5) Adding extra cuts, i.e. $\triangle$ on top of $\mathcal{M}$. It is well-known in the max-cut literature, e.g. Helmberg and Rendl [55] and Rendl et al. [95], that $\mathcal{M} + \triangle + \mathcal{S}$ gives a significant advantage over $\mathcal{M} + \triangle$. Table 5 shows the same effect observed by Rendl and co-workers, e.g. the heading *10% ρ for* $\mathcal{M} + \triangle + \mathcal{S}_3^E$, shows that incorporating *either* the optimality- or feasibility-based cuts improves the $\mathcal{M} + \triangle$ bound. However, once the triangle inqualities are incorporated, we observe effectively no difference in the optimality- versus feasibility-based strategies for generating the eigenvalue cuts.
(o6) Higher instance density.

Observations (o4)-(o6) are all justified by Lemma 3.2.3(ii), where for any semidefinite ρ subproblem, more additional cut classes restrict the convex set $\mathcal{B}$ referenced in the lemma. In particular, Lemma 3.2.3(ii) justifies (o6) since a higher instance density implies a larger selection of external cuts/subproblems outer-approximating a more restricted $\mathcal{S}_n^E$ relaxation (Corollary 3.1.3), in turn implying a more restricted convex set $\mathcal{B}$ for any ρ subproblem.

Table 6 largely echoes the results of Qualizza et al. [94] w.r.t their discussions and results of sparse versus dense eigenvalue cuts in BoxQP. There are several possible trade-offs between sparse and dense eigenvalue cuts: (i) number of cuts added per round, (ii) cut dimensionality, and (iii) bound achieved within a certain time or number of cuts. First, with respect to number of cuts, there are typically fewer dense than sparse cuts, i.e. N versus $\binom{N}{n}$ for a fully-dense instance. Table 6 only uses 5% of the possible cuts, but this is still typically greater than the number of possible dense cuts. Second, with respect to cut dimensionality, we repeat the trade-off noted by Qualizza et al. [94] when we observe that dense cuts may require more computationally

Table 5: (Table 4 BoxQP instances) Comparison of % of $\mathcal{M}$ to optimality gap closed by Algorithm 1 at criteria termination, i.e. final bounds, for optimality versus feasibility selections with three setups: (i)-(ii) outer-approximate $\mathcal{M}+\mathcal{S}_3^E$ with selection sizes 5% and 10% of $|\mathcal{P}_3^E|$, respectively; (iii) outer-approximate $\mathcal{M}+\triangle+\mathcal{S}_3^E$ (linear base $\mathcal{B} = \mathcal{M}+\triangle$) with selection size $10\% \cdot |\mathcal{P}_3^E|$. Columns labeled "Diff." show the difference between gaps closed by feasibility and optimality.

		5% ρ for $\mathcal{M}+\mathcal{S}_3^E$			10% ρ for $\mathcal{M}+\mathcal{S}_3^E$			10% ρ for $\mathcal{M}+\triangle+\mathcal{S}_3^E$			
Size	Density	Opt.	Feas.	Diff.	Opt.	Feas.	Diff.	$\mathcal{M}+\triangle$	Opt.	Feas.	Diff.
Small	Low	87.53	91.12	3.59	88.45	91.41	2.96	98.98	99.93	99.97	0.04
	Medium	85.40	87.74	2.34	86.28	87.86	1.58	98.16	99.86	99.88	0.02
	High	85.19	87.10	1.91	85.90	87.13	1.23	98.48	99.60	99.62	0.02
Medium	Low	79.22	83.97	4.75	80.73	84.73	4.00	99.75	99.98	99.99	0.01
	Medium	78.22	80.50	2.28	78.99	80.59	1.60	97.72	98.53	98.54	0.01
	High	72.17	73.26	1.09	72.72	73.28	0.56	92.72	94.50	94.50	0.00
Large	Low	73.25	76.80	3.55	74.41	77.46	3.05	98.81	99.50	99.53	0.03
	Medium	75.50	77.51	2.01	76.26	77.55	1.29	97.15	97.96	97.98	0.02
	High	70.13	70.97	0.84	70.54	70.99	0.45	90.06	91.36	91.35	-0.01
Jumbo	Low	71.58	74.68	3.10	72.41	75.00	2.59	96.91	97.89	97.90	0.01
	Medium	70.38	71.76	1.38	70.94	71.78	0.84	90.99	92.09	92.09	0.00
	High	66.63	67.36	0.73	66.77	67.37	0.60	85.36	86.26	86.26	0.00

demanding LP pivoting operations than sparse cuts. Third and finally, Table 6 shows the trade-offs between number of cuts and cut dimensionality:

- Low- and medium-density instances strongly favor sparse cuts, both with respect to time and gap closed.
- Only the highest-density instances favor the dense cuts. There is no advantage for using our proposed sparse cuts for fully-dense instances. For instances that are even partially sparse such as `spar100-075-1` in Figs. 9d and 10d, there may be an advantage in using sparse cuts for the engineering trade-off of bound achieved versus time spent.
- The LP time needed for each cut round stays roughly constant for the sparse cuts but significantly increases over the cut rounds for the dense instances.

These observations lead to the Table 3 conclusions. Cut selection via optimality converges faster than via feasibility at the cost of a final bounds gap (bridged by combined selection). But this gap can vanish after adding other cut classes, typically incorporated in Branch&Cut frameworks. In other words, when mixing different cut classes for lower bounding, only optimality-based, fast-converging, low-dimensional semidefinite cuts are needed rather than all feasibility-based ones!

5.3 Sparsity of cutting planes

5.3.1 Dimensionality or sparsity of cuts

We explore the effect of increasing dimensionality (decreasing sparsity) n of subproblems/cuts when outer-approximating relaxations ($\mathcal{M} + \mathcal{S}_n^E$). For cut dimensionality $n \in \{3, 4, 5\}$ and optimality/feasibility cut selection on BoxQP instances, the aggregated results reported in Table 7 illustrate that with increasing n:

(o7) The final bound reached by feasibility-based cuts quickly tails off. Specifically, the gap between $n = 5$ and $n = 4$ is much less marked than the one between $n = 4$ and $n = 3$, regardless of the instance size and the number of subproblems $|\mathcal{P}_n^E|$. This corresponds to Remark 3.1.4 that low-dimensional linear outer-approximations offer a good bound/complexity trade-off.

Table 6: (Table 4 BoxQP instances) Bound and time comparison, across 4 cut rounds, between Table 3 low-dimension cut selection/ordering strategies to outer-approximate $\mathcal{M}+\mathcal{S}_3^E$ and adding all violated full matrix (high-dimension) eigenvalue cuts to outer-approximate $\mathcal{M}+\mathcal{S}$. For a strategy, Algorithm 1 adds to $\mathcal{M}$ every round, given the rounds' $\mathcal{P}_3^E$ ordering, any feasibility-based low-dimensional EigCut(ρ) for ρ within the top $10\% \cdot |\mathcal{P}_3^E|$ selection. The bold entries show whether a low-dimension (Comb.) or high-dimension (Dense) approach gives a stronger bound or faster time.

(a) Percentage gap closed between $\mathcal{M}$ and $\mathcal{M}+\mathcal{S}$ bounds

		Round 1					Round 2					Round 3					Round 4				
Size	Dens.	Opt.	Feas.	Comb.	Dense	Rand.	Opt.	Feas.	Comb.	Dense	Rand.	Opt.	Feas.	Comb.	Dense	Rand.	Opt.	Feas.	Comb.	Dense	Rand.
Small	Low	41.2	22.6	**41.2**	0.0	23.8	71.6	42.9	**71.6**	0.0	39.7	81.5	66.8	**81.8**	0.0	50.8	85.4	78.4	**86.0**	0.0	59.9
	Med.	39.7	26.2	**39.7**	0.0	30.7	72.8	50.3	**72.8**	0.0	49.6	81.6	70.4	**81.8**	30.2	60.9	84.6	80.0	**84.9**	44.9	68.0
	High	41.8	29.0	**41.8**	19.6	39.5	75.9	65.1	**75.9**	62.1	60.9	83.4	78.2	**83.5**	75.2	70.3	85.2	84.0	**85.6**	81.2	75.3
Medium	Low	33.9	17.3	**33.9**	0.0	17.5	61.2	38.6	**61.2**	0.0	29.9	71.6	56.8	**71.7**	0.0	40.0	75.9	67.4	**76.3**	0.0	48.2
	Med.	27.3	18.9	**27.3**	0.0	23.7	64.7	46.3	**64.7**	0.0	41.9	74.4	63.3	**74.5**	11.3	53.3	77.3	73.8	**77.4**	26.6	60.1
	High	29.2	17.1	**29.2**	4.1	34.2	65.6	51.5	**65.6**	35.2	54.7	71.7	69.3	**71.4**	52.6	62.5	72.5	72.3	**72.8**	62.9	66.6
Large	Low	31.1	13.9	**31.1**	0.0	14.5	55.5	34.0	**55.5**	0.0	25.8	64.8	51.2	**64.9**	0.0	34.7	69.2	61.1	**69.5**	0.0	42.3
	Med.	25.9	15.2	**25.9**	0.0	26.6	64.5	48.4	**64.5**	0.0	46.7	73.3	64.0	**73.3**	11.4	56.9	75.4	73.0	**75.6**	28.4	62.6
	High	27.0	15.8	**27.0**	3.8	37.1	64.2	49.7	**64.2**	36.3	55.9	69.8	65.8	**69.5**	53.9	62.9	70.4	69.8	**70.6**	64.0	66.4
Jumbo	Low	23.0	13.6	**23.0**	0.0	15.5	52.3	34.2	**52.3**	0.0	27.2	63.5	51.1	**63.5**	0.0	37.0	68.1	62.5	**68.2**	0.0	44.7
	Med.	23.8	14.3	**23.8**	0.0	28.6	62.2	45.3	**62.2**	0.0	48.4	69.4	62.6	**69.2**	14.2	56.9	70.6	69.3	**70.9**	30.5	61.6
	High	19.7	9.8	**19.7**	4.1	28.4	56.7	31.7	**56.7**	36.5	47.2	64.7	48.5	**64.7**	54.0	55.4	66.4	57.5	**66.5**	63.9	59.8

(b) Timings per cut round in format: total time in seconds (percent of it spent on cut selection & generation e.g. non-LP time)

		Round 1					Round 2					Round 3					Round 4				
Size	Dens.	Opt.	Feas.	Comb.	Dense	Rand.	Opt.	Feas.	Comb.	Dense	Rand.	Opt.	Feas.	Comb.	Dense	Rand.	Opt.	Feas.	Comb.	Dense	Rand.
Small	Low	0.01 (65%)	0.02 (80%)	**0.02** (68%)	0.02 (55%)	0.01 (48%)	0.02 (60%)	0.02 (78%)	**0.02** (62%)	0.04 (28%)	0.01 (47%)	0.02 (60%)	0.02 (69%)	**0.02** (62%)	0.08 (15%)	0.01 (45%)	0.01 (62%)	0.02 (69%)	**0.02** (69%)	0.14 (9%)	0.01 (41%)
	Med.	0.05 (68%)	0.05 (83%)	0.05 (69%)	**0.02** (43%)	0.02 (47%)	0.06 (52%)	0.06 (73%)	**0.06** (55%)	0.08 (14%)	0.02 (41%)	0.06 (50%)	0.07 (60%)	**0.07** (54%)	0.13 (8%)	0.02 (35%)	0.06 (51%)	0.08 (53%)	**0.09** (67%)	0.14 (7%)	0.03 (32%)
	High	0.13 (68%)	0.15 (84%)	0.14 (70%)	**0.06** (15%)	0.06 (42%)	0.23 (39%)	0.22 (54%)	0.24 (41%)	**0.13** (6%)	0.1 (27%)	0.27 (32%)	0.32 (38%)	0.34 (48%)	**0.15** (6%)	0.11 (20%)	0.23 (35%)	0.35 (33%)	0.36 (44%)	**0.16** (5%)	0.11 (18%)
Medium	Low	0.03 (54%)	0.03 (73%)	**0.03** (56%)	0.08 (43%)	0.01 (40%)	0.03 (45%)	0.03 (69%)	**0.04** (48%)	0.2 (19%)	0.01 (38%)	0.03 (47%)	0.03 (61%)	**0.04** (49%)	0.38 (10%)	0.01 (33%)	0.03 (48%)	0.04 (54%)	**0.04** (55%)	0.54 (7%)	0.02 (30%)
	Med.	0.18 (51%)	0.18 (72%)	0.19 (54%)	**0.13** (31%)	0.09 (31%)	0.41 (23%)	0.26 (50%)	0.42 (26%)	**0.4** (11%)	0.13 (20%)	0.49 (20%)	0.44 (29%)	**0.54** (30%)	1.35 (3%)	0.2 (14%)	0.45 (20%)	0.61 (21%)	**0.55** (33%)	1.8 (2%)	0.24 (11%)
	High	1.17 (41%)	0.96 (69%)	1.25 (44%)	**0.6** (11%)	0.82 (19%)	5.32 (9%)	3.19 (20%)	5.37 (10%)	**2.73** (2%)	2.43 (6%)	7.18 (7%)	8.67 (8%)	7.62 (13%)	**3.66** (2%)	3.52 (4%)	5.83 (8%)	9.34 (7%)	9.19 (8%)	**4.94** (1%)	3.66 (3%)
Large	Low	0.09 (38%)	0.08 (60%)	**0.09** (40%)	0.34 (29%)	0.03 (28%)	0.11 (30%)	0.08 (55%)	**0.12** (32%)	0.95 (11%)	0.04 (27%)	0.11 (32%)	0.11 (44%)	**0.11** (34%)	1.66 (7%)	0.04 (24%)	0.11 (30%)	0.13 (36%)	**0.12** (34%)	2.64 (4%)	0.05 (19%)
	Med.	0.66 (37%)	0.51 (67%)	0.69 (41%)	**0.46** (24%)	0.39 (20%)	2.32 (11%)	1.21 (29%)	2.32 (12%)	**1.85** (6%)	0.77 (9%)	3.41 (7%)	2.68 (13%)	**3.6** (12%)	6.68 (2%)	1.4 (5%)	2.89 (9%)	4.3 (8%)	**3.31** (15%)	9.59 (1%)	1.72 (4%)
	High	2.9 (31%)	1.92 (64%)	3.02 (33%)	**1.51** (8%)	2.51 (12%)	20.26 (4%)	9.7 (12%)	20.46 (5%)	**7.87** (1%)	10.4 (3%)	28.22 (3%)	31.74 (4%)	29.56 (6%)	**10.21** (1%)	14.46 (1%)	22.94 (4%)	35.62 (4%)	34.28 (4%)	**14.27** (1%)	14.14 (1%)
Jumbo	Low	0.29 (29%)	0.25 (47%)	**0.3** (32%)	1.53 (16%)	0.14 (18%)	0.54 (16%)	0.31 (38%)	**0.55** (17%)	5.0 (5%)	0.16 (15%)	0.71 (13%)	0.49 (24%)	**0.7** (14%)	9.58 (3%)	0.22 (11%)	0.76 (11%)	0.86 (14%)	**0.83** (18%)	15.35 (2%)	0.34 (8%)
	Med.	2.82 (24%)	1.69 (54%)	2.87 (25%)	**2.23** (12%)	2.09 (10%)	20.99 (3%)	8.02 (11%)	21.13 (4%)	**11.64** (2%)	7.43 (3%)	30.92 (2%)	30.77 (3%)	**31.98** (4%)	49.28 (1%)	14.57 (1%)	23.11 (3%)	40.46 (2%)	**32.3** (4%)	72.39 (0%)	16.45 (1%)
	High	5.67 (32%)	4.12 (65%)	**5.86** (34%)	7.6 (4%)	4.91 (7%)	55.52 (3%)	13.73 (19%)	55.54 (4%)	**38.25** (1%)	20.91 (2%)	99.84 (2%)	65.89 (4%)	102.32 (3%)	**59.72** (0%)	50.07 (1%)	101.64 (2%)	159.58 (2%)	110.43 (4%)	**96.1** (0%)	60.86 (0%)

(o8) The final bound reached by optimality-based cuts decreases on low density instances, while it increases on medium and high density instances. This behavior, independent of instance size, is consistent with the Figs. 9 and 10 and Table 6 observations that higher density instances favor higher density cuts.

(o9) The gap between between feasibility- and optimality-based cuts increases on low density instances. This is consistent with Lemma 3.2.3(ii), i.e., given any two subproblems $\rho_1 \subset \rho_2$, the PSD completion in $P^{\tilde{x}}(\rho_1)$ has a larger fixing size than in $P^{\tilde{x}}(\rho_2)$ with extra fixed variables $X_{\rho_2 \setminus \rho_1}$, implying the lower-dimensional ρ_1 subproblem is better outer-approximated than ρ_2 via optimality selection.
However, the picture is less clear on medium and high density instances. Indeed, on medium density instances the gap between feasibility- and optimality-based cuts increases from $n = 3$ to $n = 4$, but decreases from $n = 4$ to $n = 5$. This is not unexpected because: (i) the bound reached by feasibility cuts tails off from $n = 4$ to $n = 5$ (see (o7)) and (ii) the gap between feasibility and optimality cuts decreases when the density of the instances increases (see (o6)).

(o10) The number of subproblems $|\mathcal{P}_n^E|$ drastically increases on high density instances, as expected from Remark 3.1.4. The appropriate number of cuts is an engineering trade-off. For a fair comparison, Table 7 uses the same parameter values ($10\% \cdot |\mathcal{P}_n^E| \approx 10^5$ for the large, high-density instances) throughout. A smaller fraction of higher-dimensional cuts per round, e.g. $1\% \cdot |\mathcal{P}_n^E|$, would have maintained tractability for the jumbo size high density models.

From observations (o7)-(o10) the following implications arise:

(i) The Section 4 estimator becomes inaccurate for increasing n. But this low-dimensional limitation is complementary to optimality-based cut selection: both technologies are most effective for small n.

(ii) The intuition of sparsifying cuts [94] is valid, with bound improvements that tail off with decreasing cut sparsity (increasing n) - this justifies our bottom-up approach to cut dimensionality n, contrary to [94, 105].

(iii) The MIP intuition [36–38] that sparse cuts are strong extends to a continuous setting.

Remark 5.3.1 *($\mathcal{S}_n^E$ cut selection strategy & dim. choices for Algorithm 1 based on size/sparsity)* Observations (o1)-(o3) and (o6)-(o10), supported by Section 3, apply to all QP instances. Thus, the observations inform heuristic guidelines on choosing the $\mathcal{S}_n^E$ cut selection strategy and n. These guidelines, which are based on QP instance size/sparsity, obtain both strong convergence and final bounds while managing computational cost:

- Choose cut dimensionality $n \in \{3, 4, 5\}$ based on instance size N and sparsity of E:
  - *For sparse E, higher n indifferent of size N.*
  - *For dense E, higher n for smaller N and lower n for larger N.*
- Choose cut selection strategy (optimality, combined) based on instance sparsity of E:
  - *For sparse E, combined cut selection.*
  - *For dense E, optimality cut selection.*

□

5.3.2 Incorporating chordal extensions

We explore the effect on final bounds for cut selection strategies using chordal extensions of an instance sparsity graph. For $n = 3$, we compare selecting cuts based on set $\mathcal{P}_3^{\overline{E}}$ (Lemma 3.1.2) and (chordal) relaxation $\mathcal{S}_3^{\overline{E}}$ versus based on set $\mathcal{P}_3^E$ (Corollary 3.1.3) and relaxation $\mathcal{S}_3^E$ for BoxQP instances. Table 8 shows that, by considering chordal extensions, feasibility final bounds (fourth

Table 7: (Table 4 BoxQP instances) Comparison of % of $\mathcal{M}$ to optimality gap closed by Algorithm 1 at criteria termination, i.e. final bounds, within 40 rounds of cuts, for optimality and feasibility selections outer-approximating ($\mathcal{M} + \mathcal{S}_n^E$) with selection size $10\% \cdot |\mathcal{P}_n^E|$ for all $n \in \{3, 4, 5\}$. For missing entries we run out of memory within 40 rounds of cuts.

		$\mathcal{M}$+$\mathcal{S}_3^E$		$\mathcal{M}$+$\mathcal{S}_4^E$		$\mathcal{M}$+$\mathcal{S}_5^E$		# subproblems ρ		
Size	Density	Opt.	Feas.	Opt.	Feas.	Opt.	Feas.	$\lvert\mathcal{P}_3^E\rvert$	$\lvert\mathcal{P}_4^E\rvert$	$\lvert\mathcal{P}_5^E\rvert$
Small	Low	88.45	91.41	88.83	94.66	87.11	94.64	389	248	196
	Medium	86.28	87.86	92.49	95.46	91.92	96.27	1,250	1,716	1,084
	High	85.90	87.13	96.19	97.92	97.41	99.26	3,104	13,528	37,998
Medium	Low	80.73	84.73	80.45	88.18	79.32	88.21	616	416	344
	Medium	78.99	80.59	88.95	91.75	89.32	93.53	3,799	6,131	3,992
	High	72.72	73.28	90.63	92.33	92.25	96.29	21,540	142,302	543,628
Large	Low	74.41	77.46	73.19	81.53	72.47	81.54	1,473	896	784
	Medium	76.26	77.55	89.64	91.40	91.23	93.99	11,055	25,595	23,142
	High	70.54	70.99	88.72	90.73	90.74	95.65	39,022	316,784	1,490,870
Jumbo	Low	72.41	75.00	73.37	80.98	71.89	81.15	3,530	2,195	1,775
	Medium	70.94	71.78	86.91	88.58	88.87	92.06	27,441	90,216	115,944
	High	66.77	67.37	86.25	87.76	-	-	91,917	1,013,518	6,551,337

Table 8: (Table 4 BoxQP instances) Comparison of % of $\mathcal{M}$ to optimality gap closed by Algorithm 1 at criteria termination, i.e. final bounds, for outer-approximating: (i) relaxation $\mathcal{M} + \mathcal{S}_3^E$; (ii) relaxation $\mathcal{M} + \mathcal{S}_3^{\overline{E}}$ based on chordal extensions; (iii) relaxation $\mathcal{M} + \mathcal{S}(\overline{\mathcal{P}_3^*})$ restricting zero-coefficient variables in chordal extensions. Algorithm 1 is used with selection size 10% of $|\mathcal{P}_3^E|$, $|\mathcal{P}_3^{\overline{E}}|$ and $|\overline{\mathcal{P}_3^*}|$, respectively; the previous three set cardinalities $|\mathcal{P}_3^{E^+}|$ are compared in the last four columns.

		$\mathcal{M}$+$\mathcal{S}_3^E$		$\mathcal{M}$+$\mathcal{S}_3^{\overline{E}}$		$\mathcal{M}$+$\mathcal{S}(\overline{\mathcal{P}_3^*})$	# subproblems ρ			
Size	Density	Opt.	Feas.	Opt.	Feas.	Opt.	$\lvert\mathcal{P}_3^E\rvert$	$\lvert\mathcal{P}_3^{\overline{E}}\rvert$	$\lvert\overline{\mathcal{P}_3^*}\rvert$	$\lvert\mathcal{P}_3^{E^+}\rvert$
Small	Low	88.45	91.41	88.20	92.24	89.41	389	3,995	1,654	9,880
	Medium	86.28	87.86	86.52	87.97	86.85	1,250	5,024	3,413	7,345
	High	85.90	87.13	86.04	87.13	86.07	3,104	4,725	4,404	5,235
Medium	Low	80.73	84.73	81.80	87.25	83.79	616	11,531	3,402	29,126
	Medium	78.99	80.59	79.07	80.65	79.66	3,799	22,509	12,763	32,755
	High	72.72	73.28	72.96	73.28	72.97	21,540	49,462	41,965	54,740
Large	Low	74.41	77.46	73.88	79.47	77.19	1,473	46,935	10,252	98,245
	Medium	76.26	77.55	75.88	77.56	76.91	11,055	74,379	39,504	98,245
	High	70.54	70.99	70.40	70.97	70.54	39,022	90,061	76,374	98,245
Jumbo	Low	72.41	75.00	71.31	76.06	74.60	3,530	122,106	25,829	226,672
	Medium	70.94	71.78	69.85	71.73	71.01	27,441	181,912	97,975	226,672
	High	66.77	67.37	66.61	67.34	66.70	91,917	211,684	179,703	226,672

column) marginally improve, although not in all cases, but optimality final bounds (in the third column) tend instead to decrease, especially on large and jumbo instances. The latter result is not surprising, considering that chordal extensions to uniformly (not chordal) sparse BoxQP instances add many new subproblems and cuts ($|\mathcal{P}_3^{\overline{E}}| \gg |\mathcal{P}_3^E|$) with variables not participating in the objective. Selecting from $\mathcal{P}_3^{\overline{E}}$ therefore involves many cuts offering little objective improvement.

One way to alleviate decreasing optimality final bounds for chordal extensions is to eliminate subproblems with many zero-coefficient variables by restricting cover $\mathcal{P}_3^{\overline{E}}$ to, for example,

$$\overline{\mathcal{P}_3^*} := \{\rho \subseteq \rho' \in \mathcal{P}_3^{\overline{E}} \mid \nexists i \in \rho \text{ s.t. } \forall j \in \rho \setminus \{i\}\ Q_{ij} = 0,\ \nexists \rho_2 \in \overline{\mathcal{P}_3^*} \text{ s.t. } \rho \subset \rho_2\}.$$

Table 8 shows improved optimality final bounds (fifth column) for relaxation $\mathcal{M} + \mathcal{S}(\overline{\mathcal{P}_3^*})$, especially for sparser instances where the number of subproblems is significantly reduced. Alterna-

tively, the optimality measure can be used to obtain the higher feasibility final bounds for cover $\mathcal{P}_3^E$ (second column) or $\mathcal{P}_3^{\overline{E}}$ (fourth column) via the combined selection measure $\mathcal{C}(\rho)$.

For BoxQP instances in particular, Tables 7 and 8 show, for low to medium density, increasing cut dimensionality may be preferable to adding chordal extensions due to: (i) improved final bounds and (ii) a more slowly growing number of subproblems. Furthermore, recall from Table 5 that the final $\mathcal{M} + \mathcal{S}_3^E$ bounds had differences of up to 4-5% for optimality versus feasibility measures. However, these 4-5% differences disappear when incorporating the triangle inequalities. Table 8 shows similarity between the $\mathcal{M}+\mathcal{S}_3^E$ and $\mathcal{M}+\mathcal{S}_3^{\overline{E}}$ bounds (in the feasibility columns), so from the experience of Table 5, we expect the difference between the $\mathcal{M}+\mathcal{S}_3^E+\triangle$ and $\mathcal{M}+\mathcal{S}_3^{\overline{E}}+\triangle$ bounds to be even smaller.

Therefore, chordal extensions, while not particularly suited to uniform sparsity (such as in BoxQP), may offer advantages [34] for other instances with (nearly [24]) chordal sparsity patterns, e.g. small $|\overline{E} \setminus E|$, via either the optimality or combined cut selection measures.

5.4 Mixing with other linear cut classes

This section shows that, even in conjunction with other cut classes for QP as base relaxations, our outer-approximations (see Remark 5.4.1) are: (i) complementary by further improving bounds (Table 9), (ii) computationally light by minimal running time overhead (Fig. 14) and (iii) light in numbers of extra generated cutting planes (Fig. 13).

To this extent, we combine the outer-approximations obtained through semidefinite cut selection with $\triangle$ inequalities. Practically, due to their $O(N^3)$ number, $\triangle$ inequalities are not all added at once (like the $\mathcal{M}$ inequalities), but instead are selected/separated by an ordering based on violation. Also, to avoid separating too many weak $\triangle$ inequalities in sparse QP instances, we heuristically first separate violated (i, j, k) triangle inequalities with all relevant objective coefficients $Q_{ij},\ Q_{ik},\ Q_{jk} \neq 0$, followed by violated (i, j, k) triangle inequalities with one zero among relevant objective coefficients. The described separation is intended to be similar to that in [18], to show that even a strong relaxation can be augmented by the semidefinite cuts the paper builds with negligible computational overhead.

Remark 5.4.1 *(Relaxations compared in Table 9, Figs. 13 and 14)*

- $\mathcal{M} + \triangle$ is used as a linear base relaxation in Algorithm 1 and a proxy for the relaxation $\mathcal{M} + 0\text{-}1/2$ [18]. The $\mathcal{M}+\triangle$ bounds in Table 9 are stronger than those of $\mathcal{M} + 0\text{-}1/2$ [18] which omits $\mathcal{M}$ cutting planes $(\forall i \in V)\ X_{ii} \geq 2x_i - 1$.
- The naïve $\mathcal{M}+\triangle+\mathcal{S}_3$ relaxation is implemented via Algorithm 1 with optimality cut selection.
- The heuristic $\mathcal{M} + \triangle + \mathcal{S}_{3-5}$ relaxation is implemented via Algorithm 1 with choices based on Remark 5.3.1, outer-approximating:
  - $\mathcal{M} + \triangle + \mathcal{S}_5$ with combined cut selection (Definition 5.2.1) for low and medium density BoxQP instances;
  - $\mathcal{M} + \triangle + \mathcal{S}_4$ with optimality cut selection for high density, <jumbo size BoxQP instances;
  - $\mathcal{M} + \triangle + \mathcal{S}_3$ with optimality cut selection for high density, jumbo size BoxQP instances.
- $\mathcal{M}^2+0\text{-}1/2$ [18] (denoted by BGL) is a very good proxy of the current CPLEX linear relaxation for BoxQP instances. Since the BGL relaxation models explicitly quadratic terms x_i^2 $(\forall i \in V$ s.t. $Q_{ii} \geq 0)$, its bounds are comparable to $\mathcal{M} + 0\text{-}1/2+$ SOCP, and is therefore a useful benchmark for both naïve $\mathcal{M} + \triangle + \mathcal{S}_3$ and heuristic $\mathcal{M} + \triangle + \mathcal{S}_{3-5}$.

□

Table 9: (Table 4 BoxQP instances) Comparison of % of $\mathcal{M}$ to optimality gap closed by Algorithm 1 at criteria termination, i.e. final bounds, using a strong base relaxation $\mathcal{M}+\triangle$ against the SDP, SDP+linear and state-of-the-art BGL [18] relaxations. The relaxations $\mathcal{M}+\triangle+\mathcal{S}_3^E$ and $\mathcal{M}+\triangle+\mathcal{S}_{3-5}^E$ are outer-approximated via Algorithm 1 as described in Remark 5.4.1 with 10% of all subproblems/cuts selected at each cut round. Columns labeled "Diff." show the difference versus another relaxation in terms of the % gap closed between that relaxation bound and optimality.

		SDP	SDP+linear		CPLEX	Base	Naïve $\mathcal{M}+\triangle+\mathcal{S}_3^E$			Heur. $\mathcal{M}+\triangle+\mathcal{S}_{3-5}^E$		
Size	Density	$\mathcal{S}$	$\mathcal{S}^{\geq}$	$\mathcal{M}+\mathcal{S}$	BGL [18]	$\mathcal{M}+\triangle$		Diff. BGL	Diff. $\mathcal{M}+\triangle$		Diff. BGL	Diff. $\mathcal{M}+\triangle$
Small	Low	80.65	99.11	99.29	99.51	98.98	99.93	85.71	93.14	99.97	93.88	97.06
	Medium	89.79	99.40	99.46	99.29	98.16	99.86	80.28	92.39	99.95	92.96	97.28
	High	94.15	99.76	99.80	99.13	98.48	99.60	54.02	73.68	99.76	72.41	84.21
Medium	Low	85.85	99.33	99.55	99.90	99.75	99.98	80.00	92.00	99.98	80.00	92.00
	Medium	93.00	98.77	98.86	98.01	97.72	98.53	26.13	35.53	99.03	51.26	57.46
	High	95.68	99.24	99.31	93.52	92.72	94.50	15.12	24.45	96.14	40.43	46.98
Large	Low	88.61	98.20	98.65	98.28	98.81	99.50	70.93	57.98	99.58	75.58	64.71
	Medium	94.96	99.05	99.25	97.48	97.15	97.96	19.05	28.42	98.73	49.60	55.44
	High	96.34	99.14	99.29	90.60	90.06	91.36	8.09	13.08	93.36	29.36	33.20
Jumbo	Low	92.90	98.35	98.84	96.28	96.91	97.89	43.28	31.72	98.07	48.12	37.54
	Medium	95.25	98.60	98.82	91.42	90.99	92.09	7.81	12.21	95.57	48.37	50.83
	High	96.67	98.96	99.16	85.68	85.36	86.26	4.05	6.15	86.26	4.05	6.15

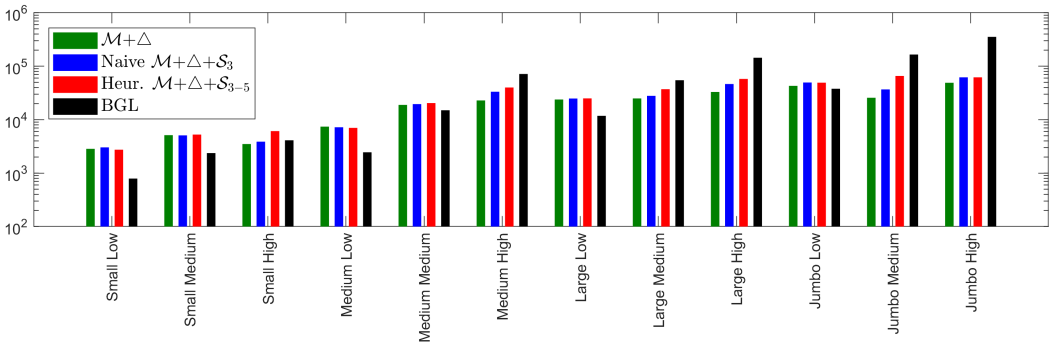


Fig. 13: Number of cuts generated for all BoxQP instances (Table 4) to obtain Table 9 bounds.

More specifically, in Table 9, improving upon both BGL and the base linear relaxation $\mathcal{M}+\triangle$ bounds, both the naïve and heuristic outer-approximations offer bounds that for sparser, non-jumbo size instances beat $\mathcal{S}$ bounds and are competitive with expensive $\mathcal{M}+\mathcal{S}$ bounds [6]. The heuristic $\mathcal{M}+\triangle+\mathcal{S}_{3-5}^E$ relaxation is slightly heavier than the naïve $\mathcal{M}+\triangle+\mathcal{S}_3^E$ one, showing further improved bounds in exchange for extra running time and cuts (Figs. 13 and 14). However, both naïve or heuristic relaxations can be used in practice, as they both involve cut numbers similar to BGL (Fig. 13) and minimal extra running time to the base relaxation $\mathcal{M}+\triangle$ (or potentially $\mathcal{M}+0\text{-}1/2$ or BGL).

6 Extending optimality-based semidefinite cuts to QCQP problems

This section extends optimality-based cut selection to QCQP instances, building on the engineering trade-offs uncovered in the BoxQP instances. Specifically, we (i) choose a low-dimensional

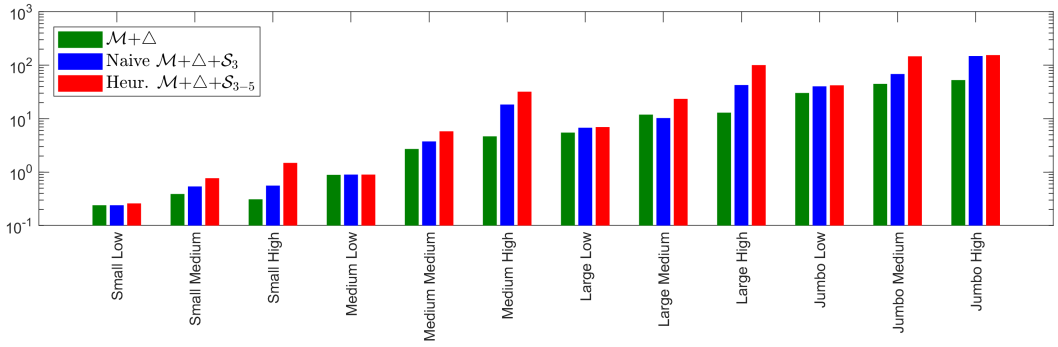


Fig. 14: Running time (in seconds) for all BoxQP instances (Table 4) to obtain Table 9 bounds.

sparse vertex cover $\mathcal{P}_n^{E_m}$ based on the problem sparsity analyzed in Sections 3.1 and 5.3, and (ii) employ the combined selection measure $\mathcal{C}(\rho)$ from Section 5.2.

Definition 6.1 *(Combined measure for QCQP cut selection based on objective-only structure)* A straightforward QCQP extension of $\mathcal{C}(\rho)$ can prioritize the structure in the objective versus constraints by scoring subproblems $\rho \in \mathcal{P}_n^{E_m}$ for cut selection via the measure,

$$\mathcal{C}^{(0)}(\rho) = \begin{cases} \mathcal{C}(\rho) \text{ (as for QP with } Q \leftarrow Q^{(0)}), \text{ if } \rho \in (\mathcal{P}_n^{E_m} \cap \mathcal{P}_n^{E_0}) \text{ (objective structure)} \\ -\lambda_{\min}(\rho), \text{ if } \rho \in (\mathcal{P}_n^{E_m} \setminus \mathcal{P}_n^{E_0}) \text{ (constraint structure).} \end{cases} \tag{$\mathcal{C}^0(\rho)$}$$

The computational cost of cut selection based on the $\mathcal{C}^{(0)}$ measure falls in the same cases as for $\mathcal{C}$ (see Definition 5.2.1). □

Fig. 15 shows that the combined measure $\mathcal{C}^{(0)}$, using only the objective structure for optimality-based cut selection, outperforms feasibility-based cut selection on illustrative medium-sized QCQP instances [12] with different sparsity and constraint structures. As expected, existing quadratic objective structure in $Q^{(0)}$, i.e. $E_0 \neq \emptyset$, directly transfers the advantages of optimality-based cut selection seen for QP instances to QCQP instances. Github [9] also shows full results for all 320 quadratic instances from [12], supporting the same conclusions.

However, the advantages of cut selection via $\mathcal{C}^{(0)}$ versus the feasibility measure diminish with relatively less objective structure, e.g. $|\mathcal{P}_n^{E_m} \cap \mathcal{P}_n^{E_0}|$ close to 0 or $|E_m| \gg |E_0|$. In such cases, some constraint structure must be explicitly taken into account by an optimality-based measure on elements of $(\mathcal{P}_n^{E_m} \setminus \mathcal{P}_n^{E_0})$. Doing so requires relating constraints to optimality, for example:

- Explicitly reformulating constraint structure into the objective for QCQP instances where the standard form of the relaxation allows so.
- Using the Lagrangian as the objective to incorporate constraint structure (see Appendix A.4).

QCQP instances with little to no quadratic structure in the objective require further research to relate constraints to optimality. But this section clearly shows the potential of cuts selected via optimality measures in the context of QCQP, with applications to a wide array of practical instances.

7 Conclusion

Cuts enforcing a semidefiniteness constraint, e.g. the eigenvalue cuts considered in this paper, are well-known in the literature. But our work, for both QP and QCQP, shows that, inside the

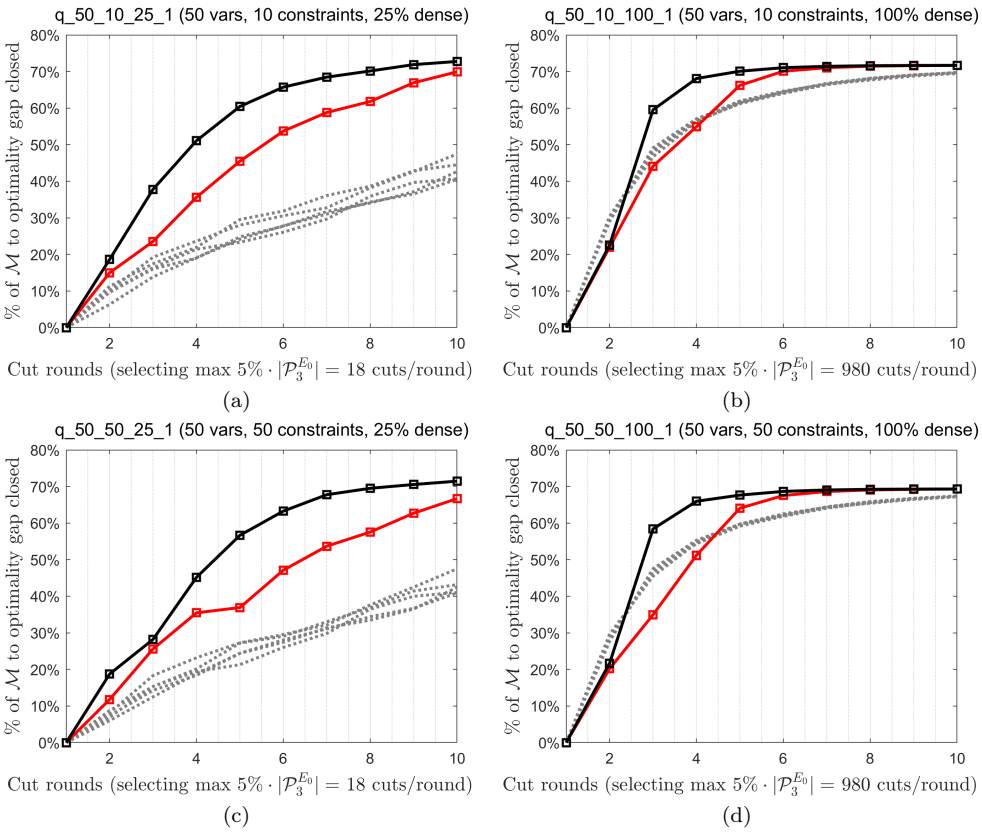


Fig. 15: Comparison of cut selection strategies in terms of % gap closed between the $\mathcal{M}$ relaxation and the optimal solution for QCQP instances from [12] with different characteristics. For each strategy, Algorithm 1 adds to $\mathcal{M}$ every round, given the rounds' $\mathcal{P}_3^{E_m}$ ordering, any feasibility-based, low-dimensional EigCut(ρ) for ρ within the top 5% · $|\mathcal{P}_3^{E_0}|$ selection.

set of all possible *feasibility-based* cuts, there exists a set of fewer, *optimality-based* cuts that tightly outer-approximate the semidefinite constraint(s). Most of this paper discusses quadratic programming, but we show our ideas also extend to quadratic constraints. This paper began with four desiderata and we motivate their fulfillment as follows:

1. **Easy integration into current technology**, e.g. we develop linear cutting planes with few nonzeros in each row. Lemma 3.2.3(ii) motivates that complementary cutting surfaces, e.g. polyhedral cuts, work in synergy with the cuts we propose. Section 5.4 discusses our computational experience with QP instances.
2. **Computationally light relaxations**, e.g. Section 5.3.1 shows the effectiveness on QP of integrating computationally cheap (low-dimension) cutting planes (Remark 3.1.4) with a neural network estimator (Section 4). The neural network takes the cut selection offline. Section 5.3 discusses the engineering trade-offs associated with cut dimensionality and chordal extensions.

3. **Relatively few cuts**, e.g. our search for cutting planes that will most improve the objective value leads to the performance in Fig. 9 and Fig. 14. We expect relatively few cuts because of our emphasis on finding *optimality-based* cuts within the set of *feasibility-based* cuts.
4. **Cutting surface generation should converge**, e.g. our combined feasibility/optimality strategy in Table 3 incorporates all feasibility-based cutting planes while favoring the optimality-based cuts. Figure 11 and Figure 9 show that we have maintained convergence without giving up the fast initial convergence.

With respect to previous work, our proposed cuts are (i) sparser with respect to the number of nonzeros in the row and (ii) explicitly selected to improve the objective. A neural network estimator is key to this cut selection strategy. Building the estimator was a challenge comprising (i) estimator output choice, e.g. the metric for cut selection, (ii) data generation, i.e. sampling uniformly, (iii) model choice, e.g. variance versus bias trade-off, (iv) feature selection, i.e. understanding the problem space, (v) problem decomposition, i.e. to escape the curse of dimensionality. Many have observed that estimators may accurately guess good decisions to make within a Branch&Cut framework [2, 16, 19, 54, 61, 67, 69, 73, 88, 90], so our contributions in building this neural network estimator may be more broadly applicable. For example, we conjecture that a similar estimator could be used for cuts based on copositive optimization, e.g. with 3D relaxations [7] or for the pooling problem which is known to have strong cuts [8, 70]. Other cut selection features common in the MIP literature, e.g. orthogonality [5], could further refine our cut selection strategy.

A Appendices

A.1 Proof of Lemma 3.2.3

Proof (i) Consider any point $\tilde{v}_C \notin \mathcal{S}^{\tilde{v}}$. Via the separation theorem [15], $\exists q \in \mathbb{R}^{|C|}$ determining one RHS cut that separates $\tilde{v}_C$ from the bounded closed set $\mathcal{S}^{\tilde{v}}$. Collectively, such cuts describe the full boundary of $\mathcal{S}^{\tilde{v}}$, determining it.

(ii) Assume fixed $q \in \mathbb{R}^{|C|}$, any $\mathcal{B}' \subseteq \mathcal{B}$ and that all fixings $v = \tilde{v}$ allow PSD completion of M. Then, denoting $\mathcal{S}(\mathcal{B}) = (\mathcal{B} \cap \mathcal{S})$, $\mathcal{S}^q(\mathcal{B}) = (\mathcal{B} \cap \mathcal{S})^q = \mathcal{B} \cap \left\{ M \mid \forall \tilde{v} :\ q \cdot v_C \geq q \cdot v_C^* = \min_{v_C \in \mathcal{S}^{\tilde{v}}} \{q \cdot v_C\} \right\}$ we have

$$\mathcal{S}(\mathcal{B}) = \bigcup_{\forall \tilde{v}} \mathcal{S}(\mathcal{B}^{\tilde{v}}) \overset{\text{(i)}}{\subseteq} \bigcup_{\forall \tilde{v}} \mathcal{S}^q(\mathcal{B}^{\tilde{v}}) = \mathcal{S}^q(\mathcal{B}), \text{ and}$$

$$\begin{aligned} \mathcal{S}^q(\mathcal{B}') \setminus \mathcal{S}(\mathcal{B}') &= \Big(\mathcal{S}^q(\mathcal{B}) \setminus \mathcal{S}^q(\mathcal{B} \setminus \mathcal{B}')\Big) \setminus \Big(\mathcal{S}(\mathcal{B}) \setminus \mathcal{S}(\mathcal{B} \setminus \mathcal{B}')\Big) \\ &= \Big(\mathcal{S}^q(\mathcal{B}) \setminus \mathcal{S}(\mathcal{B})\Big) \setminus \Big(\mathcal{S}^q(\mathcal{B} \setminus \mathcal{B}') \setminus \mathcal{S}(\mathcal{B} \setminus \mathcal{B}')\Big) \subseteq \left(\mathcal{S}^q(\mathcal{B}) \setminus \mathcal{S}(\mathcal{B})\right). \end{aligned}$$

Fixing an extra variable on top of fixing f variables via v is analogous to restricting $\mathcal{B}$ with an extra variable equality. Then, marking the dependence of $(\mathcal{B} \cap \mathcal{S}), (\mathcal{B} \cap \mathcal{S})^q$ on f as a parameter, the above translates to $\Big((\mathcal{B} \cap \mathcal{S})^q_{f+1} \setminus (\mathcal{B} \cap \mathcal{S})_{(f+1)}\Big) \subseteq \Big((\mathcal{B} \cap \mathcal{S})^q_f \setminus (\mathcal{B} \cap \mathcal{S})_f\Big)$.

□

A.2 Proof of Corollary 3.2.4

Proof Setting $x = \tilde{x}$ fixes the linear objective expression $c^T\tilde{x}$ and makes the constraints $A\tilde{x} \leq b,\ \tilde{x} \in [0,1]^N$ redundant. Then,

$$z_{qp}(\mathcal{B}^{\tilde{x}} \cap \mathcal{S}^{\tilde{x}}(\mathcal{V})) - c^T\tilde{x} = \min_{X \in \mathcal{B}^{\tilde{x}}} \left\{ f(X) \;\middle|\; \forall \rho \in \mathcal{V} :\ \begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & X_\rho \end{bmatrix} \succeq 0,\ X_{ii} \leq \tilde{x}_i\ \forall i \in \rho \right\}$$

$$= \min_{X \in \mathcal{B}^{\tilde{x}}} \left\{ f(X) \;\middle|\; \forall \rho \in \mathcal{V},\ \forall Q'_\rho \in \mathbb{R}^{n \times n} :\ Q'_\rho \bullet X_\rho \geq \min_{X_\rho} \left\{ Q'_\rho \bullet X_\rho \;\middle|\; \begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & X_\rho \end{bmatrix} \succeq 0,\ X_{ii} \leq \tilde{x}_i\ \forall i \in \rho \right\} \right\} \tag{5}$$

$$\geq \min_{X \in \mathcal{B}^{\tilde{x}}} \{ f(X) \mid \forall \rho \in \mathcal{V} :\ f(X_\rho) \geq f(X^*_\rho | \tilde{x}_\rho) \}, \tag{6}$$

where:

- Eq. (5) follows from Lemma 3.2.3(i) with: $M \leftarrow \begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & X_\rho \end{bmatrix}$ admitting PSD completion at each fixed $\tilde{x}_\rho$ point; $v_C \leftarrow \text{vec}(X_\rho)$ and corresponding v and fixing pattern; $\mathcal{S} \leftarrow \mathcal{S}(\{\rho\})$.
- Eq. (6) chooses only optimality-based cuts matching the QP objective e.g. $(\forall \rho \in \mathcal{V})\ Q'_\rho = Q_\rho$.

Consequently,

$$\mathcal{B} \cap \mathcal{S}(\mathcal{V}) = \bigcup_{\tilde{x}} \mathcal{B}^{\tilde{x}} \cap \mathcal{S}^{\tilde{x}}(\mathcal{V}) \supseteq \bigcup_{\tilde{x}} \mathcal{B}^{\tilde{x}} \cap \{ \forall \rho \in \mathcal{V} :\ f(X_\rho) \geq f(X^*_\rho | \tilde{x}_\rho) \} \Rightarrow z_{qp}(\mathcal{B} \cap \mathcal{S}(\mathcal{V})) \geq z^L_{qp}(\mathcal{B} \cap \mathcal{S}(\mathcal{V})).$$

□

A.3 Neural network training choices

To train estimators for $f^*_n\ \forall n \in \{2, 3, 4, 5\}$, the following neural architecture and training choices are made:

- Topology (by experimentation): $3 - 4$ hidden layers, each with $\approx 50 - 64$ neurons;
- Activation function in the input/hidden layers: *tanh*;
- Learning algorithm (back-propagation): scaled conjugate gradient;
- Performance function: mean squared error;
- 1 million sampled data points randomly partitioned for training, validation and testing in proportions 75%/ 15%/ 10%, respectively;
- Weights initialization: Nguyen-Widrow (Matlab Neural Network Toolbox default);
- Stopping criteria: gradient/performance thresholds, validation set performance;
- Training time: $\approx 5 - 20$ hours on an Intel i7-4770 CPU.

A.4 A Lagrangian Approach for QCQP Optimality-Based Cut Selection

A natural way to relate constraints to optimality in the QCQP extended formulation is via the Lagrangian function. The Lagrangian of the QCQP relaxation is a proxy for the QP relaxation objective and fixing $x = \tilde{x}$ as in Corollary 3.2.4 reduces it to a quadratic function of variables X and multipliers/duals $\beta^{(m)} \in \mathbb{R}^m_+$,

$$\mathcal{L}(X, \beta | \tilde{x}) := Q^{(0)} \bullet X + \sum_{k=1}^{m} \beta^{(m)}_k (Q^{(k)} \bullet X) = \sum_{k=0}^{m} \beta_k (Q^{(k)} \bullet X),$$

where, for notation convenience, the vector $\beta \in \mathbb{R}^{m+1}_+$ extends $\beta^{(m)}$ with the element 1 to accommodate the objective in the summation. Applying Corollary 3.2.4 to obtain implied optimality-based cuts from $\mathcal{L}(X, \beta | \tilde{x})$ for any low-dimensional vertex cover $\mathcal{V}$, results in subproblems $\rho \in \mathcal{V}$ similar to $P^{\tilde{x}}(\rho)$, but with a bilinear objective in variables X_ρ and β, i.e.

$$\mathcal{L}_0(\rho) := \mathcal{L}(X^*_\rho, \beta^* | \tilde{x}_\rho) = \min_{X_\rho,\ \beta_\rho} \left\{ \sum_{k \in \Theta_\rho} \beta_k Q^{(k)}_\rho \bullet X_\rho \;\middle|\; \begin{bmatrix} 1 & \tilde{x}_\rho^T \\ \tilde{x}_\rho & X_\rho \end{bmatrix} \succeq 0,\ X_{ii} \leq \tilde{x}_i\ \forall i \in \rho \right\}$$

$$\text{where } \Theta_\rho := \{ k \in 0, \ldots, m \mid \forall i, j \in \rho,\ i > j : Q^{(k)}_{ij} \neq 0 \}.$$

Estimating $\mathcal{L}_0(\rho)$ involves learning a collection of nonconvex surfaces corresponding to a SDP problem with bilinear objective. Therefore, the training, architecture and evaluation of a $\mathcal{L}_0(\rho)$ estimator is more complex and

costly than for the Section 4 neural nets that learn a collection of convex surfaces. However, by fixing $\beta = \tilde{\beta}$, we can construct linear ρ subproblems such as,

$$\mathcal{L}_1(\rho) := \mathcal{L}(X^*_\rho|\tilde{\beta}, \tilde{x}_\rho) = \min_{X_\rho} \left\{ \sum_{k\in\Theta_\rho} \tilde{\beta}_k Q^{(k)}_\rho \bullet X_\rho \;\middle|\; \begin{bmatrix} 1 & \tilde{x}^T_\rho \\ \tilde{x}_\rho & X_\rho \end{bmatrix} \succeq 0,\ X_{ii} \le \tilde{x}_i\ \forall i \in \rho \right\},$$

$$\mathcal{L}_2(\rho) := \sum_{k\in\Theta_\rho} \tilde{\beta}_k \min_{X_\rho} \left\{ Q^{(k)}_\rho \bullet X_\rho \;\middle|\; (\tilde{x}_\rho, X_\rho) \in \mathcal{S}^{\tilde{x}}(\rho) \right\}.$$

Furthermore, given an LP solution point $(\tilde{X}, \tilde{\beta}, \tilde{x})$, $\forall \rho \in \mathcal{V}$ an expected objective improvement measure for optimality selection such as $\hat{\mathcal{I}}_X(\rho)$ in Section 3.3 can be built for each $\mathcal{L}_i(\rho)$ $\forall i \in \{0,1,2\}$ in the form

$$\hat{\mathcal{L}}_i(\rho) - \mathcal{L}(\tilde{X}_\rho, \tilde{\beta}, \tilde{x}_\rho) = \hat{\mathcal{L}}_i(\rho) - \sum_{k\in\Theta_\rho} \beta_k (Q^{(k)}_\rho \bullet \tilde{X}_\rho), \qquad (\hat{\mathcal{I}}^{(i)}_X(\rho))$$

where $\forall i \in \{0,1,2\}$ $\hat{\mathcal{L}}_i(\rho)$ estimates $\mathcal{L}_i(\rho)$. Fixing $\tilde{\beta}$ implies $\forall \rho \in \mathcal{V}$ $\hat{\mathcal{L}}_1(\rho) \ge \hat{\mathcal{L}}_0(\rho)$, or $\hat{\mathcal{I}}^{(1)}_X(\rho) \ge \hat{\mathcal{I}}^{(0)}_X(\rho)$. Each optimality measure, when estimated by neural nets, presents computational and bound strength tradeoffs:

- $\hat{\mathcal{I}}^{(0)}_X(\rho)$ requires one evaluation of a neural net for semidefinite subproblems with bilinear objective, but leads to worse final bounds than cut selection based on $\hat{\mathcal{I}}^{(1)}_X(\rho)$ $\left(\hat{\mathcal{I}}^{(1)}_X(\rho) \ge \hat{\mathcal{I}}^{(0)}_X(\rho)\right)$;
- $\hat{\mathcal{I}}^{(1)}_X(\rho)$ **(aggregating all constraints)** requires one evaluation of a neural net for semidefinite subproblems with linear objective, built in Section 4.
- $\hat{\mathcal{I}}^{(2)}_X(\rho)$ **(constraint-by-constraint)** requires $|\Theta(\rho)|$ evaluations of a neural net from Section 4, where $|\Theta(\rho)|$ is high if the variable structures in ρ are heavily repeated though many constraints.

Therefore, unlike $\hat{\mathcal{I}}^{(0)}_X(\rho)$, $\hat{\mathcal{I}}^{(1)}_X(\rho)$ and $\hat{\mathcal{I}}^{(2)}_X(\rho)$ avoid: (i) extra model complexity (the trained neural nets from Section 4 can be reused) and (ii) dual variables β that can not be included in a generated cut for the primal. However, due to fixing $\tilde{\beta}$ at every cut round, an optimality-only selection based on Algorithm 1 with adapted optimality measures $\hat{\mathcal{I}}^{(1)}_X$, $\hat{\mathcal{I}}^{(2)}_X$ is likely to not converge on some instances. The underlying justification is that certain β fixings can nullify any constraints associated to positive optimality measures. To ensure convergence, a combined (optimality and feasibility) ordering similar to $\mathcal{C}(\rho)$ in Section 5.2,

$$i \in \{1,2\},\ \forall \rho \in \mathcal{V},\ \mathcal{C}^{(i)}(\rho) = \begin{cases} \hat{\mathcal{I}}^{(i)}_X(\rho) + M, & \text{if } \hat{\mathcal{I}}^{(i)}_X(\rho) > 0 \text{ and } \lambda_{\min}(\rho) < 0, \\ -\lambda_{\min}(\rho) & \text{otherwise,} \end{cases} \qquad (\mathcal{C}^{(i)}(\rho))$$

where M is an arbitrary large positive number.

Whether cut selection based on $\mathcal{C}^{(1)}(\rho)$ or $\mathcal{C}^{(2)}(\rho)$ outperforms feasibility cut selection, however, depends on the evolution of dual values $\tilde{\beta}$ at each cut round. Ideally, the *local* optimality structure at $\tilde{\beta}$ that $\mathcal{C}^{(1)}(\rho)$ and $\mathcal{C}^{(2)}(\rho)$ exploit needs to remain relevant throughout cut rounds, despite or via $\tilde{\beta}$ changes. The conditions in which this happens require further investigation outside the paper's scope.

References

1. T. Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1): 1–41, 2009.
2. A. M. Alvarez, Q. Louveaux, and L. Wehenkel. A machine learning-based approximation of strong branching. *INFORMS Journal on Computing*, 29(1):185–195, 2017.
3. M. S. Andersen and L. Vandenberghe. Chompack: a python package for chordal matrix computations, 2015.
4. E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, third edition, 1999.
5. G. Andreello, A. Caprara, and M. Fischetti. Embedding {0, 1/2}-cuts in a branch-and-cut framework: A computational study. *INFORMS Journal on Computing*, 19(2):229–238, 2007.
6. K. M. Anstreicher. Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. *Journal of Global Optimization*, 43(2-3):471 – 484, 2009.
7. K. M. Anstreicher and S. Burer. Computable representations for convex hulls of low-dimensional quadratic forms. *Mathematical Programming*, 124(1-2):33–43, 2010.
8. R. Baltean-Lugojan and R. Misener. Piecewise parametric structure in the pooling problem: from sparse strongly-polynomial solutions to NP-hardness. *Journal of Global Optimization*, 71(4):655–690, 2018.

9. R. Baltean-Lugojan and R. Misener. Implementation to select cutting planes for quadratic semidefinite outer-approximation via trained neural networks, 2018. URL https://github.com/cog-imperial/SDPCutSel-via-NN/.
10. X. Bao, N. V. Sahinidis, and M. Tawarmalani. Multiterm polyhedral relaxations for nonconvex, quadratically-constrained quadratic programs. *Optimization Methods and Software*, 24(4-5):485 – 504, 2009.
11. X. Bao, N. V. Sahinidis, and M. Tawarmalani. Semidefinite relaxations for quadratically constrained quadratic programming: A review and comparisons. *Mathematical Programming*, 129(1):129–157, 2011.
12. X. Bao, A. Khajavirad, N. V. Sahinidis, and M. Tawarmalani. Global optimization of nonconvex problems with multilinear intermediates. *Mathematical Programming Computation*, 7(1):1–37, 2015.
13. W. W. Barrett, C. R. Johnson, and R. Loewy. Critical graphs for the positive definite completion problem. *SIAM Journal on Matrix Analysis and Applications*, 20(1):117–130, 1998.
14. P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4-5):597–634, 2009.
15. A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2. Siam, 2001.
16. Y. Bengio, A. Lodi, and A. Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *arXiv preprint arXiv:1811.06128*, 2018.
17. A. Billionnet and S. Elloumi. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. *Mathematical Programming*, 109(1):55–68, 2007.
18. P. Bonami, O. Günlük, and J. Linderoth. Globally solving nonconvex quadratic programming problems with box constraints via integer programming methods. *Mathematical Programming Computation*, 10(3): 333–382, 2018.
19. P. Bonami, A. Lodi, and G. Zarpellon. Learning a classification of mixed-integer quadratic programming problems. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 595–604. Springer, 2018.
20. P. Bonami, A. Lodi, J. Schweiger, and A. Tramontani. Solving quadratic programming by cutting planes. *SIAM Journal on Optimization*, 29(2):1076–1105, 2019.
21. B. Borchers and J. E. Mitchell. An improved branch and bound algorithm for mixed integer nonlinear programs. *Computers & Operations Research*, 21(4):359–367, 1994.
22. F. Boukouvala, R. Misener, and C. A. Floudas. Global optimization advances in mixed-integer nonlinear programming, MINLP, and constrained derivative-free optimization, CDFO. *European Journal of Operational Research*, 252(3):701 – 727, 2016.
23. S. Boyd and L. Vandenberghe. Localization and cutting-plane methods. *From Stanford EE 364b lecture notes*, 2007.
24. A. Brandstädt and C. T. Hoàng. On clique separators, nearly chordal graphs, and the maximum weight stable set problem. *Theoretical Computer Science*, 389(1-2):295–306, 2007.
25. C. Buchheim and E. Traversi. Quadratic combinatorial optimization using separable underestimators. *INFORMS Journal on Computing*, 30(3):424–437, 2018.
26. C. Buchheim and A. Wiegele. Semidefinite relaxations for non-convex quadratic mixed-integer programming. *Mathematical Programming*, 141(1):435–452, 2013.
27. S. Burer. Optimizing a polyhedral-semidefinite relaxation of completely positive programs. *Mathematical Programming Computation*, 2(1):1–19, 2010.
28. S. Burer and D. Vandenbussche. A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. *Mathematical Programming*, 113(2):259–282, 2008.
29. S. Burer and D. Vandenbussche. Globally solving box-constrained nonconvex quadratic programs with semidefinite-based finite branch-and-bound. *Computational Optimization & Applications*, 43(2):181–195, 2009.
30. H. Chel, A. Majumder, and D. Nandi. Scaled conjugate gradient algorithm in neural network based approach for handwritten text recognition. In D. Nagamalai, E. Renault, and M. Dhanuskodi, editors, *Trends in Computer Science, Engineering and Information Technology*, pages 196–210, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
31. J. Chen and S. Burer. Globally solving nonconvex quadratic programming problems via completely positive programming. *Mathematical Programming Computation*, 4(1):33–52, 2012.
32. J. Y. Choi, L. M. DeAlba, L. Hogben, M. S. Maxwell, and A. Wangsness. The p0-matrix completion problem. *Electronic Journal of Linear Algebra*, 9(1-20):99–100, 2002.
33. S. Coniglio and M. Tieves. On the generation of cutting planes which maximize the bound improvement. In *International Symposium on Experimental Algorithms*, pages 97–109. Springer, 2015.
34. J. Dahl, L. Vandenberghe, and V. Roychowdhury. Covariance selection for nonchordal graphs via chordal embedding. *Optimization Methods & Software*, 23(4):501–520, 2008.
35. S. S. Dey and M. Molinaro. Theoretical challenges towards cutting-plane selection. *Mathematical Programming*, pages 1–30, 2018.

36. S. S. Dey, M. Molinaro, and Q. Wang. How good are sparse cutting-planes? In *International Conference on Integer Programming and Combinatorial Optimization*, pages 261–272. Springer, 2014.
37. S. S. Dey, A. Iroume, and M. Molinaro. Some lower bounds on sparse outer approximations of polytopes. *Operations Research Letters*, 43(3):323–328, 2015.
38. S. S. Dey, M. Molinaro, and Q. Wang. Analysis of sparse cutting planes for sparse MILPs with applications to stochastic MILPs. *Mathematics of Operations Research*, 43(1):304–332, 2017.
39. H. Dong. Relaxing nonconvex quadratic functions by multiple adaptive diagonal perturbations. *SIAM Journal on Optimization*, 26(3):1962–1985, 2016.
40. H. Dong and N. Krislock. Semidefinite approaches for MIQCP: Convex relaxations and practical methods. In B. Defourny and T. Terlaky, editors, *Modeling and Optimization: Theory and Applications*, pages 49–75, Cham, 2015. Springer International Publishing.
41. R. M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *Journal of Approximation Theory*, 10(3):227–236, 1974.
42. S. Elloumi and A. Lambert. Global solution of non-convex quadratically constrained quadratic programs. *Optimization Methods and Software*, 34(1):98–114, 2019.
43. K. Fujisawa, M. Fukuda, M. Kojima, K. Nakata, and M. Yamashita. *SDPA-C (semidefinite Programming Algorithm-Completion Method). User's Manual-Version 6-10*. Inst. of Technology, 2004.
44. M. Fukuda, M. Kojima, K. Murota, and K. Nakata. Exploiting sparsity in semidefinite programming via matrix completion i: General framework. *SIAM Journal on Optimization*, 11(3):647–674, 2001.
45. F. Furini, E. Traversi, P. Belotti, A. Frangioni, A. Gleixner, N. Gould, L. Liberti, A. Lodi, R. Misener, H. Mittelmann, N. V. Sahinidis, S. Vigerske, and A. Wiegele. QPLIB: a library of quadratic programming instances. *Mathematical Programming Computation*, 11(2):237–265, 2019.
46. 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.
47. K. R. Garren. Bounds for the eigenvalues of a matrix. *NASA Technical Note*, 1968.
48. X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proc. International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.
49. I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*, volume 1. MIT press Cambridge, 2016.
50. R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. Positive definite completions of partial Hermitian matrices. *Linear algebra and its applications*, 58:109–124, 1984.
51. M. Grötschel, M. Jünger, and G. Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations research*, 32(6):1195–1220, 1984.
52. P. M. Gruber. Asymptotic estimates for best and stepwise approximation of convex bodies i. *Forum Mathematicum*, 5(5):281–298, 1993.
53. M. T. Hagan and M. B. Menhaj. Training feedforward networks with the Marquardt algorithm. *IEEE transactions on Neural Networks*, 5(6):989–993, 1994.
54. H. He, H. Daume III, and J. M. Eisner. Learning to search in branch and bound algorithms. In *Advances in neural information processing systems*, pages 3293–3301, 2014.
55. C. Helmberg and F. Rendl. Solving quadratic (0, 1)-problems by semidefinite programs and cutting planes. *Mathematical programming*, 82(3):291–315, 1998.
56. H. Hendel. Adaptive large neighborhood search for mixed integer programming. Technical Report 18-60, ZIB, Takustr. 7, 14195 Berlin, 2018.
57. L. Hogben. Graph theoretic methods for matrix completion problems. *Linear Algebra and Its Applications*, 328(1-3):161–202, 2001.
58. R. B. Holmes. On random correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, 12(2): 239–272, 1991.
59. IBM CPLEX Optimizer, 2019. URL https://www.ibm.com/analytics/cplex-optimizer. Version 12.8.
60. C. R. Johnson and B. K. Kroschel. The combinatorially symmetric P-matrix completion problem. *Electronic Journal of Linear Algebra*, 1(1):5, 1996.
61. E. B. Khalil, P. Le Bodic, L. Song, G. L. Nemhauser, and B. N. Dilkina. Learning to branch in mixed integer programming. In *AAAI*, pages 724–731, 2016.
62. E. B. Khalil, B. Dilkina, G. L. Nemhauser, S. Ahmed, and Y. Shao. Learning to run heuristics in tree search. In *IJCAI*, pages 659–666, 2017.
63. S. Kim and M. Kojima. Exact solutions of some nonconvex quadratic optimization problems via SDP and SOCP relaxations. *Computational Optimization and Applications*, 26(2):143–154, 2003.
64. S. Kim, M. Kojima, and M. Yamashita. Second order cone programming relaxation of a positive semidefinite constraint. *Optimization Methods and Software*, 18(5):535–541, 2003.
65. N. Krislock, J. Malick, and F. Roupin. Improved semidefinite bounding procedure for solving max-cut problems to optimality. *Mathematical Programming*, 143(1-2):61–86, 2014.
66. Y. A. LeCun, L. Bottou, G. B. Orr, and K. R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

67. L. HS Lelis, L. Otten, and R. Dechter. Predicting the size of depth-first branch and bound search trees. In *IJCAI*, pages 594–600, 2013.
68. S. Leyffer. Integrating SQP and branch-and-bound for mixed integer nonlinear programming. *Computational optimization and applications*, 18(3):295–309, 2001.
69. A. Lodi and G. Zarpellon. On learning and branching: a survey. *TOP*, 25(2):207–236, 2017.
70. J. Luedtke, C. D'Ambrosio, J. Linderoth, and J. Schweiger. Strong convex nonlinear relaxations of the pooling problem. *arXiv preprint arXiv:1803.02955*, 2018.
71. A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. *Proc. International Conference on Machine Learning*, 30(1):3, 2013.
72. J. Malick and F. Roupin. On the bridge between combinatorial optimization and nonlinear optimization: a family of semidefinite bounds for 0–1 quadratic problems leading to quasi-Newton methods. *Mathematical Programming*, 140(1):99–124, 2013.
73. A. Marcos Alvarez, L. Wehenkel, and Q. Louveaux. Machine learning to balance the load in parallel branch-and-bound. *Technical Report, Université de Liège*, 2015.
74. A. Marcos Alvarez, L. Wehenkel, and Q. Louveaux. Online learning for strong branching approximation in branch-and-bound. *Technical Report, Université de Liège*, 2016.
75. G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part 1-convex underestimating problems. *Mathematical Programming*, 10(1):147 – 175, 1976.
76. F. Mezzadri. How to generate random matrices from the classical compact groups. *arXiv preprint math-ph/0609050*, 2006.
77. F. P. Miller, A. F. Vandome, and J. McBrewster. *Amazon web services*. Alpha Press, 2010.
78. R. Misener and C. A. Floudas. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming B*, 136:155–182, 2012.
79. R. Misener and C. A. Floudas. Global optimization of mixed-integer models with quadratic and signomial functions: A review. *Applied and Computational Mathematics*, 11(3):317–336, 2012.
80. R. Misener and C. A. Floudas. GloMIQO: Global Mixed-Integer Quadratic Optimizer. *Journal of Global Optimization*, 57(1):3–50, 2013.
81. R. Misener and C. A. Floudas. ANTIGONE: Algorithms for coNTinuous Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 59(2-3):503–526, 2014.
82. R. Misener, J. B. Smadbeck, and C. A. Floudas. Dynamically generated cutting planes for mixed-integer quadratically constrained quadratic programs and their incorporation into GloMIQO 2. *Optimization Methods and Software*, 30(1):215–249, 2015.
83. S. Mishra, R. Prusty, and P. K. Hota. Analysis of Levenberg-Marquardt and scaled conjugate gradient training algorithms for artificial neural network based LS and MMSE estimated channel equalizers. In *Man and Machine Interfacing (MAMI), 2015 International Conference on*, pages 1–7. IEEE, 2015.
84. M. Mittelbach, B. Matthiesen, and E. A. Jorswieck. Sampling uniformly from the set of positive definite matrices with trace constraint. *IEEE Transactions on Signal Processing*, 60(5):2167–2179, 2012.
85. M. F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4): 525–533, 1993.
86. MOSEK Fusion API for Python., 2019. URL https://docs.mosek.com/8.1/pythonfusion.pdf. Version 8.1.
87. K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota. Exploiting sparsity in semidefinite programming via matrix completion ii: Implementation and numerical results. *Mathematical Programming*, 95 (2):303–327, 2003.
88. G. Nannicini, P. Belotti, J. Lee, J. Linderoth, F. Margot, and A. Wächter. A probing algorithm for MINLP with failure prediction by SVM. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 154–169. Springer, 2011.
89. T. E. Oliphant. Python for scientific computing. *Computing in Science & Engineering*, 9(3), 2007.
90. L. Otten and R. Dechter. And/or branch-and-bound on a computational grid. *Journal of Artificial Intelligence Research*, 59:351–435, 2017.
91. M. Padberg. The Boolean quadric polytope - some characteristics, facets and relatives. *Mathematical Programming*, 45(1):139–172, 1989.
92. M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33(1):60–100, 1991.
93. R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.
94. A. Qualizza, P. Belotti, and F. Margot. Linear programming relaxations of quadratically constrained quadratic programs. In J. Lee and S. Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 407–426. Springer New York, 2012.
95. F. Rendl, G. Rinaldi, and A. Wiegele. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Mathematical Programming*, 121(2):307–335, 2010.

96. A. Sabharwal, H. Samulowitz, and C. Reddy. Guiding combinatorial optimization with UCT. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, pages 356–361. Springer, 2012.
97. N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8(2):201–205, 1996.
98. L. M. Saini and M. K. Soni. Artificial neural network-based peak load forecasting using conjugate gradient methods. *IEEE Transactions on Power Systems*, 17(3):907–912, 2002.
99. Y. Sari. Performance evaluation of the various training algorithms and network topologies in a neural-network-based inverse kinematics solution for robots. *International Journal of Advanced Robotic Systems*, 11(4):64, 2014.
100. A. Saxena, P. Bonami, and J. Lee. Convex relaxations of non-convex mixed integer quadratically constrained programs: extended formulations. *Mathematical Programming*, 124(1-2):383–411, 2010.
101. A. Saxena, P. Bonami, and J. Lee. Convex relaxations of non-convex mixed integer quadratically constrained programs: projected formulations. *Mathematical Programming*, 130(2):359–413, 2011.
102. B. Sharma and K. Venugopalan. Comparison of neural network training functions for hematoma classification in brain CT images. *IOSR Journal of Computer Engineering*, 16:31–35, 2014.
103. H. D. Sherali and W. P. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*. Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, Dordrecht, Netherlands, 1999.
104. H. D. Sherali and B. M. P. Fraticelli. Enhancing RLT relaxations via a new class of semidefinite cuts. *Journal of Global Optimization*, 22(1-4):233–261, 2002.
105. H. D. Sherali, E. Dalkiran, and J. Desai. Enhancing RLT-based relaxations for polynomial programming problems via a new class of v-semidefinite cuts. *Computational Optimization and Applications*, 52(2):483–506, 2012.
106. O. V. Shylo and H. Shams. Boosting binary optimization via binary classification: A case study of job shop scheduling. *arXiv preprint arXiv:1808.10813*, 2018.
107. J. Song, R. Lanka, A. Zhao, Y. Yue, and M. Ono. Learning to search via retrospective imitation. *arXiv preprint arXiv:1804.00846*, 2018.
108. R. Sotirov. SDP relaxations for some combinatorial optimization problems. In *Handbook on Semidefinite, Conic and Polynomial Optimization*, pages 795–819. Springer, 2012.
109. J. F. Sturm, I. Polik, and T. Terlaky. SeDuMi. http://sedumi.ie.lehigh.edu, 2019. Version 1.3.
110. M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103:225–249, 2005.
111. D. Vandenbussche and G. L. Nemhauser. A branch-and-cut algorithm for nonconvex quadratic programs with box constraints. *Mathematical Programming*, 102(3):559–575, 2005.
112. S. A. Vavasis. Quadratic programming is in NP. Technical report, Cornell University, 1990.
113. J. P. Vielma, I. Dunning, J. Huchette, and M. Lubin. Extended formulations in mixed integer conic quadratic programming. *Mathematical Programming Computation*, 9(3):369–418, 2017.
114. S. Vigerske. *Decomposition in Multistage Stochastic Programming and a Constraint Integer Programming Approach to Mixed-Integer Nonlinear Programming*. PhD in Mathematics, Humboldt-University Berlin, 2012.
115. M. Walter. Sparsity of lift-and-project cutting planes. In *Operations Research Proceedings 2012*, pages 9–14. Springer, 2014.
116. F. Wesselmann and U. Stuhl. Implementing cutting plane management and selection techniques. Technical report, University of Paderborn, 2012.
117. X. Zheng, X. Sun, and D. Li. Improving the performance of MIQP solvers for quadratic programs with cardinality and minimum threshold constraints: A semidefinite program approach. *INFORMS Journal on Computing*, 26(4):690–703, 2014.