Disjoint bilinear programming: a two-stage robust optimization perspective

Zhen, J.; Marandi, A.; de Moor, D.; den Hertog, D.; Vandenberghe, L.

DOI
10.1287/ijoc.2022.1163
Publication date
2022
Document Version
Submitted manuscript
Published in
INFORMS Journal on Computing

Submitted to
manuscript (Please, provide the manuscript number!)

Disjoint Bilinear Optimization: A Two-stage Robust Optimization Perspective

Jianzhe Zhen
Department of Information Technology and Electrical Engineering, ETH Zürich, Switzerland

Ahmadreza Marandi*
Department of Industrial Engineering and Innovation Sciences, Eindhoven University of Technology, The Netherlands

Danique de Moor, Dick den Hertog
Department of Operations Management, University of Amsterdam, Amsterdam, The Netherlands

Lieven Vandenberghe
Electrical and Computer Engineering Department, University of California, Los Angeles, USA

In this paper, we focus on a subclass of quadratic optimization problems, that is, disjoint bilinear optimization problems. We first show that disjoint bilinear optimization problems can be cast as two-stage robust linear optimization problems with fixed-recourse and right-hand-side uncertainty, which enables us to apply robust optimization techniques to solve the resulting problems. To this end, a solution scheme based on a blending of three popular robust optimization techniques is proposed. For disjoint bilinear optimization problems with a polyhedral feasible region and a general convex feasible region, we show that under mild regularity conditions, the convex relaxations of the original bilinear formulation and its two-stage robust reformulation obtained from a reformulation-linearization based technique and linear decision rules, respectively, are equivalent. For generic bilinear optimization problems, the convex relaxations from the reformulation-linearization based technique are generally tighter than the one from linear decision rules. Numerical experiments on bimatrix games, synthetic disjoint bilinear problem instances, and convex maximization problems demonstrate the efficiency and effectiveness of the proposed solution scheme.

Key words: bilinear optimization, linear decision rules, reformulation-linearization technique, mixed integer convex optimization.

1. Introduction

In a disjoint bilinear optimization problem, the optimization variables can be partitioned into two vectors, say $\boldsymbol{x}$ and $\boldsymbol{y}$, which reside in two disjoint convex sets $\mathcal{X}$ and $\mathcal{Y}$, respectively. The interaction between $\boldsymbol{x}$ and $\boldsymbol{y}$ is captured through a bilinear objective function, that is, the objective function is linear in $\boldsymbol{x}$ for every $\boldsymbol{y} \in \mathcal{Y}$, and linear in $\boldsymbol{y}$ for every $\boldsymbol{x} \in \mathcal{X}$. Mathematically, a generic disjoint

* Corresponding author.

bilinear optimization problem can be formulated as:

$$\min_{\substack{\boldsymbol{x}\in\mathcal{X}\\ \boldsymbol{y}\in\mathcal{Y}}} \ \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{y}, \tag{BO}$$

where $\mathcal{X}\subseteq\mathbb{R}^{n_x}$, $\mathcal{Y}\subseteq\mathbb{R}^{n_y}$, and $\boldsymbol{Q}\in\mathbb{R}^{n_x\times n_y}$. Note that if one of the feasible sets $\mathcal{X}$ or $\mathcal{Y}$ has only few extreme points, e.g., $\mathcal{Y}$ is simplicial, the optimal solution of (BO) can be efficiently found by enumerating all the vertices. Moveover, if $\boldsymbol{Q}^\top\boldsymbol{x}\in\mathbb{R}^{n_y}$ resides in a specific orthant for all $\boldsymbol{x}\in\mathcal{X}$, and $\mathcal{Y}$ is a hyperbox, the optimal $\boldsymbol{y}$ can be readily determined, which can be used to compute the optimal $\boldsymbol{x}$. Here we assume that the two sets $\mathcal{X}$ and $\mathcal{Y}$ are compact and convex with nonempty relative interior. In general, (BO) is difficult to solve even if $\mathcal{X}$ and $\mathcal{Y}$ are polyhedral (Chen and Deng 2006). We propose to reformulate (BO) into a two-stage robust optimization problem, and investigate the resulting problem from a two-stage robust optimization perspective.

A wide range of real-life applications can be formulated as disjoint bilinear optimization problems. Some possible applications include bilinear assignment problems (Ćustić et al. 2017), fixed charge network flow problems (Rebennack et al. 2009), inverse optimal value problems (Ahmed and Guan 2005), multicommodity flow network interdiction problems (Lim and Smith 2007), Markovian assignment problems (Konno 1971b), and constrained bimatrix games (Firouzbakht et al. 2016, Mangasarian and Stone 1964). Furthermore, any concave or difference-of-convex minimization problem over a convex set can be reformulated as a disjoint bilinear problem (see Section 5.2), e.g., facility location and transportation problems with concave quadratic cost (Soland 1974). Many more applications of (BO) can be found in the PhD thesis by Vaish (1974).

One of the first algorithms to solve (BO) was proposed by Geoffrion (1972), which is known as the generalized Benders decomposition. This algorithm does not guarantee optimality of the found solution in general (Sahinidis and Grossmann 1991). Besides generalized Benders decomposition, there are also many other algorithms proposed to solve (BO); see, e.g., Konno (1971a, 1976), Vaish and Shetty (1976), Gallo and Ülkücü (1977), Sherali and Shetty (1980), Thieu (1988). Sherali and Shetty (1980) and Thieu (1988) proposed finite convergent algorithms for (BO) with polyhedral feasible sets. To the best of our knowledge, there is no finite convergent algorithm for (BO) with a general convex feasible set. For an overview of the solution methods, we refer the reader to the book by Floudas and Pardalos (2008, pp. 279-288) and the references therein. Furthermore, it may be worth to note that since bilinear problems belong to a subclass of quadratic optimization problems, any solution method for nonconvex quadratic optimization problems can be used to solve (BO); see, e.g., Al-Khayyal et al. (1995), Audet et al. (1999), Zheng et al. (2011), Anstreicher (2012), Jiang and Li (2019), Lipp and Boyd (2016), Park and Boyd (2017).

It is known that two-stage robust optimization problems can be reformulated into bilinear optimization problems, e.g., (Zeng and Zhao 2013). In fact, many authors propose to use techniques

for bilinear optimization problems to solve two-stage robust optimization problems (e.g., Zeng and Zhao (2013), Ardestani-Jaafari and Delage (2019), Xu and Burer (2018), Hanasusanto and Kuhn (2018)). In this paper, however, we propose to solve bilinear optimization problems using two-stage robust optimization techniques.

Since the initial introduction of two-stage robust optimization by Ben-Tal et al. (2004), there has been a wealth of practical problems that have been solved using two-stage robust linear optimization such as inventory models (Ben-Tal et al. 2004, 2005), facility location planning (Ardestani-Jaafari and Delage 2017, Atamtürk and Zhang 2007, Gabrel et al. 2014), energy production scheduling (Bertsimas et al. 2013, Ng and Sy 2014), project management (Wiesemann et al. 2012), portfolio optimization (Calafiore 2008, 2009, Rocha and Kuhn 2012), and capacity expansion planning (Ordóñez and Zhao 2007). Two-stage robust optimization problems are in general intractable and NP-hard (Guslitzer 2002). Fortunately, for two-stage robust linear optimization problems with fixed recourse, good solutions can be found using linear decision rules. Rather than allowing the wait-and-see variable to depend arbitrarily on the uncertain parameter, linear decision rules restrict the dependence to be affine. In this way, the resulting problem is again a linear robust optimization problem that can be solved using standard robust optimization techniques, see Ben-Tal et al. (2009). The key benefit is that the problem with linear decision rules is in the same complexity class as the static robust version where all decisions have to be made here-and-now. For some special cases it has been shown that linear decision rules are optimal (Bertsimas et al. 2010, Iancu et al. 2013). There are also several other papers that establish optimality or give theoretical a-priori bounds on the optimal value (Bertsimas and Goyal 2012, Bertsimas and Bidkhori 2015, El Housni and Goyal 2020).

The main contributions of this paper are as follows:

1. We establish a strong relationship between a reformulation-linearization based technique and linear decision rules. In particular, under mild regularity conditions, we show that for (BO) with a convex $\mathcal{X}$ and a polyhedral $\mathcal{Y}$, applying linear decision rules to its two-stage robust optimization reformulation is *equivalent to* applying a reformulation-linearization based technique to the original (BO). This extends the result of Ardestani-Jaafari and Delage (2019, Proposition 8), where the authors focus on two-stage robust optimization problems, and show that applying a reformulation-linearization based technique to the bilinear reformulation is *at least as tight as* applying linear decision rules to the two-stage robust linear optimization problem.

2. We further extend the established result to bilinear constrained bilinear optimization problems, and show that for these problems, the convex relaxations from the reformulation-linearization based technique are generally tighter than the ones from linear decision rules.

3. We propose a solution scheme based on a blending of three robust optimization techniques, those are, linear decision rules, a progressive approximation scheme, and an exact mixed integer reformulation. Thus, the proposed solution scheme consists of three steps. We first solve the convex relaxation of (BO) obtained from linear decision rules, where the optimal objective value constitutes a lower bound of (BO). Then, by leveraging the obtained solution, we retrieve a finite set of binding scenarios. These binding scenarios are in fact feasible solutions of (BO), which are then used to warm-start the mountain climbing procedure and further improve the associated upper bound for (BO). Finally, the exact mixed integer reformulation of (BO) is solved using the best obtained solution so far as a warm-start. Numerical experiments on constrained bimatrix games, synthetic disjoint bilinear problem instances, and convex maximization problems demonstrate the effectiveness and efficiency of the proposed approach in comparison with the state-of-the-art solvers such as Gurobi, SCIP, CPLEX, and MOSEK.

The remainder of this paper is organized as follows. In §2, we first show how to reformulate (BO) to a two-stage robust optimization problem, and then review a robust optimization technique, i.e., linear decision rules. There are three subsections in §3: in §3.1, by leveraging a "primal worst equals dual best" strong duality result of Beck and Ben-Tal (2009) for robust optimization problems, we derive a convex relaxation of (BO) using linear decision rules; in §3.2, a strong relationship between linear decision rules and the relaxation proposed by Ardestani-Jaafari and Delage (2019) is established; in §3.3, we extend the results of §3.2 to bilinear constrained bilinear optimization problems. In §4, we first consider a progressive approximation scheme and an exact reformulation for (BO) in § 4.1 and §4.2, respectively, and then construct a new solution scheme based on the techniques discussed in §4.3. The numerical experiments on constrained bimatrix games, synthetic bilinear problems, and convex maximization problems are conducted in §5.

Notations and definitions. We use $[N]$, $N \in \mathbb{N}$, to denote the set of running indices, $\{1, \ldots, N\}$. We generally use bold faced characters such as $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ to represent vectors and matrices, respectively, x_i to denote the ith element of the vector $\boldsymbol{x}$, $\boldsymbol{A}_i \in \mathbb{R}^M$ to denote the ith column of the matrix $\boldsymbol{A}$, and A_{ij} to denote the entry of $\boldsymbol{A}$ in the ith row and jth column. We use $\lfloor x_i \rceil$ to denote the nearest integer of $x_i \in \mathbb{R}$, and $[-1, 1]^{M \times N}$ to denote the set of M by N matrices with elements that are uniformly drawn from the interval $[-1, 1]$. Special vectors include $\mathbf{0}$, $\mathbf{1}$, and $\boldsymbol{e}_i$ which are respectively the vector of zeros, the vector of ones, and the standard unit basis vector. We denote $\mathcal{R}^{N,M}$ ($\mathcal{R}^{N,M}_+$) as the space of all measurable functions from $\mathbb{R}^N$ to $\mathbb{R}^M$ ($\mathbb{R}^M_+$) that are bounded on compact sets. For $\boldsymbol{z} \in \mathcal{R}^{N,M}$ and $l \in [M]$, we denote by $\boldsymbol{z}_{\setminus\{l\}}$ the function $[z_1, ..., z_{l-1}, z_{l+1}, ..., z_M] \in \mathcal{R}^{N,M-1}$. We denote $\mathbb{S}^{n \times n}$ as the set of all symmetric matrices.

Definition 1 (Perspective Function) *The perspective* $f^{\text{per}}:\mathbb{R}^{n_x}\times\mathbb{R}\to[-\infty,+\infty]$ *of a function* $f:\mathbb{R}^{n_x}\to[-\infty,+\infty]$ *is defined by:*

$$f^{\text{per}}(\boldsymbol{x},t)=\begin{cases} tf(\boldsymbol{x}/t) & \text{if } t>0\\ \delta^*_{\text{dom}(f^*)}(\boldsymbol{x}) & \text{if } t=0\\ \infty & \text{otherwise.}\end{cases}$$

The perspective function is proper, convex and closed if and only if f is proper, convex and closed (see, e.g., Rockafellar (1970)). We refer to Combettes (2018) for the definition and properties of perspective functions.

One may expect computational challenges due to division by zero or near-zero values in perspective functions. Roos et al. (2018) shows that if a convex function $f:\mathbb{R}^n\to\mathbb{R}$ is conic representable, then the perspective of f is conic representable in the same cone. The resulting conic optimization problems can then be solved using existing solvers, e.g., MOSEK.

2. Bilinear problems and two-stage robust optimization

We focus on problem (BO) in which at least one of the two feasible sets $\mathcal{X}$ and $\mathcal{Y}$ is polyhedral. §2.1 presents a two-stage robust linear reformulation of (BO). One of the most prominent robust optimization techniques, i.e., linear decision rules, is discussed in §2.2.

2.1. Reformulation to two-stage robust linear problems

We first show that (BO) with a general convex set $\mathcal{X}$ and a polyhedral $\mathcal{Y}$ can be equivalently reformulated into a two-stage robust linear optimization problem.

PROPOSITION 1. *If* $\mathcal{Y}=\left\{\boldsymbol{y}\in\mathbb{R}^{n_y}_+ \;\middle|\; \boldsymbol{A}_y\boldsymbol{y}\geq\boldsymbol{b}_y\right\}$ *is nonempty, where* $\boldsymbol{A}_y\in\mathbb{R}^{m_y\times n_y}$ *and* $\boldsymbol{b}_y\in\mathbb{R}^{m_y}$*, then Problem* (BO) *has the same optimal value as:*

$$\max_{\tau}\ \left\{\tau \;\middle|\; \exists\boldsymbol{z}\in\mathcal{R}^{n_x,m_y}_+:\tau\leq\boldsymbol{b}_y^\top\boldsymbol{z}(\boldsymbol{x}),\ \boldsymbol{A}_y^\top\boldsymbol{z}(\boldsymbol{x})\leq\boldsymbol{Q}^\top\boldsymbol{x}\ \ \forall\boldsymbol{x}\in\mathcal{X}\ \right\}. \tag{RO}$$

Proof. We first reformulate (BO) into a robust optimization problem with bilinear uncertainty:

$$\max_{\tau}\ \left\{\tau \;\middle|\; \tau\leq\boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{y}\ \ \forall\boldsymbol{x}\in\mathcal{X}\ \ \forall\boldsymbol{y}\in\mathcal{Y}\ \right\}. \tag{1}$$

Using the result of Zhen et al. (2020b, Proposition 1) for robust constraints with bilinear uncertainties, i.e., by dualizing over $\boldsymbol{y}$ in (1), we obtain the equivalent reformulation (RO). □

Problem (RO) is a two-stage robust linear optimization problem with a *fixed recourse* and *right-hand-side* uncertainty. The first-stage or *here-and-now* decision τ is decided before the realization of the uncertain parameter $\boldsymbol{x}$, and the second-stage or *wait-and-see* decision $\boldsymbol{z}$ is determined after the value of $\boldsymbol{x}$ is revealed. The coefficients of $\boldsymbol{z}$ (i.e., $\boldsymbol{b}_y$ and $\boldsymbol{A}_y$) are independent of $\boldsymbol{x}$, which corresponds to the stochastic programming format known as fixed recourse. Two-stage robust

linear optimization problems are generally intractable because the wait-and-see decision is in fact a decision rule, or infinite dimensional variable, instead of a finite vector of decision variables (Ben-Tal et al. 2004).

The main advantage of (1) is that it can be (approximately) solved by any method applicable to two-stage robust linear problems such as linear decision rules (Ben-Tal et al. 2004), piece-wise linear decision rules (Chen et al. 2008, See and Sim 2009), quadratic decision rules (Ben-Tal et al. 2009), polynomial decision rules (Bertsimas et al. 2011), Fourier-Motzkin elimination (Zhen et al. (2017)), finite adaptability approaches (Hanasusanto et al. (2014), Postek and den Hertog (2016), Bertsimas and Dunning (2016), Georghiou et al. (2019), etc.), and copositive programming approaches (Xu and Burer 2018, Hanasusanto and Kuhn 2018).

REMARK 1. Many problems naturally require $\boldsymbol{y} \geq 0$, which is the reason we impose it in Proposition 1. In case the nonnegativity restriction on $\boldsymbol{y}$ is omitted, one will end up with equality constraints in (RO). The usual way of dealing with equality constraints in a two-stage robust problem is by elimination of a wait-and-see decision per equality constraint (Zhen and den Hertog 2017).

REMARK 2. If $\mathcal{X}$ is a hyperbox and $\mathcal{Y}$ is polyhedral, using Jaillet et al. (2016, Proposition 1), one can show that there exists a static decision rule that is optimal for $\boldsymbol{z}$ in (RO) if the elements of each row of $\boldsymbol{Q}$ are all nonnegative or all nonpositive, i.e., $(\boldsymbol{Q}^\top)_i \geq \boldsymbol{0}$ or $-(\boldsymbol{Q}^\top)_i \geq \boldsymbol{0}$ for each $i \in [n_x]$, where $(\boldsymbol{Q}^\top)_i$ denote the i-th row of $\boldsymbol{Q}$. In Appendix A, we show that bilinear problems with quadratic feasible sets can be reformulated as second order cone problems.

2.2. Linear decision rules

In the previous section we have shown that (BO) with polyhedral $\mathcal{Y}$ can be reformulated into the two-stage robust linear optimization problem (RO), which is computationally intractable in general (Guslitzer (2002)). Fortunately, for two-stage robust linear optimization problems with fixed recourse, "good" solutions can be found using simple decision rules. For instance, rather than allowing $\boldsymbol{z}$ in (RO) to depend arbitrarily on the uncertain parameter $\boldsymbol{x}$, linear decision rules restrict the dependence to be affine. In this way, the resulting problem is again a robust linear optimization problem that can be solved using standard robust optimization techniques. By imposing linear decision rules:

$$\mathcal{L}_+^{n_x,m_y} = \left\{ \boldsymbol{z} \in \mathcal{R}_+^{n_x,m_y} \;\middle|\; \begin{array}{l} \exists \boldsymbol{z}^0, \boldsymbol{z}_i^1,\ i \in [n_x]: \\ \boldsymbol{z}(\boldsymbol{x}) = \boldsymbol{z}^0 + \sum\limits_{i \in [n_x]} \boldsymbol{z}_i^1 x_i \end{array} \right\} \subset \mathcal{R}_+^{n_x,m_y}, \tag{2}$$

on the wait-and-see variables $\boldsymbol{z} \in \mathcal{R}_+^{n_x,m_y}$, the following problem yields a lower bound for (RO):

$$\max_{\tau} \ \left\{ \tau \mid \exists \boldsymbol{z} \in \mathcal{L}_+^{n_x,m_y} : \tau \leq \boldsymbol{b}_y^\top \boldsymbol{z}(\boldsymbol{x}),\ \boldsymbol{A}_y^\top \boldsymbol{z}(\boldsymbol{x}) \leq \boldsymbol{Q}^\top \boldsymbol{x}\ \ \forall \boldsymbol{x} \in \mathcal{X} \right\},$$

or equivalently,

$$\max_{\tau,\boldsymbol{z}^0,\{\boldsymbol{z}_i^1\}_{i=1}^{n_x}} \left\{ \tau \;\middle|\; \begin{array}{ll} \tau \le \boldsymbol{b}_y^\top \boldsymbol{z}^0 + \sum_{i\in[n_x]} \boldsymbol{b}_y^\top \boldsymbol{z}_i^1 x_i & \forall \boldsymbol{x}\in\mathcal{X} \\ \boldsymbol{A}_y^\top \boldsymbol{z}^0 + \sum_{i\in[n_x]} \boldsymbol{A}_y^\top \boldsymbol{z}_i^1 x_i \le \boldsymbol{Q}^\top \boldsymbol{x} & \forall \boldsymbol{x}\in\mathcal{X} \\ \boldsymbol{z}^0 + \sum_{i\in[n_x]} \boldsymbol{z}_i^1 x_i \ge \boldsymbol{0} & \forall \boldsymbol{x}\in\mathcal{X} \end{array} \right\}. \tag{RO$_\mathrm{L}$}$$

Since (RO$_\mathrm{L}$) is a static robust linear optimization problem, it admits a tractable robust counterpart for various convex sets $\mathcal{X}$ (Ben-Tal et al. 2015). If the tractable counterpart of (RO$_\mathrm{L}$) is difficult to derive, one can alternatively solve (RO$_\mathrm{L}$) using a cutting-plane method (Bertsimas et al. 2015) or via the dual of (RO$_\mathrm{L}$) (Gorissen et al. 2014). In case $\mathcal{X}$ is simplicial, one can use Bertsimas and Goyal (2012, Theorem 1) to show that there exists a linear decision rule that is optimal for $\boldsymbol{z}$ in (RO), but for this case the original problem (BP) is simple and the optimal solutions can be obtained by enumerating all n_x+1 vertices of the simplex. Bertsimas and Goyal (2012), Bertsimas and Bidkhori (2015) and El Housni and Goyal (2020) give a thorough analysis of the worst-case optimality gap of static and linear decision rules for two-stage robust linear optimization problems with a fixed recourse and right-hand-side uncertainty. Xu and Burer (2018) propose an SDP relaxation of two-stage robust linear optimization problems with right-hand-side uncertainty that is at least as good as linear decision rules.

Besides linear decision rules, more complex decision rules can also be applied to solve (RO) approximately (Chen et al. 2008, See and Sim 2009, Ben-Tal et al. 2009, Bertsimas et al. 2011). For instance, quadratic decision rules can be used to approximately solve (RO) if $\mathcal{X}$ is ellipsoidal, which then can be reformulated as a semi-definite programming problem (Ben-Tal et al. 2009). One can also use Zhen et al. (2017) to establish the optimality of a piecewise affine decision rule for (RO). For example, if $\mathcal{X}$ is a box, there exists a piecewise affine function with only one breakpoint that is optimal for (RO), and the techniques proposed in Gorissen and den Hertog (2013) and Ardestani-Jaafari and Delage (2016) can then be applied to approximate (RO). Unfortunately, even when the structure of optimal decision rules is known, it is often hard to find optimal solutions due to the computational intractability of such decision rules.

3. Interpretation of linear decision rules for bilinear optimization

In this section, we investigate the relationship between linear decision rules and relaxation techniques. In §3.1, we first derive the convex reformulation of (RO$_\mathrm{L}$). The relationship between linear decision rules for (RO) and an existing reformulation-linearization based technique for (BO) is established in § 3.2. We then extend our result to bilinear constrained bilinear problems in §3.3.

3.1. Relationship with convex relaxation methods

In the following theorem, we construct a convex relaxation of (BO), which is in fact the "dual-best" problem of the "primal-worst" of static robust problem (RO$_\mathrm{L}$) in §2.2 .

THEOREM 1. *If the feasible sets* $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^{n_x} \mid \boldsymbol{f}(\boldsymbol{x}) \le \boldsymbol{0}\}$ *and* $\mathcal{Y} = \left\{\boldsymbol{y} \in \mathbb{R}^{n_y}_+ \mid \boldsymbol{A}_y\boldsymbol{y} \ge \boldsymbol{b}_y\right\}$ *are nonempty and compact in* (BO)*, where* $\boldsymbol{f} : \mathbb{R}^{n_x} \to \mathbb{R}^{m_x}$ *is a vector-valued function with element-wise convex functions in* $\boldsymbol{x}$*,* $\boldsymbol{A}_y \in \mathbb{R}^{m_y \times n_y}$*, and* $\boldsymbol{b}_y \in \mathbb{R}^{m_y}$*, then the maximum of* ($\mathrm{RO_L}$) *coincides with the minimum of the following convex optimization problem:*

$$\begin{aligned}
\min_{\boldsymbol{y}\in\mathbb{R}^{n_y}_+,\boldsymbol{\gamma},\boldsymbol{x}} \quad & \sum_{i\in[n_x]}\sum_{j\in[n_y]} Q_{ij}\gamma_{ij} \\
\text{s.t.} \quad & \boldsymbol{f}(\boldsymbol{x}) \le \boldsymbol{0}, \;\; \boldsymbol{A}_y\boldsymbol{y} \ge \boldsymbol{b}_y \\
& \boldsymbol{f}^{\mathrm{per}}(\boldsymbol{\gamma}_j, y_j) \le \boldsymbol{0} && j \in [n_y] \\
& \boldsymbol{f}^{\mathrm{per}}\left(\left(\boldsymbol{\gamma}\boldsymbol{A}_y^\top\right)_k - (\boldsymbol{b}_y)_k\boldsymbol{x}, (\boldsymbol{A}_y\boldsymbol{y} - \boldsymbol{b}_y)_k\right) \le \boldsymbol{0} && k \in [m_y].
\end{aligned} \tag{CR}$$

Proof. Let us first rewrite ($\mathrm{RO_L}$) into the following maxmin problem:

$$\begin{aligned}
\max_{\boldsymbol{z}^0,\{\boldsymbol{z}_i^1\}_{i=1}^{n_x}} \quad & \min_{\boldsymbol{x}\in\mathcal{X}} \left\{ \boldsymbol{b}_y^\top\boldsymbol{z}^0 + \sum_{i\in[n_x]} \boldsymbol{b}_y^\top\boldsymbol{z}_i^1 x_i \right\} \\
\text{s.t.} \quad & \min_{\{\boldsymbol{x}_j^{con}\}_{j=1}^{n_y}\in\mathcal{X}} \left\{ \boldsymbol{Q}_j^\top\boldsymbol{x}_j^{con} - \sum_{i\in[n_x]} (\boldsymbol{A}_y)_j^\top\boldsymbol{z}_i^1(\boldsymbol{x}_j^{con})_i - (\boldsymbol{A}_y)_j^\top\boldsymbol{z}^0 \right\} \ge 0 \quad j \in [n_y] \;\; : y_j \\
& \min_{\{\boldsymbol{x}_k^{sig}\}_{k=1}^{m_y}\in\mathcal{X}} \left\{ (\boldsymbol{z})_k^0 + \sum_{i\in[n_x]} z_{ki}^1(\boldsymbol{x}_k^{sig})_i \right\} \ge 0 \qquad\qquad k \in [m_y] \;\; : \theta_k,
\end{aligned} \tag{3}$$

where y_j and θ_k are the dual variables of the corresponding constraints in (3). Since ($\mathrm{RO_L}$) has constraint-wise uncertainty, we use $\boldsymbol{x}$, $\boldsymbol{x}^{con}$, and $\boldsymbol{x}^{sig}$, to distinguish the uncertain parameter $\boldsymbol{x}$ from different constraints of ($\mathrm{RO_L}$). Problem (3) is a static robust linear optimization problem, which can be referred to as the "primal-worst" problem. Beck and Ben-Tal (2009) show that strong duality holds if we dualize (3) over $\boldsymbol{z}^0, \boldsymbol{z}_i^1$, $i \in [n_x]$:

$$\begin{aligned}
\min_{\substack{\boldsymbol{y}\in\mathbb{R}^{n_y}_+,\boldsymbol{\theta}\in\mathbb{R}^{m_y}_+,\boldsymbol{x}\in\mathcal{X} \\ \{\boldsymbol{x}_j^{con}\}_{j=1}^{n_y}\in\mathcal{X} \\ \{\boldsymbol{x}_k^{sig}\}_{k=1}^{m_y}\in\mathcal{X}}} \quad & \sum_{j\in[n_y]} y_j\boldsymbol{Q}_j^\top\boldsymbol{x}_j^{con} \\
\text{s.t.} \quad & \boldsymbol{\theta} = \boldsymbol{A}_y\boldsymbol{y} - \boldsymbol{b}_y && : \boldsymbol{z}^0 \\
& (\boldsymbol{b}_y)_k\boldsymbol{x} - \sum_{j\in[n_y]} y_j(\boldsymbol{A}_y)_{kj}\boldsymbol{x}_j^{con} + \theta_k\boldsymbol{x}_k^{sig} = \boldsymbol{0} \quad k \in [m_y] \;\; : \boldsymbol{z}_k.
\end{aligned}$$

This resulting problem is the so-called "dual-best" problem. By eliminating θ_k and $\boldsymbol{x}_k^{sig}$ using the equality constraints, we get:

$$\begin{aligned}
\min_{\substack{\boldsymbol{y}\in\mathbb{R}^{n_y}_+,\boldsymbol{x}\in\mathcal{X} \\ \{\boldsymbol{x}_j^{con}\}_{j=1}^{n_y}\in\mathcal{X}}} \quad & \sum_{j\in[n_y]} y_j\boldsymbol{Q}_j^\top\boldsymbol{x}_j^{con} \\
\text{s.t.} \quad & \boldsymbol{A}_y\boldsymbol{y} \ge \boldsymbol{b}_y \\
& \boldsymbol{f}\left(\frac{\sum_{j\in[n_y]} y_j(\boldsymbol{A}_y)_{kj}\boldsymbol{x}_j^{con} - (\boldsymbol{b}_y)_k\boldsymbol{x}}{(\boldsymbol{A}_y\boldsymbol{y} - \boldsymbol{b}_y)_k}\right) \le 0 \quad k \in [m_y].
\end{aligned}$$

Due to the existence of product of variables, $y_j \boldsymbol{x}_j^{con}$, this problem is in general nonconvex. In the following, we convexify this problem via a simple variable substitution technique proposed in Dantzig (1963) for generalized linear optimization problems. First, we replace the product variable $y_j \boldsymbol{x}_j^{con}$ by $\boldsymbol{\gamma}_j$, and obtain:

$$\begin{aligned}
\min_{\substack{\boldsymbol{y}\in\mathbb{R}_+^{n_y},\boldsymbol{x}\in\mathcal{X},\\ \boldsymbol{\gamma}\in\mathbb{R}^{n_x\times n_y}}} \quad & \sum_{j\in[n_y]} \boldsymbol{Q}_j^\top \boldsymbol{\gamma}_j \\
\text{s.t.} \quad & \boldsymbol{A}_y \boldsymbol{y} \geq \boldsymbol{b}_y \\
& \boldsymbol{f}\left(\frac{\boldsymbol{\gamma}_j}{y_j}\right) \leq 0 \qquad\qquad j\in[n_y] \\
& \boldsymbol{f}\left(\frac{\sum_{j\in[n_y]}(\boldsymbol{A}_y)_{kj}\boldsymbol{\gamma}_j - (\boldsymbol{b}_y)_k \boldsymbol{x}}{(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k}\right) \leq 0 \quad k\in[m_y].
\end{aligned}$$

Since $\boldsymbol{y}\geq \boldsymbol{0}$ and $\boldsymbol{A}_y\boldsymbol{y}\geq \boldsymbol{b}_y$, we finally obtain (CR) by multiplying $\boldsymbol{f}\left(\frac{\boldsymbol{\gamma}_j}{y_j}\right)$ with y_j and $\boldsymbol{f}\left(\frac{\sum_{j\in[n_y]}(\boldsymbol{A}_y)_{kj}\boldsymbol{\gamma}_j-(\boldsymbol{b}_y)_k\boldsymbol{x}}{(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k}\right)$ with $(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k$ in the inequalities for $j\in[n_y]$ and $k\in[m_y]$. Since the perspective function of a convex function is convex (Rockafellar 1970), problem (CR) is indeed a convex optimization problem. □

Note that (CR) is a *convex relaxation* of (BO) in the sense that for any solution $(\boldsymbol{x},\boldsymbol{y})$ feasible to the latter problem, there exists $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma})$ that is a feasible solution to the former problem with the same objective value. However, the converse is not necessarily true, and so, (CR) yields a lower bound for (BO). Convex relaxations via perspective functions are often used to formulate the exact convex hull of a union of convex sets, which is then used as a convex relaxation of disjunctive convex optimization problems (Ceria and Soares 1999, Stursberg and Panek 2002, Grossmann and Lee 2003, Bonami et al. 2015) and mixed integer nonlinear optimization problems (Günlük and Linderoth 2010, Hijazi et al. 2012, Moehle and Boyd 2015). Hijazi et al. (2012) shows that for disjunctive convex optimization problems, convex relaxations via perspective functions are generally tighter than simply relaxing the discrete variables to continuous ones.

Remark 3. Note that (CR) constitutes a convex relaxation even if $\mathcal{X}$ and $\mathcal{Y}$ in Theorem 1 are unbounded, however, the maximum of ($\mathrm{RO_L}$) and the minimum of (CR) may no longer coincide. Thanks to Zhen et al. (2020c, Lemma 6), Theorem 1 remains valid if $\mathcal{X}$ has nonempty relative interior and there exists a feasible $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma})$, where $\boldsymbol{y}$ is in the interior of $\mathcal{Y}$. Here, we assume $\mathcal{Y}$ has no implicit equality because such equalities can be efficiently removed from $\mathcal{Y}$ (Fukuda 2013).

3.2. Relationship with reformulation-linearization techniques

Reformulation-linearization techniques are well-known for deriving convex relaxation of nonconvex optimization problems with product of variables. We refer to Sherali and Liberti (2008) and the references therein for an overview of the existing reformulation-linearization techniques.

In the following example, we show that for (BO) with polyhedral feasible sets $\mathcal{X}$ and $\mathcal{Y}$, the resulting convex relaxation from Theorem 1 coincides with the relaxation obtained by the reformulation-linearization technique (RLT-SA) proposed in Sherali and Alameddine (1992) for bilinear problems.

EXAMPLE 1. Consider (BO) with nonempty polyhedral feasible sets $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^{n_x}_+ \mid \boldsymbol{A}_x\boldsymbol{x} \geq \boldsymbol{b}_x\}$ and $\mathcal{Y} = \left\{\boldsymbol{y} \in \mathbb{R}^{n_y}_+ \mid \boldsymbol{A}_y\boldsymbol{y} \geq \boldsymbol{b}_y\right\}$, where $\boldsymbol{A}_x \in \mathbb{R}^{m_x \times n_x}$, $\boldsymbol{A}_y \in \mathbb{R}^{m_y \times n_y}$, and $\boldsymbol{b}_x \in \mathbb{R}^{m_x}$, $\boldsymbol{b}_y \in \mathbb{R}^{m_y}$. We apply RLT-SA to derive the linear relaxation of (BO). Firstly, by multiplying the nonnegative scalars y_j and $(\boldsymbol{A}_y\boldsymbol{y} - \boldsymbol{b}_y)_k$ on both side of inequalities $\boldsymbol{x} \geq \boldsymbol{0}$ and $\boldsymbol{A}_x\boldsymbol{x} \geq \boldsymbol{b}_x$, we have:

$$\begin{aligned}
\min_{\substack{\boldsymbol{x}\boldsymbol{y}^\top \in \mathbb{R}^{n_x \times n_y}_+ \\ \boldsymbol{x} \in \mathbb{R}^{n_x}_+, \boldsymbol{y} \in \mathbb{R}^{n_x}_+}} \quad & \boldsymbol{x}^\top \boldsymbol{Q} \boldsymbol{y} \\
\text{s.t.} \quad & \boldsymbol{A}_x\boldsymbol{x} \geq \boldsymbol{b}_x, \;\; \boldsymbol{A}_y\boldsymbol{y} \geq \boldsymbol{b}_y \\
& \boldsymbol{A}_x\boldsymbol{x}\boldsymbol{y}^\top \geq \boldsymbol{b}_x\boldsymbol{y}^\top \\
& \boldsymbol{A}_y\boldsymbol{y}\boldsymbol{x}^\top \geq \boldsymbol{b}_y\boldsymbol{x}^\top \\
& (\boldsymbol{A}_x\boldsymbol{x} - \boldsymbol{b}_x)\left(\boldsymbol{A}_y\boldsymbol{y} - \boldsymbol{b}_y\right)^\top \geq \boldsymbol{0}.
\end{aligned} \tag{4}$$

Secondly, by replacing all the products of variables $\boldsymbol{x}y_j$ in (4) by their linear relaxation $\boldsymbol{\gamma}_j \in \mathbb{R}^{n_x}$, we obtain the following convex relaxation:

$$\begin{aligned}
\min_{\substack{\boldsymbol{\gamma} \in \mathbb{R}^{n_x \times n_y}_+ \\ \boldsymbol{x} \in \mathbb{R}^{n_x}_+, \boldsymbol{y} \in \mathbb{R}^{n_y}_+}} \quad & \sum_{i \in [n_x]} \sum_{j \in [n_y]} Q_{ij}\gamma_{ij} \\
\text{s.t.} \quad & \boldsymbol{A}_x\boldsymbol{x} \geq \boldsymbol{b}_x, \;\; \boldsymbol{A}_y\boldsymbol{y} \geq \boldsymbol{b}_y \\
& \boldsymbol{A}_x\boldsymbol{\gamma} \geq \boldsymbol{b}_x\boldsymbol{y}^\top \\
& \boldsymbol{A}_y\boldsymbol{\gamma}^\top \geq \boldsymbol{b}_y\boldsymbol{x}^\top \\
& \boldsymbol{A}_x\boldsymbol{\gamma}\boldsymbol{A}_y^\top \geq \boldsymbol{b}_x\boldsymbol{y}^\top\boldsymbol{A}_y^\top + \boldsymbol{A}_x\boldsymbol{x}\boldsymbol{b}_y^\top - \boldsymbol{b}_x\boldsymbol{b}_y^\top.
\end{aligned} \tag{5}$$

Alternatively, this linear relaxation (5) is the same as (CR) in Theorem 1, i.e., the resulting problem when linear decision rules are used for (BO) with polyhedral feasible sets. It is known (Sherali and Liberti 2008) that relaxation (5) from RLT-SA is in general tighter than the relaxation of (BO) from McCormick envelopes (McCormick 1976). □

For polyhedral feasible sets $\mathcal{X}$ and $\mathcal{Y}$ as defined in Example 1, the variable $\boldsymbol{y}$ in (BO) is dualized first to obtain the two-stage robust reformulation (RO) in §2.1. Alternatively, one can first dualize (BO) over $\boldsymbol{x}$ instead to obtain the following two-stage robust reformulation of (BO):

$$\max_{\tau, \boldsymbol{w}} \quad \left\{\tau \mid \exists \boldsymbol{w} \in \mathcal{R}^{n_y, m_x}_+ : \tau \leq \boldsymbol{b}_x^\top \boldsymbol{w}(\boldsymbol{y}),\; \boldsymbol{A}_x^\top \boldsymbol{w}(\boldsymbol{y}) \leq \boldsymbol{Q}\boldsymbol{y} \;\; \forall \boldsymbol{y} \in \mathcal{Y}\right\}. \tag{RD}$$

Note that (RD) can also be obtained by applying the consecutive dualization to (RO) using Bertsimas and de Ruiter (2016, Theorem 1). Similarly as in the proof of Theorem 1, after imposing linear decision rules to $\boldsymbol{w}$, one can derive the corresponding convex relaxation of (RD). It can be verified

that the obtained convex relaxation is again (5). Hence, we recover Bertsimas and de Ruiter (2016, Theorem 2), where the authors show that solving two-stage robust linear optimization problems and their equivalent dual reformulations via linear decision rules obtain the same optimal value.

In this subsection, we show that one can also derive (CR) from (BO) via a reformulation-linearization based technique. Firstly, by multiplying the nonnegative scalars y_j and $(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k$ on both side of inequalities $f(\boldsymbol{x})\leq \boldsymbol{0}$ in (BO) for $j\in[n_y]$ and $k\in[m_y]$, we introduce redundant constraints (6c) and (6d):

$$\min_{\boldsymbol{y}\in\mathbb{R}^{n_y}_+,\boldsymbol{x}\in\mathbb{R}^{n_x}_+} \quad \boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{y} \tag{6a}$$

$$\text{s.t.} \quad \boldsymbol{f}(\boldsymbol{x})\leq\boldsymbol{0},\ \ \boldsymbol{A}_y\boldsymbol{y}\geq\boldsymbol{b}_y \tag{6b}$$

$$y_j\boldsymbol{f}(\boldsymbol{x})\leq\boldsymbol{0} \qquad j\in[n_y] \tag{6c}$$

$$(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k\boldsymbol{f}(\boldsymbol{x})\leq\boldsymbol{0} \qquad k\in[m_y]. \tag{6d}$$

Secondly, one can then multiply $\boldsymbol{x}$ with y_j/y_j and $(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k/(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k$ in (6c) and (6d) for $j\in[n_y]$ and $k\in[m_y]$, respectively, and replace all the products of variables $\boldsymbol{x}y_j$ in (6) by their linear relaxation $\boldsymbol{\gamma}_j\in\mathbb{R}^{n_x}$, to obtain (CR). Note that the constraints (6c) and (6d) are obviously redundant in (6), but may be non-redundant in the relaxed problem (CR). This reformulation-linearization technique (to which we refer by RLT-AD) was first proposed in Ardestani-Jaafari and Delage (2019) for two-stage robust linear optimization problems, where the authors use convex analysis techniques to show that the convex relaxation from RLT-AD for the bilinear reformulation is *at least as good as* the approximation from using linear decision rules. Theorem 1 and Remark 3 provide regularity conditions under which the convex relaxation of (BO) using RLT-AD has the *same* optimal value as the approximation of (RO) using linear decision rules. The established relationship between RLT-AD and linear decision rules enables us to directly apply existing results from two-stage robust optimization to investigate bilinear problems.

Remark 4. The results of Bertsimas and Goyal (2012), Bertsimas and Bidkhori (2015) and El Housni and Goyal (2020) on the worst-case optimality gap of linear decision rules can be directly applied to evaluate the performance of RLT-AD due to the equivalence between linear decision rules and RLT-AD.

Remark 5. For some special structured uncertainty sets for two-stage robust linear optimization problems, Chen et al. (2008), Chen and Zhang (2009), Bertsimas et al. (2019), and de Ruiter and Ben-Tal (2017) propose to consider a lifted uncertainty set by introducing auxiliary variables

and redundant constraints, and solve two-stage robust linear optimization problems using segregated/extended/enhanced linear decision rules. For example, consider (RO) with an ellipsoidal uncertainty set $\mathcal{X} = \{\boldsymbol{x} \mid ||\boldsymbol{L}\boldsymbol{x}||_2 \leq \rho\}$, which can be simply lifted into (de Ruiter and Ben-Tal 2017):

$$\widehat{\mathcal{X}} = \left\{ (\boldsymbol{x}, \boldsymbol{\kappa}, \boldsymbol{v}) \;\middle|\; |\boldsymbol{L}\boldsymbol{x}| \leq \boldsymbol{\kappa},\ \kappa_i^2 \leq v_i,\ \mathbf{1}^\top \boldsymbol{v} \leq \rho,\ i \in [m_x] \right\},$$

where $\boldsymbol{L} \in \mathbb{R}^{m_x \times n_x}$. The numerical experiments in the existing literature suggest significant improvements can be achieved by using lifted linear decision rules $\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{\kappa}, \boldsymbol{v}) \in \mathcal{L}^{n_x+2m_x, m_y}$ for two-stage robust optimization problems. These existing lifting techniques can be readily adopted by RLT-AD, i.e., the same improvements can be achieved by directly applying RLT-AD to (BO) with the lifted feasible set $\widehat{\mathcal{X}}$. Since these lifting techniques only work for special structured sets, we will not consider them in the numerical experiments.

Remark 6. Note that the number of constraints in (CR) is much larger than in the original (BO). One practical yet effective way of solving (CR) is to use an iterative constraint generation method, that is, first solve a relaxed (CR) with only a subset of constraints, and check whether the obtained solution violates remaining constraints, then, only add (some of) the violating constraints to the problem and solve it. One can repeat this procedure until there is no violating constraint, in which case the obtained solution is optimal for (CR).

3.3. Extension to bilinear constrained bilinear optimization

In this section, we extend RLT-AD to derive convex relaxation for generic bilinear problems with bilinear constraints and non-polyhedral feasible regions, and relate the obtained convex relaxation to the approximation from using a hybrid of static and linear decision rules.

Consider a generalization of (BO) with bilinear constraints and non-polyhedral feasible regions:

$$\begin{aligned} \min_{\substack{\boldsymbol{x}\in\mathcal{X} \\ \boldsymbol{y}\in\mathbb{R}_+^{n_y}}} \quad & \boldsymbol{x}^\top \boldsymbol{Q}^{(0)} \boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{A}_y \boldsymbol{y} \geq \boldsymbol{b}_y,\ \ \boldsymbol{g}(\boldsymbol{y}) \leq \mathbf{0} \\ & \boldsymbol{x}^\top \boldsymbol{Q}^{(t)} \boldsymbol{y} \geq c_t \qquad\qquad t \in [T], \end{aligned} \tag{GBO}$$

where $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}_+^{n_x} \mid \boldsymbol{A}_x \boldsymbol{x} \geq \boldsymbol{b}_x,\ \boldsymbol{f}(\boldsymbol{x}) \leq \mathbf{0}\}$, $\boldsymbol{A}_x \in \mathbb{R}^{m_x \times n_x}$, $\boldsymbol{b}_x \in \mathbb{R}^{m_x}$, $\boldsymbol{A}_y \in \mathbb{R}^{m_y \times n_y}$, $\boldsymbol{b}_y \in \mathbb{R}^{m_y}$, $\boldsymbol{Q}^{(t)} \in \mathbb{R}^{n_x \times n_y}$ for $t = 0, \ldots, T$, and $\boldsymbol{f} : \mathbb{R}^{n_x} \to \mathbb{R}^{m_x}$ and $\boldsymbol{g} : \mathbb{R}^{n_y} \to \mathbb{R}^{m_y}$ are vector-valued functions with element-wise nonlinear convex functions in $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. A plethora of problems can be written into the form of (GBO). Note that in (GBO), the feasible regions of $\boldsymbol{x}$ and $\boldsymbol{y}$ are

neither polyhedral nor disjoint, hence Theorem 1 cannot be applied. Similarly as for (6), one can analogously apply RLT-AD to obtain the following convex relaxation:

$$
\begin{aligned}
\min_{\substack{\boldsymbol{\gamma}\in\mathbb{R}_+^{n_x\times n_y}\\ \boldsymbol{x}\in\mathbb{R}_+^{n_x},\boldsymbol{y}\in\mathbb{R}_+^{n_y}}} \quad & \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} Q_{ij}^{(0)}\gamma_{ij} & \\
\text{s.t.} \quad & \boldsymbol{A}_x\boldsymbol{x}\geq\boldsymbol{b}_x,\ \ \boldsymbol{f}(\boldsymbol{x})\leq\boldsymbol{0},\ \ \boldsymbol{f}^{\text{per}}(\boldsymbol{\gamma}_j,y_j)\leq\boldsymbol{0} & j\in[n_y]\\
& \boldsymbol{A}_y\boldsymbol{y}\geq\boldsymbol{b}_y,\ \ \boldsymbol{g}(\boldsymbol{y})\leq\boldsymbol{0},\ \ \boldsymbol{g}^{\text{per}}(\boldsymbol{\gamma}_i^\top,x_i)\leq\boldsymbol{0} & i\in[n_x]\\
& \boldsymbol{A}_x\boldsymbol{\gamma}\geq\boldsymbol{b}_x\boldsymbol{y}^\top,\ \ \boldsymbol{A}_y\boldsymbol{\gamma}^\top\geq\boldsymbol{b}_y\boldsymbol{x}^\top & \\
& \boldsymbol{A}_x\boldsymbol{\gamma}\boldsymbol{A}_y^\top\geq\boldsymbol{b}_x\boldsymbol{y}^\top\boldsymbol{A}_y^\top+\boldsymbol{A}_x\boldsymbol{x}\boldsymbol{b}_y^\top-\boldsymbol{b}_x\boldsymbol{b}_y^\top & \\
& \boldsymbol{f}^{\text{per}}\left(\left(\boldsymbol{\gamma}\boldsymbol{A}_y^\top\right)_k-(\boldsymbol{b}_y)_k\boldsymbol{x},(\boldsymbol{A}_y\boldsymbol{y}-\boldsymbol{b}_y)_k\right)\leq\boldsymbol{0} & k\in[m_y]\\
& \boldsymbol{g}^{\text{per}}\left((\boldsymbol{A}_x\boldsymbol{\gamma})_l-(\boldsymbol{b}_x)_l\boldsymbol{y},(\boldsymbol{A}_x\boldsymbol{x}-\boldsymbol{b}_x)_l\right)\leq\boldsymbol{0} & l\in[m_x]\\
& \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} Q_{ij}^{(t)}\gamma_{ij}\geq c_t & t\in[T],
\end{aligned}
\tag{GCR}
$$

where the vectors $\boldsymbol{\gamma}_i^\top\in\mathbb{R}^{n_x}$ and $\boldsymbol{\gamma}_j\in\mathbb{R}^{n_y}$ are the i-th row vector and j-th column vector of the matrix $\boldsymbol{\gamma}\in\mathbb{R}^{n_x\times n_y}$, respectively.

In the remainder of this section, we relate the obtained convex relaxation of (GBO) to the approximation of the two-stage reformulation of (GBO) from a hybrid of static and linear decision rules. To this end, we first reformulate (GBO) into a two-stage robust nonlinear problem with non-fixed recourse.

Proposition 2. *If* (GBO) *admits a Slater point, then* (GBO) *has the same optimal value as:*

$$
\max_{\tau}\left\{\tau\ \middle|\ \begin{array}{ll}\exists\boldsymbol{z}\in\mathcal{R}_+^{n_x,n_z}\\ \exists\boldsymbol{u}\in\mathcal{R}_+^{n_x,T}\\ \exists\boldsymbol{\lambda}\in\mathcal{R}_+^{n_x,m_y}\\ \exists\{\boldsymbol{v}_k\}_{k=1}^K\subset\mathcal{R}^{n_x,m_y}\end{array}:\ \begin{array}{l}\tau\leq\boldsymbol{b}_y^\top\boldsymbol{z}(\boldsymbol{x})-\sum_{k=1}^K(g_k^*)^{per}(\boldsymbol{v}_k(\boldsymbol{x}),\lambda_k(\boldsymbol{x}))+\boldsymbol{c}^\top\boldsymbol{u}(\boldsymbol{x})\\ \boldsymbol{A}_y^\top\boldsymbol{z}(\boldsymbol{x})-{\boldsymbol{Q}^{(0)}}^\top\boldsymbol{x}+\sum_{t=1}^T u_t(\boldsymbol{x}){\boldsymbol{Q}^{(k)}}^\top\boldsymbol{x}\leq\sum_{k=1}^K\boldsymbol{v}_k(\boldsymbol{x})\end{array}\ \forall\boldsymbol{x}\in\mathcal{X}\right\}.
\tag{GRO^y}
$$

Proof. We first reformulate (GBO) into a robust optimization problem with bilinear uncertainty:

$$
\max_{\tau}\left\{\tau\ \middle|\ \tau\leq\boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{y}\ \ \forall\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\geq\boldsymbol{0}:\ \begin{array}{ll}\boldsymbol{A}_y\boldsymbol{y}\geq\boldsymbol{b}_y & :\boldsymbol{z}\\ \boldsymbol{g}(\boldsymbol{y})\leq\boldsymbol{0} & :\boldsymbol{\lambda},\ \{\boldsymbol{v}_k\}_{k=1}^K\\ \boldsymbol{x}^\top\boldsymbol{Q}^{(t)}\boldsymbol{y}\geq c_t\ \ t\in[T] & :\boldsymbol{u}\end{array}\right\}.
\tag{7}
$$

We then fix $\boldsymbol{x}\in\mathcal{X}$, and dualize the semi-infite constraint over $\boldsymbol{y}$ using standard robust optimization techniques (Ben-Tal et al. 2015), and obtain (GRO^y). By strong duality for convex optimization problems, which applies because (GBO) admits a Slater point and the embedded minimization problem in the constraint of (7) is convex in $\boldsymbol{y}$ for any fixed $\boldsymbol{x}\in\mathcal{X}$, the optimal values of (7) and (GRO^y) indeed coincide. □

The two-stage robust nonlinear reformulation (GRO^y) has a non-fixed recourse and the constraints are nonlinear in the wait-and-see decision variables. Imposing linear decisions to all the wait-and-see decision variables $(\boldsymbol{z}, \boldsymbol{u}, \boldsymbol{\lambda}, \{\boldsymbol{y}_k\}_k)$ leads to a nonconvex optimization problem. In order to obtain a convex approximation, we propose to use a hybrid of static and linear decision rules, i.e., we impose linear decision rules to $\boldsymbol{z}$ and static decision rules to $(\boldsymbol{u}, \boldsymbol{\lambda}, \{\boldsymbol{y}_k\}_k)$, and obtain:

$$\max_{\tau} \left\{ \tau \;\middle|\; \begin{array}{ll} \exists \boldsymbol{z} \in \mathcal{L}_+^{n_x, n_z} & \\ \exists \boldsymbol{u} \geq \boldsymbol{0} & \\ \exists \boldsymbol{\lambda} \geq \boldsymbol{0} & : \\ \exists \{\boldsymbol{v}_k\}_{k=1}^K & \end{array} \begin{array}{ll} \tau \leq \boldsymbol{b}_y^\top \boldsymbol{z}(\boldsymbol{x}) - \sum_{k=1}^K (g_k^*)^{per}(\boldsymbol{v}_k, \lambda_k) + \boldsymbol{c}^\top \boldsymbol{u} & \forall \boldsymbol{x} \in \mathcal{X} \\ \boldsymbol{A}_y^\top \boldsymbol{z}(\boldsymbol{x}) - {\boldsymbol{Q}^{(0)}}^\top \boldsymbol{x} + \sum_{t=1}^T u_t {\boldsymbol{Q}^{(t)}}^\top \boldsymbol{x} \leq \sum_{k=1}^K \boldsymbol{v}_k & \forall \boldsymbol{x} \in \mathcal{X} \end{array} \right\},$$

or equivalently,

$$\max_{\substack{\tau, \boldsymbol{z}^0, \{\boldsymbol{z}_i^1\}_{i=1}^{n_x} \\ \boldsymbol{u}, \boldsymbol{\lambda} \geq \boldsymbol{0} \\ \{\boldsymbol{v}_k\}_{k=1}^K}} \left\{ \tau \;\middle|\; \begin{array}{ll} \tau \leq \boldsymbol{b}_y^\top \boldsymbol{z}^0 + \sum_{i \in [n_x]} \boldsymbol{b}_y^\top \boldsymbol{z}_i^1 x_i - \sum_{k=1}^K (g_k^*)^{per}(\boldsymbol{v}_k, \lambda_k) + \boldsymbol{c}^\top \boldsymbol{u} & \forall \boldsymbol{x} \in \mathcal{X} \\ \boldsymbol{A}_y^\top \boldsymbol{z}^0 + \sum_{i \in [n_x]} \boldsymbol{A}_y^\top \boldsymbol{z}_i^1 x_i - {\boldsymbol{Q}^{(0)}}^\top \boldsymbol{x} + \sum_{t=1}^T u_t {\boldsymbol{Q}^{(t)}}^\top \boldsymbol{x} \leq \sum_{k=1}^K \boldsymbol{v}_k & \forall \boldsymbol{x} \in \mathcal{X} \\ \boldsymbol{z}^0 + \sum_{i \in [n_x]} \boldsymbol{z}_i^1 x_i \geq \boldsymbol{0} & \forall \boldsymbol{x} \in \mathcal{X} \end{array} \right\}. \quad (\text{GRO}_{\text{H}}^y)$$

We now show that (GRO_{H}^y) can be interpreted as a partial RLT-AD relaxation of (GBO).

Corollary 1. *If* (GBO) *admits a Slater point and* $\mathcal{X}$ *is compact, then* (GRO_{H}^y) *can be reformulated as the following convex optimization problem:*

$$\begin{aligned} \min_{\substack{\boldsymbol{\gamma} \in \mathbb{R}_+^{n_x \times n_y} \\ \boldsymbol{x} \in \mathbb{R}_+^{n_x}, \boldsymbol{y} \in \mathbb{R}_+^{n_y}}} \quad & \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} Q_{ij}^{(0)} \gamma_{ij} & \\ \text{s.t.} \quad & \boldsymbol{A}_x \boldsymbol{x} \geq \boldsymbol{b}_x, \;\; \boldsymbol{f}(\boldsymbol{x}) \leq \boldsymbol{0}, \;\; \boldsymbol{f}^{\text{per}}(\boldsymbol{\gamma}_j, y_j) \leq \boldsymbol{0} & j \in [n_y] \\ & \boldsymbol{A}_y \boldsymbol{y} \geq \boldsymbol{b}_y, \;\; \boldsymbol{g}(\boldsymbol{y}) \leq \boldsymbol{0} & \\ & \boldsymbol{A}_x \boldsymbol{\gamma} \geq \boldsymbol{b}_x \boldsymbol{y}^\top, \;\; \boldsymbol{A}_y \boldsymbol{\gamma}^\top \geq \boldsymbol{b}_y \boldsymbol{x}^\top & \\ & \boldsymbol{A}_x \boldsymbol{\gamma} \boldsymbol{A}_y^\top \geq \boldsymbol{b}_x \boldsymbol{y}^\top \boldsymbol{A}_y^\top + \boldsymbol{A}_x \boldsymbol{x} \boldsymbol{b}_y^\top - \boldsymbol{b}_x \boldsymbol{b}_y^\top & \\ & \boldsymbol{f}^{\text{per}}\left(\left(\boldsymbol{\gamma} \boldsymbol{A}_y^\top \right)_k - (\boldsymbol{b}_y)_k \boldsymbol{x}, (\boldsymbol{A}_y \boldsymbol{y} - \boldsymbol{b}_y)_k \right) \leq \boldsymbol{0} & k \in [m_y] \\ & \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} Q_{ij}^{(t)} \gamma_{ij} \geq c_t & t \in [T]. \end{aligned} \quad (\text{GCR}_{\text{H}}^y)$$

Proof. It follows from the proof of Theorem 1 that by dualizing (GRO_{H}^y) over $(\boldsymbol{z}^0, \{\boldsymbol{z}_i^1\}_{i=1}^{n_x} \boldsymbol{u}, \boldsymbol{\lambda}, \{\boldsymbol{v}_k\}_{k=1}^K)$ and eliminating τ, we obtain (GCR_{H}^y). It follows from Beck and Ben-Tal (2009) that the optimal values of (GRO_{H}^y) and (GCR_{H}^y) coincide because $\mathcal{X}$ is compact, and (GRO_{H}^y) is convex in $(\boldsymbol{z}^0, \{\boldsymbol{z}_i^1\}_{i=1}^{n_x}, \boldsymbol{u}, \boldsymbol{\lambda}, \{\boldsymbol{v}_k\}_{k=1}^K)$. □

Note that ($\mathrm{GCR}^y_{\mathrm{H}}$) is indeed a partial RLT-AD relaxation of (GBO), because the only difference between (GCR) and ($\mathrm{GCR}^y_{\mathrm{H}}$) is that the constraints in (GCR):

$$\begin{aligned}
&\boldsymbol{g}^{\mathrm{per}}(\boldsymbol{\gamma}_i^\top, x_i) \le \mathbf{0} && i \in [n_x]\\
&\boldsymbol{g}^{\mathrm{per}}\left((\boldsymbol{A}_x\boldsymbol{\gamma})_l - (\boldsymbol{b}_x)_l\boldsymbol{y}, (\boldsymbol{A}_x\boldsymbol{x} - \boldsymbol{b}_x)_l\right) \le \mathbf{0} && l \in [m_x],
\end{aligned}$$

are not considered in ($\mathrm{GCR}^y_{\mathrm{H}}$), which implies the convex relaxation (GCR) is at least as tight as ($\mathrm{GCR}^y_{\mathrm{H}}$) (and ($\mathrm{GRO}^y_{\mathrm{H}}$)) for (GBO).

Remark 7. In Proposition 2, we dualize $\boldsymbol{y}$ in (7) to obtain the two-stage reformulation (GRO^y). One can analogously derive the corresponding (GRP^x) by dualizing $\boldsymbol{x}$ in (7). It then follows from Corollary 1 that if (GBO) admits a Slater point and $\mathcal{Y}$ is compact, where $\mathcal{Y} = \{\boldsymbol{y} \in \mathbb{R}^{n_y}_+ \mid \boldsymbol{A}_y\boldsymbol{y} \ge \boldsymbol{b}_y,\ \boldsymbol{g}(\boldsymbol{y}) \le \mathbf{0}\}$, then the corresponding ($\mathrm{GRP}^x_{\mathrm{H}}$) can be reformulated as

$$\begin{aligned}
\min_{\substack{\boldsymbol{\gamma}\in\mathbb{R}_+^{n_x\times n_y}\\ \boldsymbol{x}\in\mathbb{R}_+^{n_x},\boldsymbol{y}\in\mathbb{R}_+^{n_y}}}\ & \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} Q^{(0)}_{ij}\gamma_{ij}\\
\text{s.t.}\ & \boldsymbol{A}_x\boldsymbol{x} \ge \boldsymbol{b}_x,\ \ \boldsymbol{f}(\boldsymbol{x}) \le \mathbf{0}\\
& \boldsymbol{A}_y\boldsymbol{y} \ge \boldsymbol{b}_y,\ \ \boldsymbol{g}(\boldsymbol{y}) \le \mathbf{0},\ \ \boldsymbol{g}^{\mathrm{per}}(\boldsymbol{\gamma}_i^\top, x_i) \le \mathbf{0} && i \in [n_x]\\
& \boldsymbol{A}_x\boldsymbol{\gamma} \ge \boldsymbol{b}_x\boldsymbol{y}^\top,\ \ \boldsymbol{A}_y\boldsymbol{\gamma}^\top \ge \boldsymbol{b}_y\boldsymbol{x}^\top\\
& \boldsymbol{A}_x\boldsymbol{\gamma}\boldsymbol{A}_y^\top \ge \boldsymbol{b}_x\boldsymbol{y}^\top\boldsymbol{A}_y^\top + \boldsymbol{A}_x\boldsymbol{x}\boldsymbol{b}_y^\top - \boldsymbol{b}_x\boldsymbol{b}_y^\top\\
& \boldsymbol{g}^{\mathrm{per}}\left((\boldsymbol{A}_x\boldsymbol{\gamma})_l - (\boldsymbol{b}_x)_l\boldsymbol{y}, (\boldsymbol{A}_x\boldsymbol{x} - \boldsymbol{b}_x)_l\right) \le \mathbf{0} && l \in [m_x]\\
& \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} Q^{(t)}_{ij}\gamma_{ij} \ge c_t && t \in [T].
\end{aligned} \tag{$\mathrm{GCR}^x_{\mathrm{H}}$}$$

Similarly, the convex relaxation ($\mathrm{GCR}^x_{\mathrm{H}}$) is a partial RLT-AD relaxation of (GBO) because the constraints

$$\begin{aligned}
&\boldsymbol{f}^{\mathrm{per}}(\boldsymbol{\gamma}_j, y_j) \le \mathbf{0} && j \in [n_y]\\
&\boldsymbol{f}^{\mathrm{per}}\left(\left(\boldsymbol{\gamma}\boldsymbol{A}_y^\top\right)_k - (\boldsymbol{b}_y)_k\boldsymbol{x}, (\boldsymbol{A}_y\boldsymbol{y} - \boldsymbol{b}_y)_k\right) \le \mathbf{0} && k \in [m_y],
\end{aligned}$$

are not considered in ($\mathrm{GCR}^x_{\mathrm{H}}$), which implies the convex relaxation (GCR) is at least as tight as ($\mathrm{GCR}^x_{\mathrm{H}}$) (and ($\mathrm{GRP}^x_{\mathrm{H}}$)) for (GBO).

4. Solution methods

In §4.1, we construct a progressive approximation scheme for (BO). We then consider an existing exact reformulation for (BO) in §4.2. Finally, in §4.3, we introduce a new solution scheme based on the discussed techniques.

4.1. A progressive approximation scheme

We first show that if $\boldsymbol{\gamma}$ is a rank-one matrix, where $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$ denotes an optimal solution for (CR), then one can use $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$ to construct an optimal solution for (BO).

PROPOSITION 3. *If $\boldsymbol{\gamma}$ is a rank-one matrix, where $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$ is an optimal solution for* (CR)*, then $(\boldsymbol{\gamma}_j/y_{0j}, \boldsymbol{y}_0)$, where $j \in [n_y]$, $y_{0j} \neq 0$, constitutes an optimal solution for* (BO)*.*

Proof. Since $\boldsymbol{\gamma}$ is a rank-one matrix, there exists a vector $\boldsymbol{x}^\star$, such that $\boldsymbol{\gamma} = \boldsymbol{x}^\star \boldsymbol{y}_0^\top$. Let $\boldsymbol{x}^\star = \frac{\boldsymbol{\gamma}_j}{y_{0j}}$ for $j \in [n_y], y_{0j} \neq 0$. Note that since $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$ is feasible (and optimal) for (CR), we have $(\boldsymbol{x}^\star, \boldsymbol{y}_0)$ is feasible for (BO), and obtains the same optimal value as $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$ for (CR). □

Proposition (3) asserts that if the obtained $\boldsymbol{\gamma}$ from solving (CR) is rank-one, then $(\boldsymbol{x}_0, \boldsymbol{y}_0)$ solves (BO). However, $\boldsymbol{\gamma}$ may not be rank-one in general. We then propose to adopt the approach of Hadjiyiannis et al. (2011) to efficiently compute effective candidate solutions for (BO). In Hadjiyiannis et al. (2011), the authors propose to use binding scenarios to progressively approximate two-stage robust linear optimization problems. This method takes scenarios that are binding for the problem solved with linear decision rules, hoping that this set of binding scenarios is also binding for the optimal decision rule. We propose to construct a finite set of candidate feasible solutions for (BO) from the optimal solution $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$ of (CR):

$$\widetilde{\mathcal{X}} = \{\boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_{n_y}\}, \quad \text{where} \quad \boldsymbol{x}_j = \begin{cases} \dfrac{\boldsymbol{\gamma}_j}{y_{0j}} & \text{if} \quad y_{0j} > 0, \\ \boldsymbol{x}_0 & \text{if} \quad y_{0j} = 0, \end{cases} \quad \text{for all} \quad j \in [n_y], \tag{8}$$

and $\boldsymbol{\gamma}_j$ is the j-th column of $\boldsymbol{\gamma} \in \mathbb{R}^{n_x \times n_y}$. Note that $\widetilde{\mathcal{X}} \subseteq \mathcal{X}$ by construction, and each candidate solution $\boldsymbol{x}_j$ in $\widetilde{\mathcal{X}}$, $j \in [n_y] \cup \{0\}$, can be interpreted as a binding scenario for (RO_L).

One can observe that for any given $\boldsymbol{x} \in \mathcal{X}$, the optimal $\boldsymbol{y}$ can be obtained by solving the following linear optimization problem:

$$\min_{\boldsymbol{y} \in \mathcal{Y}} \ \boldsymbol{x}^\top \boldsymbol{Q} \boldsymbol{y}.$$

Thus, if we know the optimal $\boldsymbol{x}^\star$ for (BO), then the optimal $\boldsymbol{y}^\star$ can be readily obtained. The set $\widetilde{\mathcal{X}}$ constructed in (8) does not necessarily contain $\boldsymbol{x}^\star$. Nevertheless, any candidate solution $\boldsymbol{x}_j$ in $\widetilde{\mathcal{X}}$, $j \in [n_y] \cup \{0\}$, can be used to efficiently compute an upper bound on the optimal value of (BO) by solving the following linear optimization problem:

$$\boldsymbol{y}'_j = \arg\min_{\boldsymbol{y} \in \mathcal{Y}} \ \boldsymbol{x}_j^\top \boldsymbol{Q} \boldsymbol{y}. \tag{$\widetilde{\text{BO}}_y$}$$

Indeed, since $\boldsymbol{x}_j \in \mathcal{X}$ and $\boldsymbol{y}'_j \in \mathcal{Y}$ by construction, the optimal value $\boldsymbol{x}_j^\top \boldsymbol{Q} \boldsymbol{y}'_j$ provides an upper bound for (BO). By leveraging a mountain climbing procedure from global optimization (e.g., see,

Nahapetyan (2008)), one can solve the following linear optimization problem with $\boldsymbol{y}'_j$ obtained from $(\widetilde{\text{BO}}_y)$:

$$\boldsymbol{x}'_j = \arg\min_{\boldsymbol{x}\in\mathcal{X}} \ \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{y}'_j \tag{$\widetilde{\text{BO}}_x$}$$

to obtain a possibly tighter upper bound $(\boldsymbol{x}'_j)^\top \boldsymbol{Q}\boldsymbol{y}'_j$ for (BO). Then, solving $(\widetilde{\text{BO}}_y)$ with the fixed $\boldsymbol{x}'_j$ again gives a possibly tighter upper bound $(\boldsymbol{x}'_j)^\top \boldsymbol{Q}\boldsymbol{y}''_j$ for (BO). We perform the mountain climbing procedure for every candidate solution in $\widetilde{\mathcal{X}}$. In Algorithm 1, we summarize the described mountain climbing procedure to improve the upper bound obtained from the candidate solutions.

Algorithm 1 Mountain climbing for candidate solutions $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma}, \epsilon)$

Output: $(\boldsymbol{x}^\star, \boldsymbol{y}^\star, \text{Ub}, \text{Lb})$
procedure COMPUTING A TIGHTER UPPER BOUND
  $\widetilde{\mathcal{X}} \leftarrow$ set of candidate solutions constructed as in (8)
  $\boldsymbol{x}^\star \leftarrow \boldsymbol{x}_0$, $\boldsymbol{y}^\star \leftarrow \boldsymbol{y}_0$, $\text{Ub} \leftarrow \boldsymbol{x}_0^\top \boldsymbol{Q}\boldsymbol{y}_0$, $\text{Lb} \leftarrow \sum_{i\in[n_x]}\sum_{j\in[n_y]} Q_{ij}\gamma_{ij}$
  if $\text{Ub} - \text{Lb} > \epsilon$ **then**
    for $\boldsymbol{x} \in \widetilde{\mathcal{X}}$ **do**
      $\Delta \leftarrow 1$
      while $\Delta > 0$ **do**
        $\boldsymbol{y} \leftarrow$ solve $(\widetilde{\text{BO}}_y)$ with fixed $\boldsymbol{x}$
        $\text{Ub}_y \leftarrow \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{y}$
        $\boldsymbol{x} \leftarrow$ solve $(\widetilde{\text{BO}}_x)$ with fixed $\boldsymbol{y}$
        $\text{Ub}_x \leftarrow \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{y}$
        $\Delta \leftarrow \text{Ub}_y - \text{Ub}_x$
      endwhile
      if $\text{Ub}_x \le \text{Ub}$ **then**
        $(\boldsymbol{x}^\star, \boldsymbol{y}^\star, \text{Ub}) \leftarrow (\boldsymbol{x}, \boldsymbol{y}, \text{Ub}_x)$
      endif
    endfor
  endif

4.2. Mixed binary convex reformulation of (BO)

In this subsection, we show that disjoint bilinear optimization problem (BO) admits a mixed binary convex reformulation. By dualizing (BO) only with respect to $\boldsymbol{y}$, we obtain the following min-max bi-level problem

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{z}} \ \left\{ \boldsymbol{b}_y^\top \boldsymbol{z} \ \mid \ \boldsymbol{A}_y^\top \boldsymbol{z} \le \boldsymbol{Q}^\top \boldsymbol{x}, \ \boldsymbol{z} \in \mathbb{R}_+^{m_y} \right\}, \tag{BiO}$$

which can be reformulated as the following mixed binary convex problem

$$\min_{\substack{\boldsymbol{z},\boldsymbol{y},\boldsymbol{v}\\ \boldsymbol{x}\in\mathcal{X}}} \left\{ \boldsymbol{b}_y^\top \boldsymbol{z} \;\middle|\; \begin{array}{ll} \boldsymbol{A}_y^\top \boldsymbol{z} \le \boldsymbol{Q}^\top \boldsymbol{x}, & \boldsymbol{z}\in\mathbb{R}_+^{m_y} \\ \boldsymbol{A}_y \boldsymbol{y} \ge \boldsymbol{b}_y, & \boldsymbol{y}\in\mathbb{R}_+^{n_y} \\ \boldsymbol{A}_y \boldsymbol{y} - \boldsymbol{b}_y \le M\boldsymbol{v}, & \boldsymbol{z} \le M(\mathbf{1}-\boldsymbol{v}), \quad \boldsymbol{v}\in\{0,1\}^{m_y} \end{array} \right\}, \tag{MBC}$$

where $M \in \mathbb{R}_{++}$ is a sufficiently large number. Problem (MBC) is a mixed binary convex problem because $\mathcal{X}$ is a general convex set. A similar reformulation for the subproblem in two-stage robust linear optimization problems with a complete recourse and a polyhedral uncertainty set is proposed by Zeng and Zhao (2013), which then extended to problems without complete recourse (Ayoub and Poss 2016, Simchi-Levi et al. 2019) and nonlinear two-stage robust problems (de Ruiter et al. 2019).

4.3. Solution method in pseudocode

In the case that the minimum of (CR) is strictly smaller than the tightest upper bound, then the obtained solution $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$ may be suboptimal. We propose the following procedure to solve(BO). We first solve (CR) to obtain $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma})$. If $(\boldsymbol{x}_0, \boldsymbol{y}_0)$ is not certifiably optimal, we execute Algorithm 1 to obtain a feasible solution $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$, which may associate with a tighter upper bound $\mathrm{Ub}^\star$ for (BO). If $\mathrm{Ub}^\star$ coincides with the optimal value of (CR), then $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$ is an optimal solution of (BO). Otherwise, we propose to solve the exact mixed integer reformulation (MBC) of (BO) using $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$ as a warm-start. To restrict the feasible region of (MBC), we introduce an additional lower bound constraint

$$\boldsymbol{b}_y^\top \boldsymbol{z} \ge \sum_{i\in[n_x]} \sum_{j\in[n_y]} Q_{ij}\gamma_{ij}. \tag{9}$$

This additional lower bound constraint can be introduced without affecting the optimal solution of (MBC). The effectiveness of considering such a constraint in (MBC) is evaluated in §5.

Algorithm 2 provides the pseudocode of the proposed solution scheme. Whenever the solutions in lines 9 and 11 of Algorithm 1 are not unique, the ones obtained under the default setting of the solver are used. In what follows, we refer to our solution method by "R4B", which stands for the developed **r**obust optimization technique **f**or **b**ilinear problems.

Remark 8. Instead of solving the mixed binary convex problem (MBC), one can preprocess the corresponding (RO) by eliminating some of the wait-and-see variables via Fourier-Motzkin elimination, and then solve the corresponding (CR) to obtain additional candidate solutions. If none of the candidate solution is certifiably optimal, this procedure can be repeated until the optimal solution is obtained. As more wait-and-see variables are eliminated, the problem becomes larger, and the lower bound from (CR) becomes tighter. At the same time, by evaluating more candidate solutions, the upper bound from Algorithm 1 also becomes tighter. Finally, if all the wait-and-see variables are eliminated, one can solve the resulting static robust optimization problem (if not

Algorithm 2 R4B with a given tolerance error ϵ

Output: $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$
procedure SOLVING THE BILINEAR PROBLEM (BO)
  $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma}) \leftarrow$ solve (CR)
  $(\boldsymbol{x}^\star, \boldsymbol{y}^\star, \mathrm{Ub}, \mathrm{Lb}) \leftarrow$ execute Algorithm 1 with input $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{\gamma}, \epsilon)$
  if $\mathrm{Ub} - \mathrm{Lb} > \epsilon$ **then**
    $(\boldsymbol{x}^\star, \boldsymbol{y}^\star) \leftarrow$ solve (MBC) with initial solution $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$ and the additional constraint (9)
  endif

too large) to find the optimal solution. Therefore, Fourier-Motzkin elimination provides a finite procedure that guarantees the optimality. Despite its simplicity and appealing theoretical properties, Fourier-Motzkin elimination does not perform well numerically for the considered problems. Therefore, we relegate the detailed discussion in Appendix B.

5. Numerical experiments

In this section, we provide numerical experiments to evaluate the efficiency and effectiveness of the proposed method. In order to compare the proposed method with off-the-shelf solvers, we first consider disjoint bilinear optimization problems with polytopal feasible sets. In particular, we conduct numerical experiments on constrained bimatrix games in Section 5.1.1 and randomly generated disjoint bilinear problems in Section 5.1.2. In Section 5.2, we consider convex maximization problems, where the objective function is convex piece-wise linear and the feasible set is convex (we consider both polytopal and non-polytopal feasible sets).

5.1. Disjoint bilinear problems with polytopal feasible sets

We compare the performance of different methods on finding a Nash equilibrium of randomly generated constrainted bimatrix games and solving disjoint bilinear instances. The experiments were carried out on an Intel i5-7300U 2.60GHz Windows computer with 8GB of RAM. We used YALMIP (Löfberg (2004)) to pass the semi-definite optimization problems to MOSEK 9.2.27 (MOSEK ApS (2017)), and Linear Optimization problems to Gurobi 9.1.0. We programmed R4B in MATLAB R2020b. We compare R4B with three nonconvex quadratic optimization solvers: IBM ILOG CPLEX Optimization Studio V20.1.0 (we refer to it as CPLEX from now on), Gurobi 9.1.0, and SCIP 7.0.0 (Maher et al. (2017)). To apply CPLEX and SCIP, we use Julia 1.5.3 with JuMP 0.21.5. To have a fair comparison, we only report the running time of the solvers, and do not take into account the modeling time and the time used to pass the problems to the solvers.

In our numerical experiments, the methods are terminated if one of the following two criteria is met: the difference between the generated lower and upper bounds is less than the prescribed

tolerance error ($e = 10^{-4}$), or the running time exceeds the time limit. The best solution obtained before termination is reported.

5.1.1. Nash Equilibrium of Constrained Bimatrix Games A bimatrix game is a two-player game where each player has a finite set of pure strategies to play. The probability vectors $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and $\boldsymbol{y} \in \mathbb{R}^{n_y}$ represent the mixed strategies of Player I and II, respectively. Let $\boldsymbol{A}_x, \boldsymbol{A}_y \in \mathbb{R}^{n_x \times n_y}$ be the payoff matrices of Player I and II, respectively. Mangasarian and Stone (1964) show that a Nash equilibrium of an unconstrained bimatrix game can be obtained by solving the following disjoint bilinear problem:

$$\max_{\substack{\boldsymbol{x}\in\mathbb{R}^{n_x}_+, \boldsymbol{y}\in\mathbb{R}^{n_y}_+ \\ b_x, b_y \in \mathbb{R}}} \left\{ \boldsymbol{x}^\top(\boldsymbol{A}_x + \boldsymbol{A}_y)\boldsymbol{y} - b_x - b_y \;\middle|\; \begin{array}{l} \boldsymbol{A}_x\boldsymbol{y} \le b_x\mathbf{1},\ \mathbf{1}^\top\boldsymbol{y} = 1 \\ \boldsymbol{A}_y^\top\boldsymbol{x} \le b_y\mathbf{1},\ \mathbf{1}^\top\boldsymbol{x} = 1 \end{array} \right\}. \tag{10}$$

The optimal value of (10) is always zero and the optimal $\boldsymbol{x}$ and $\boldsymbol{y}$ is a Nash equilibrium with payoffs b_x and b_y for Player I and II, respectively.

Unconstrained bimatrix games have been studied extensively in the literature. For small-sized games, an algorithm proposed by Avis et al. (2010) (Online solver available at `http://banach.lse.ac.uk`) can find all the Nash equilibriums. Since the algorithm uses an enumeration technique to find the equilibriums, it is not applicable to large games. Probably the most well-known algorithm for unconstrained bimatrix games was proposed by Lemke and Howson (1964), which uses a pivoting algorithm to find a set of Nash equilibriums. As shown in Chen and Deng (2006), there is currently no polynomial-time algorithm to find an equilibrium of a bimatrix game in general. Therefore, there are many approximations proposed to find an equilibrium of a bimatrix game. Ahmadi and Zhang (2019) recently propose an alternative bilinear reformulation for bimatrix games, and provide semidefinite relaxations for the proposed reformulation. In Appendix C, we compare the performance of five algorithms on randomly generated unconstrained bimatrix games with the payoff matrices of size 20×20. For larger-sized games, we have the same observations. The appendix shows that both Lemke-Howson algorithm (denoted by LH) and R4B outperform SDP (Ahmadi and Zhang 2019, Algorithm 1), SCIP, Gurobi, and CPLEX for every instance.

Despite the remarkable performance of the Lemke-Howson algorithm for the moderate-sized unconstrained bimatrix games, it cannot be applied to constrained bimatrix games, where only specific types of strategies for players are acceptable. It is often more appealing to have a Nash equilibrium with mixed strategies, which can be obtained by introducing constraints, e.g., $\boldsymbol{a}_x^\top\boldsymbol{x} \le d_x$ and $\boldsymbol{a}_y^\top\boldsymbol{y} \le d_y$, to unconstrained bimatrix games (10) (Firouzbakht et al. 2016). The optimal solution of the following disjoint bilinear problem gives a Nash equilibrium with mixed strategies:

$$\max_{\substack{\boldsymbol{x}\in\mathbb{R}^{n_x}_+, \boldsymbol{y}\in\mathbb{R}^{n_y}_+ \\ b_x, b_y, \nu_x, \nu_y \in \mathbb{R}_+}} \left\{ \boldsymbol{x}^\top(\boldsymbol{A}_x + \boldsymbol{A}_y)\boldsymbol{y} - \nu_x d_x - \nu_y d_y - b_x - b_y \;\middle|\; \begin{array}{l} \boldsymbol{A}_x\boldsymbol{y} \le b_x\mathbf{1} + \boldsymbol{a}_x\nu_x,\ \mathbf{1}^\top\boldsymbol{y} = 1,\ \boldsymbol{a}_y^\top\boldsymbol{y} \le d_y \\ \boldsymbol{A}_y^\top\boldsymbol{x} \le b_y\mathbf{1} + \boldsymbol{a}_y\nu_y,\ \mathbf{1}^\top\boldsymbol{x} = 1,\ \boldsymbol{a}_x^\top\boldsymbol{x} \le d_x \end{array} \right\}. \tag{11}$$

We emphasize that the optimal value of (11) is zero if there exists an equilibrium that satisfies the linear constraints. Firouzbakht et al. (2016) show that if $d_x \geq \min_{i\in[n_x]} \boldsymbol{a}_{x_i}$ and $d_y \geq \min_{i\in[n_y]} \boldsymbol{a}_{y_i}$, then such an equilibrium exists. Let us first rewrite (11) in the form of (BO). We define

$$\bar{\boldsymbol{Q}} = -\begin{bmatrix} \boldsymbol{A}_x + \boldsymbol{A}_y & \boldsymbol{0}_{n_x} & \boldsymbol{0}_{n_x} & \boldsymbol{0}_{n_x} & \boldsymbol{0}_{n_x} \\ \boldsymbol{0}_{n_y}^\top & 0 & 0 & 0 & -1 \\ \boldsymbol{0}_{n_y}^\top & 0 & 0 & 0 & 1 \\ \boldsymbol{0}_{n_y}^\top & 0 & 0 & 0 & -d_y \\ \boldsymbol{0}_{n_y}^\top & -1 & 1 & -d_x & 0 \end{bmatrix}, \quad \bar{\boldsymbol{A}}_x = \begin{bmatrix} -\boldsymbol{a}_x^\top & 0 & 0 & 0 & 0 \\ -\boldsymbol{1}_{n_x}^\top & 0 & 0 & 0 & 0 \\ \boldsymbol{1}_{n_x}^\top & 0 & 0 & 0 & 0 \\ \boldsymbol{0}_{n_x}^\top & 0 & 0 & 0 & 1 \\ \boldsymbol{0}_{n_x}^\top & 0 & 0 & 0 & -1 \\ -\boldsymbol{A}_y^\top & \boldsymbol{1}_{n_y} & -\boldsymbol{1}_{n_y} & \boldsymbol{a}_y & \boldsymbol{0}_{n_y} \end{bmatrix},$$

$$\bar{\boldsymbol{A}}_y = \begin{bmatrix} -\boldsymbol{a}_y^\top & 0 & 0 & 0 & 0 \\ -\boldsymbol{1}_{n_y}^\top & 0 & 0 & 0 & 0 \\ \boldsymbol{1}_{n_y}^\top & 0 & 0 & 0 & 0 \\ \boldsymbol{0}_{n_y}^\top & 0 & 0 & 0 & 1 \\ \boldsymbol{0}_{n_y}^\top & 0 & 0 & 0 & -1 \\ -\boldsymbol{A}_x^\top & \boldsymbol{1}_{n_x} & -\boldsymbol{1}_{n_x} & \boldsymbol{a}_x & \boldsymbol{0}_{n_x} \end{bmatrix}, \quad \bar{\boldsymbol{d}}_x = \begin{bmatrix} -d_x \\ -1 \\ 1 \\ 1 \\ -1 \\ \boldsymbol{0}_{n_y} \end{bmatrix}, \quad \bar{\boldsymbol{d}}_y = \begin{bmatrix} -d_y \\ -1 \\ 1 \\ 1 \\ -1 \\ \boldsymbol{0}_{n_m} \end{bmatrix},$$

where $\boldsymbol{1}_n$ and $\boldsymbol{0}_n$ denote the vectors of all ones and zeros in $\mathbb{R}^n$, respectively. Then, (11) is equivalent to

$$\begin{aligned} \min_{\bar{\boldsymbol{x}},\bar{\boldsymbol{y}}} \quad & \bar{\boldsymbol{x}}^\top \bar{\boldsymbol{Q}} \bar{\boldsymbol{y}} \\ \text{s.t.} \quad & \bar{\boldsymbol{A}}_x \bar{\boldsymbol{x}} \geq \bar{\boldsymbol{d}}_x, \quad \bar{\boldsymbol{x}} \in \mathbb{R}_+^{n_x+4}, \\ & \bar{\boldsymbol{A}}_y \bar{\boldsymbol{y}} \geq \bar{\boldsymbol{d}}_y, \quad \bar{\boldsymbol{y}} \in \mathbb{R}_+^{n_y+4}, \end{aligned} \tag{12}$$

where $\bar{\boldsymbol{x}}^\top = [\boldsymbol{x}^\top,\, b_y^+,\, b_y^-,\, \nu_y\, \tau_x]^\top$ and $\bar{\boldsymbol{y}}^\top = [\boldsymbol{y}^\top,\, b_x^+,\, b_x^-,\, \nu_x\, \tau_y]^\top$, $b_x = b_x^+ - b_x^-$, $b_y = b_y^+ - b_y^-$, and τ_x and τ_y are auxiliary variables with value 1 to reformulate the linear term in the objective. We compare the performance of SCIP, CPLEX, Gurobi, R4B, and the mixed binary convex optimization approach by directly solving (MBC) using Gurobi (denoted by NE_{MBC}) on solving 50 randomly generated constrained bimatrix games with payoff matrices in $[0,1]^{30\times 30}$, vectors $\boldsymbol{a}_x, \boldsymbol{a}_y$ in $[0,1]^{30}$, $d_x \in \left[\min_{i\in[30]} \boldsymbol{a}_{x_i}, \max_{i\in[30]} \boldsymbol{a}_{x_i}\right]$, and $d_y \in \left[\min_{i\in[30]} \boldsymbol{a}_{y_i}, \max_{i\in[30]} \boldsymbol{a}_{y_i}\right]$.

Figure 2 gives the number of instances that are successfully solved by each method within 600 seconds. As one can see, all 50 instances are solved by R4B in less than 200 seconds, while none of the other methods can solve all instances within the time limit. More specifically, CPLEX and Gurobi can solve only 5 instances to optimality, and SCIP can solve only 2 instances, while NE_{MBC} solves 13 instances.

It is also important to evaluate the quality of the obtained solutions by each method. As we mentioned, R4B can solve all instances implying that the obtained solutions are optimal. CPLEX is able to find an optimal solution for each instance, though, it has difficulty in verifying the optimality of the obtained solution. For Gurobi and SCIP, optimal solutions are obtained for 18 and 2 instances, respectively.

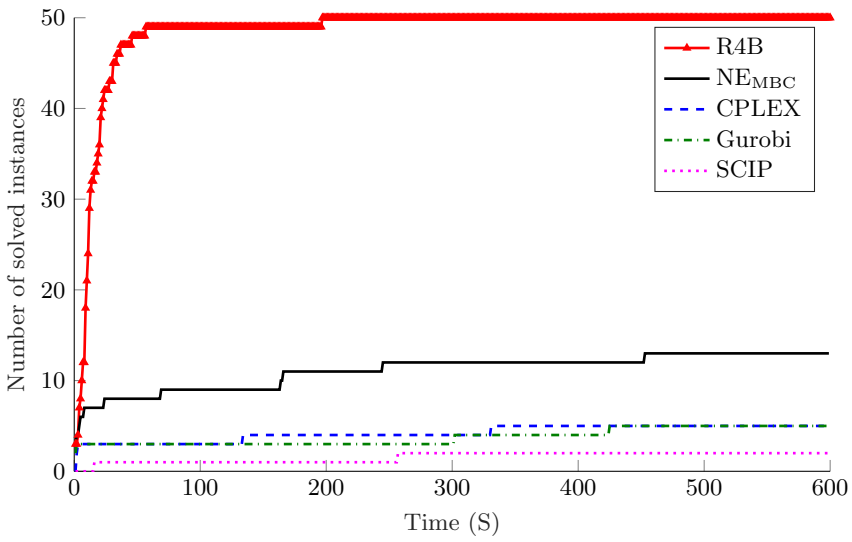

Figure 1 **Performance of different solvers on solving randomly generated constrained bimatrix games.**

5.1.2. Randomly generated bilinear instances The considered instances for (11) in §5.1 often have dense payoff matrices $\boldsymbol{A}_x$ and $\boldsymbol{A}_y$, which lead to dense disjoint bilinear problems. In this section, we consider instances with diverse structures and evaluate the performance of different solution methods.

To construct the randomly generated instances, we vary n_x and n_y in $\{50, 100\}$ to investigate the effect of dimension in the performances of the methods. Moreover, we also generate instances with different number of constraints, and consider m_x and m_y to take a value in $\{20, 300\}$. Since the density of the matrices may also affect the performance of the considered solution methods, we consider two density values, 30% and 80%, for both the coefficient matrices $\boldsymbol{A}_x$ and $\boldsymbol{A}_y$ as well as $\boldsymbol{Q}$ in the objective function. Let us denote by d_A the density of $\boldsymbol{A}_x$ and $\boldsymbol{A}_y$, and by d_Q the density of $\boldsymbol{Q}$. For given values of n_x, n_y, m_x, m_y, d_Q, and d_A, we randomly generate 5 instances (following the steps taken by Alarie et al. (2001)) as follows: we generate sparse uniformly distributed random matrix $\boldsymbol{Q} \in [-20, 20]^{n_x \times n_y}$ with density d_Q. Similarly, we generate sparse uniformly distributed random matrices $\boldsymbol{A}_x \in [1, 41]^{m_x \times n_x}$ and $\boldsymbol{A}_y \in [1, 41]^{m_y \times n_y}$ with density d_A. We also generate uniformly distributed right-hand-side vectors $\boldsymbol{b}_x \in [-10, 10]^{m_x}$ and $\boldsymbol{b}_y \in [-10, 10]^{m_y}$. We remove the rows of $\boldsymbol{A}_x$ and $\boldsymbol{A}_y$ containing only zeros and the corresponding values in $\boldsymbol{b}_x$ and $\boldsymbol{b}_y$. So, the actual density of $\boldsymbol{A}_x$ and $\boldsymbol{A}_y$ may be higher than d_A. To make sure that the feasible region is bounded, we also add the constraints $\sum_{i=1}^{n_x} x_i \leq n_x$ and $\sum_{i=1}^{n_y} y_i \leq n_y$. We only generate instances whose feasible regions are nonempty, checked by minimizing a random linear objective function over the feasible region.

We solve these 320 instances with R4B, CPLEX, Gurobi, and SCIP. We set the time limit of 1200 seconds on all methods. Figure 2 shows the number of solved instances as time progresses.

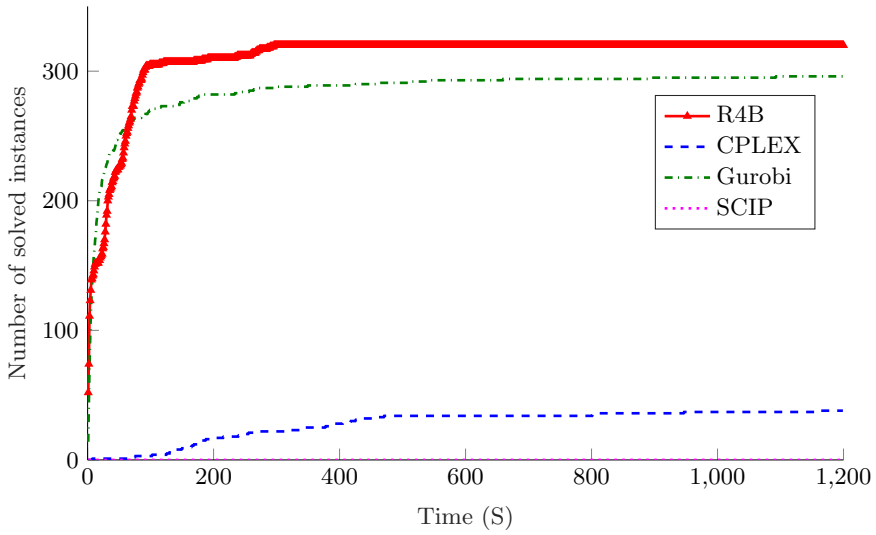

Figure 2 **Performance of different methods on solving randomly generated bilinear instances.**

Similar to the results for constrained bimatrix games, R4B can solve all 320 bilinear instances to optimality within 300 seconds. Gurobi is the second best method, as it can solve 296 instances to optimality. In comparison with R4B, Gurobi can solve around 100 instances slightly faster but has difficulty solving the large instances within the time limit. For these 24 instances, Gurobi can find an optimal solution but cannot prove the optimality within the time limit.

On the other hand, CPLEX and SCIP have difficulties in solving most of the instances. CPLEX can solve only 37 of these instances. Among the remaining 282 instances, CPLEX can find an optimal solution of 148 instances, but has difficulty proving the optimality. For 116 instances, CPLEX reports an optimality gap of less than 10% and for the other 18 instances the optimality gap can go up to 94%. We remark that for one instance, CPLEX V20.1.0 and V12.10.0 report infeasibility while CPLEX V12.9.0 is able to find an optimal solution (see *here* for more information on the bug report). SCIP seems to have difficulties even in finding good solutions for almost all instances. Except for one instance, where the optimality gap is 5%, the solutions obtained by SCIP has more than 100% optimality gap.

As we consider diverse structures in constructing instances, we also report the average performance of each method on the instances with the same structure in Appendix D. Based on these results, we see that CPLEX can deal with rather small instances where Q is sparse. Gurobi seems to have difficulties when the number of constraints is large with sparse coefficient matrix. One reason behind this computational difficulty may be Gurobi's use of RLT, which provides a looser relaxation if we have a sparser coefficient matrix. For R4B, the number of rows in $\boldsymbol{A}_y$, which corresponds to the number of binary variables in the reformulation (MBC), affects the performance

significantly. The more rows we have in $\boldsymbol{A}_y$, the more binary variables we have in (MBC), hence the longer it gets to solve the problem.

5.2. Convex maximization problem

Many convex maximization problems can be equivalently reformulated into a special case of (BO). More specifically, if $f:\mathbb{R}^{n_x}\to[-\infty,+\infty]$ is a proper closed convex function, then the following convex maximization problem

$$\max_{\boldsymbol{x}\in\mathcal{X}} \ f(\boldsymbol{x}) \tag{CM}$$

has the same optimal value as the disjoint bilinear problem

$$\min_{\substack{\boldsymbol{x}\in\mathcal{X}\\(\boldsymbol{y},\tau)\in\mathcal{Y}}} \ \tau-\boldsymbol{x}^\top\boldsymbol{y}, \tag{13}$$

where $\mathcal{Y}=\{(\boldsymbol{y},\tau)\in\mathbb{R}^{n_x+1} \mid \boldsymbol{y}\in\mathrm{dom}(f^*),\ f^*(\boldsymbol{y})\le\tau\}$ is a nonempty convex set, because $f=f^{**}$. Note that if $\mathcal{X}$ or $\mathcal{Y}$ in (13) is a polyhedral set, then (13) is an instance of (BO) with a polyhedral feasible region, and R4B can be applied.

In this section, we focus on the following generic sum-of-max-linear-terms maximization problem

$$\max_{\boldsymbol{x}\in\mathcal{X}} \ \sum_{k\in\mathcal{K}}\max_{j\in\mathcal{J}_k}\{\boldsymbol{Q}_j^\top\boldsymbol{x}+c_j\}, \tag{14}$$

or equivalently

$$\max_{\substack{\boldsymbol{x}\in\mathcal{X}\\\boldsymbol{y}\in\mathcal{Y}}} \ \boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{y}+\boldsymbol{c}^\top\boldsymbol{y}, \tag{15}$$

where $\boldsymbol{Q}\in\mathbb{R}^{n_x\times n_y}$, $\boldsymbol{c}\in\mathbb{R}^{n_y}$, $[n_y]$ is the union of the mutually disjoint sets $\mathcal{J}_k, k\in\mathcal{K}$ and $\mathcal{Y}=\left\{\boldsymbol{y}\in\mathbb{R}_+^{n_y}\mid\sum_{j\in\mathcal{J}_k}y_j=1,\ k\in\mathcal{K}\right\}$. We consider two cases of $\mathcal{X}$, those are, $\mathcal{X}=\mathcal{X}_1$ and $\mathcal{X}=\mathcal{X}_2$, where

$$\mathcal{X}_1=\{\boldsymbol{x}\in\mathbb{R}_+^{n_x}\mid\boldsymbol{A}_x^\top\boldsymbol{x}\le\boldsymbol{b}_x\}\quad\text{and}\quad\mathcal{X}_2=\left\{\boldsymbol{x}\in\mathcal{X}_1\ \middle|\ \log\left(\sum_{i\in n_x}\exp(x_i)\right)\le\Gamma\right\},$$

and $\boldsymbol{A}_x\in\mathbb{R}^{n_x\times L}$, $\boldsymbol{b}_x\in\mathbb{R}^L$, and Γ is a positive scalar.

The experiments are conducted on an Intel Core i7-8665U 1.90GHz Windows computer with 32.0GB of RAM. Computations are implemented in MATLAB (R2020a) using YALMIP (Löfberg 2004) and the solvers Gurobi 9.1.1 and MOSEK 9.2.27. All instances with $\mathcal{X}=\mathcal{X}_1$ in (15) are solved with Gurobi, while for $\mathcal{X}=\mathcal{X}_2$, we use MOSEK instead, because the considered optimization problems involve exponential cones.

For $\mathcal{X}=\mathcal{X}_1$, we generate 13 problem instances using the same numerical setting as in Selvi et al. (2020); see Appendix F for more details. We compare the performance of R4B with two alternative

approaches, those are, the recent adjustable robust approach from Selvi et al. (2020) and the exact mixed binary convex optimization approach, given by

$$\max_{\substack{\boldsymbol{z},\boldsymbol{v}\\ \boldsymbol{x}\in\mathcal{X}}} \left\{ \boldsymbol{z}^\top \mathbf{1} \;\middle|\; \begin{array}{ll} \boldsymbol{z}_k \leq \boldsymbol{Q}_j^\top \boldsymbol{x} + c_j + M(1-v_j), & j \in \mathcal{J}_k, k \in \mathcal{K} \\ \sum_{j\in\mathcal{J}_k} v_j = 1 & \\ \boldsymbol{v} \in \{0,1\}^{n_y} & \end{array} \right\}, \tag{MBCO}$$

where (MBCO) is obtained by introducing auxiliary variables for every max-term in (14), i.e., $z_k = \max_{j\in\mathcal{J}_k} \left\{\boldsymbol{Q}_j^\top \boldsymbol{x} + c_j\right\}$, and $M \in \mathbb{R}_{++}$ is a sufficiently large number. For R4B, we consider (MBCO) instead of (MBC), since it involves less optimization variables. Moreover, we compare the performance of R4B with partial R4B, that is, Algorithm 2 excluding steps 5 - 7. For $\mathcal{X} = \mathcal{X}_2$, for each instance we add a geometric constraint. We compare the performance of R4B only with partial R4B and (MBCO) in MOSEK, since the method of Selvi et al. (2020) is only able to deal with polyhedral feasible regions, and hence not for $\mathcal{X}_2$. Since MOSEK does not support warm-starts for conic problems, we do not consider warm-starts for (MBCO) in R4B. Therefore, we solve (MBCO) in R4B only with the additional upper bound constraint obtained from (CR), to restrict the feasible region. We describe the convex relaxation of (14) in more detail in Appendix E. The lower bounds (Lb), optimality gap (Gap), and the computation time (Time) of the considered approaches for $\mathcal{X} = \mathcal{X}_1$ and $\mathcal{X} = \mathcal{X}_2$ are given in Table 1 and Table 2, respectively, where we set a limit of 1000 seconds on the computation time of R4B, Gurobi, and MOSEK.

Table 1 shows that R4B yields a global optimum for instances 1, 2, 3, 7, and 8. Gurobi yields a global optimum for instances 1, 2, 3, 7, and 11. For the instances 1, 2, and 7, R4B finds the global optimum faster than Gurobi, whereas for instance 3, Gurobi has a better computation time than R4B. Opposed to R4B, Gurobi does not find a global optimum for instance 8 but does find a global optimum for instance 11. Although including an additional upper bound constraint from (CR) can help to restrict the feasible region of (MBCO), it may be surprising that by considering such a constraint, R4B can no longer solve instance 11 within 1000 seconds. We observe that if this additional upper bound constraint is ignored, R4B finds a global optimum for instance 11 in 40.35 seconds. For the remaining instances, the gaps obtained by Gurobi are larger than those obtained by R4B. Moreover, we observe that the gap of Gurobi beyond 1000 seconds does not rapidly converge to the level of the gap of R4B. For example, increasing the computation time of Gurobi from 1000 seconds to 2000 seconds for instance 12 only improves the gap by 6.04%.

Our proposed R4B improves the gap obtained from Partial R4B only for instance 3, for which R4B even yields a global optimum. The upper bounds from Partial R4B indeed coincide with the approach from Selvi et al. (2020) as shown in Theorem 1, while Algorithm 1 consistently outperforms the lower bounding scheme of the approach from Selvi et al. (2020), as can be seen in

Instance	R4B			Gurobi				
	Lb	Gap	Time	Lb	Gap	Time		
#1	23.29	0.00	0.07	23.29	0.00	0.10		
#2	233.94	0.00	0.08	233.94	0.00	0.20		
#3	1081.62	0.00	5.02	1081.62	0.00	2.76		
#4	4117.40	9.27	1000*	4117.40	9.77	1000*		
#5	23034.21	21.30	1000*	23034.21	31.03	1000*		
#6	65163.02	28.09	1000*	65126.90	44.76	1000*		
#7	113.71	0.00	0.11	113.71	0.00	0.16		
#8	3052.92	0.00	0.99	2949.84	71.90	1000*		
#9	2783.46	3.99	1000*	2292.51	275.97	1000*		
#10	3119.30	6.55	1000*	2892.73	600.43	1000*		
#11	3002.49	0.98	1000*	3002.43	0.00	36.06		
#12	3349.04	3.08	1000*	3182.70	23.54	1000*		
#13	17124.53	2.59	1000*	15694.33	470.36	1000*		
	Partial R4B			**Selvi et al. (2020)**			**Gurobi Restricted**	
	Lb	Gap	Time	Lb	Gap	Time	Lb	Gap
#1	23.29	0.00	0.07	23.29	0.00	0.16	23.29	0.00
#2	233.94	0.00	0.08	233.94	0.00	0.17	233.94	29.35
#3	1081.62	8.12	1.17	1025.68	14.01	0.40	1081.62	18.13
#4	4117.40	9.27	1.66	3923.70	14.66	0.93	4117.40	47.45
#5	23034.21	21.30	25.38	21520.68	29.83	18.89	22744.06	93.76
#6	65163.02	28.09	212.29	61866.91	34.91	134.97	65028.82	47.07
#7	113.71	0.00	0.11	108.21	5.08	0.20	113.71	209.94
#8	3052.92	0.00	0.99	2976.29	2.57	4.55	2001.14	24389.54
#9	2783.46	3.99	15.74	2418.38	19.69	154.57	2289.09	767.66
#10	3119.30	6.55	61.98	2759.03	20.46	1279.51	2766.74	917.18
#11	3002.43	0.98	1.73	2921.97	3.76	0.93	3002.43	201.34
#12	3348.99	3.08	2.71	3337.12	3.45	2.05	2825.63	1102.75
#13	17124.53	2.59	8.05	16953.85	3.62	57.25	14132.75	1462.88

Table 1 Comparison of R4B with Gurobi, partial R4B and the approach of Selvi et al. (2020) for $\mathcal{X} = \mathcal{X}_1$. Gurobi Restricted reports the obtained lower bound within the same computation time as Partial R4B. Lb denotes the obtained lower bounds, Gap denotes the optimality gaps, and Time denotes the computation time of the considered approaches. We set a time limit of 1000 seconds on the computation time for R4B and Gurobi, so if the computation time is 1000*, the global optimum cannot be found within 1000 seconds.

Table 1. Therefore, the gaps obtained from Partial R4B are at least as tight as the ones from the approach of Selvi et al. (2020). Moreover, the approach of Selvi et al. (2020) only finds a global optimum for instances 1 and 2, while Partial R4B finds a global optimum for instances 1, 2, 7, and 8. We emphasize that the lower bounds we report for the approach from Selvi et al. (2020) may differ from the lower bounds reported by Selvi et al. (2020) due to the usage of different solvers. Moreover, we observe high computation times of the approach of Selvi et al. (2020) for instances 9, 10, and 13, while if MOSEK is used instead, then the corresponding computation times are 5.55 (2660.62), 25.09 (2939.68), and 4.24 (16953.85), respectively. Given the same computation time as Partial R4B, Gurobi (Gurobi Restricted) finds the global optimum for instances 1, 2, and 7, but not for instance 8. For the remaining instances, the gaps and lower bounds obtained by Partial

Instance	R4B*			MOSEK		
	Lb	Gap	Time	Lb	Gap	Time
#1	14.58	0.00	0.22	14.58	0.00	0.14
#2	136.22	0.77	1000*	136.22	1099.67	1000*
#3	837.94	11.06	1000*	833.84	516.36	1000*
#4	3148.34	11.55	1000*	2881.00	5503.18	1000*
#5	17726.70	29.48	1000*	16377.75	1625.51	1000*
#6	50570.13	37.46	1000*	48972.70	9704.33	1000*
#7	33.73	0.00	1.70	33.73	0.00	0.48
#8	1056.83	25.56	1000*	1059.75	45151.97	1000*
#9	1548.11	56.19	1000*	1203.11	39788.34	1000*
#10	1017.22	36.52	1000*	996.39	58114.76	1000*
#11	1596.50	14.11	1000*	1610.69	2759.7	1000*
#12	1908.03	11.25	1000*	1915.56	4027.14	1000*
#13	11810.92	22.72	1000*	11794.04	41069.58	1000*
	Partial R4B			**MOSEK Restricted**		
	Lb	Gap	Time	Lb	Gap	
#1	14.58	0.00	0.22	14.58	0.00	
#2	136.22	0.77	0.78	116.35	4722.03	
#3	837.94	11.06	2.75	715.66	910.25	
#4	3148.34	11.55	10.14	2881.00	6133.39	
#5	17726.70	29.48	154.88	16377.75	1628.82	
#6	50570.13	37.46	975.34	48972.70	9704.35	
#7	33.73	7.61	1.48	33.73	0.00	
#8	1056.83	25.56	13.44	1058.42	46190.64	
#9	1548.11	56.19	40.71	1484.48	32896.46	
#10	1017.22	36.52	203.16	996.39	59081.29	
#11	1596.50	14.11	2.81	1588.59	4483.84	
#12	1908.03	11.25	5.34	1915.56	5042.82	
#13	11810.92	22.72	43.32	11794.04	4191.26	

Table 2 Comparison of R4B with Gurobi and Partial R4B for $\mathcal{X} = \mathcal{X}_2$. Gurobi Restricted reports the obtained lower bound within the same computation time as Partial R4B. Lb denotes the obtained lower bounds, Gap denotes the optimality gaps, and Time denotes the computation time of the considered approaches. We set a time limit of 1000 seconds on the computation time for R4B and Gurobi, so if the computation time is 1000*, the global optimum cannot be found within 1000 seconds.

R4B are better than the ones found by Gurobi Restricted. Hence, within the same computation time, Partial R4B performs better than Gurobi.

In Table 2, we consider $\mathcal{X} = \mathcal{X}_2$ and choose the value of Γ such that it restricts the feasible region of $\mathcal{X}_1$. Table 2 shows that both R4B and MOSEK find the global optimum for instances 1 and 7. Partial R4B and MOSEK under the same computational effort as Partial R4B (MOSEK Restricted) only find the global optimum for instance 1. The gap found by R4B within 1000 seconds is much smaller than the gap found by MOSEK for unsolved instances. By including the upper bound constraints in (MBCO) in R4B, we observe that the gap is reduced enormously compared to MOSEK. Moreover, for instances 3, 4, 5, 6, 8, 9, 10, 11, 12, and 13, the lower bounds found by R4B are better than those found by MOSEK. R4B improves the gap of Partial R4B for instance 7, while for the remaining instances, the gaps are exactly the same with a much longer computation

time. Except for instance 1 and 7, the gaps obtained by Partial R4B are much lower than those found by MOSEK within the same amount of time. Moreover, except for instances 1, 7, and 12, the lower bounds obtained by Partial R4B are better than the ones obtained by MOSEK under the same computational effort.

6. Conclusion and future research

In this paper, we investigate disjoint bilinear optimization problems from a two-stage robust optimization perspective. A new way of deriving a reformulation-linearization technique for disjoint bilinear optimization problems is introduced. We further propose a new algorithm to solve disjoint bilinear optimization problems, which we call R4B. We compare the numerical performance of R4B with existing solvers and popular methods, and observe that R4B outperforms the considered solvers and existing approaches for almost all instances.

One immediate future research direction would be to derive new reformulation-linearization techniques by considering other decision rules, e.g., quadratic decision rules, for disjoint bilinear optimization problems with a (non-convex) quadratic feasible set.

Acknowledgment

The authors would like to thank Erick Delage, Daniel Kuhn, and Napat Rujeerapaiboon for their valuable suggestions on an earlier version of this paper.

Appendices

A. Bilinear problems with quadratic feasible sets

Consider (BO) with feasible sets $\mathcal{X} = \left\{\boldsymbol{x} \in \mathbb{R}^{n_x} \mid \boldsymbol{x}^\top \boldsymbol{A}_x \boldsymbol{x} + \boldsymbol{b}_x^\top \boldsymbol{x} \leq c_x\right\}$ and $\mathcal{Y} = \left\{\boldsymbol{y} \in \mathbb{R}^{n_y} \mid \boldsymbol{y}^\top \boldsymbol{A}_y \boldsymbol{y} \leq c_y\right\}$, where $\boldsymbol{A}_x \in \mathbb{S}^{n_x \times n_x}$, $\boldsymbol{A}_y \in \mathbb{S}_{++}^{n_y \times n_y}$, $\boldsymbol{b}_x \in \mathbb{R}^{n_x}$, $c_x \in \mathbb{R}$ and $c_y \in \mathbb{R}_+$. We refer to (BO) of this form as (BQP).

Assumption 1. *The matrices $\boldsymbol{A}_x$ and $-\boldsymbol{Q}\boldsymbol{A}_y^{-1}\boldsymbol{Q}^\top$ are simultaneously diagonalizable, i.e., there exists an orthonormal matrix $\boldsymbol{S}$ such that:*

$$\boldsymbol{S}^\top \boldsymbol{A}_x \boldsymbol{S} = \operatorname{diag}(\alpha_1, ..., \alpha_{n_x}) \quad \text{and} \quad -c_y \boldsymbol{S}^\top \boldsymbol{Q}\boldsymbol{A}_y^{-1}\boldsymbol{Q}^\top \boldsymbol{S} = \operatorname{diag}(\delta_1, ..., \delta_{n_x}), \tag{16}$$

and there does not exist an $i \in [n_x]$ such that $\delta_i \leq 0$ and $\alpha_i \leq 0$ with at least one strict inequality.

For instance, if $\boldsymbol{A}$ and/or $\boldsymbol{B}$ are positive semidefinite, $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{S}^{n \times n}$, then the two matrices are simultaneously diagonalizable. We refer to Golub and van Loan (1989) for the existing methods for simultaneously diagonalizing two matrices.

The following proposition states that (BQP) can be written as a conic quadratic program.

PROPOSITION 4. *If Assumption 1 holds, then the solution* $(\boldsymbol{x}^\star, \boldsymbol{y}^\star)$*, where* $\boldsymbol{x}^\star \in \mathcal{X}$ *and* $\boldsymbol{y}^\star \in \mathcal{Y}$*, is optimal for* (BQP) *if and only if there exists a* $(\boldsymbol{v}^\star, \boldsymbol{u}^\star)$ *that is optimal for the following conic quadratic optimization problem:*

$$\begin{aligned} \min_{\boldsymbol{v},\boldsymbol{u}} \quad & \sum_{i\in[n_x]} \delta_i u_i \\ \text{s.t.} \quad & \sum_{i\in[n_x]} (\alpha_i u_i + \beta_i v_i) \le c_x \\ & v_i^2 - u_i \le 0 \qquad\qquad i \in [n_x], \end{aligned} \tag{17}$$

where α_i*,* δ_i*,* $i \in [n_x]$*, are as given in* (16)*,* $\boldsymbol{x}^\star = \boldsymbol{S}\boldsymbol{v}^\star$*, and* $\boldsymbol{y}^\star = \dfrac{-\boldsymbol{A}_y^{-1}\boldsymbol{Q}^\top \boldsymbol{S}\boldsymbol{v}^\star \sqrt{c_y}}{\sqrt{\boldsymbol{\delta}^\top \boldsymbol{u}^\star}}$*.*

Proof. Let us consider (BQP):

$$\min_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}\in\mathcal{Y}} \ \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{y}.$$

By optimizing over $\boldsymbol{y}$, the problem is equivalent to:

$$\min_{\boldsymbol{x}\in\mathcal{X}} \ -\sqrt{c_y \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{A}_y^{-1}\boldsymbol{Q}^\top \boldsymbol{x}}. \tag{18}$$

It can be easily verified that $\boldsymbol{x}^\star$ is optimal for (18) if and only if $\boldsymbol{x}^\star$ is optimal for:

$$\min_{\boldsymbol{x}\in\mathcal{X}} \ -c_y \boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{A}_y^{-1}\boldsymbol{Q}^\top \boldsymbol{x}. \tag{19}$$

Since the matrices $\boldsymbol{A}_x$ and $-c_y \boldsymbol{Q}\boldsymbol{A}_y^{-1}\boldsymbol{Q}^\top$ are simultaneously diagonalizable, applying Ben-Tal and den Hertog (2014, Theorem 3), Problem (19) can be equivalently reformulated as (17) by substituting $\boldsymbol{x} = \boldsymbol{S}\boldsymbol{v}$ and $\boldsymbol{\beta} = \boldsymbol{S}^\top \boldsymbol{b}_x$ in (19), where $\boldsymbol{S}$ satisfies (16), which exists by assumption, and then replacing v_i^2 with its epigraph variable u_i for each $i \in [n_x]$. □

We would like to remark that a similar result can be derived even if $\mathcal{X}$ is an intersection of a unit ball and a (nonconvex) quadratic constraint (Ben-Tal and den Hertog 2014), i.e., $\mathcal{X} = \left\{\boldsymbol{x} \in \mathbb{R}^{n_x} \mid \boldsymbol{x}^\top \boldsymbol{x} \le 1, \ \boldsymbol{x}^\top \boldsymbol{A}_x \boldsymbol{x} + \boldsymbol{b}_x^\top \boldsymbol{x} \le c_x\right\}$. Furthermore, one can extend the result of Proposition 4 to (BQP) with $\boldsymbol{A}_y \in \mathbb{S}^{n_y \times n_y}$, i.e., $\mathcal{Y}$ might be non-convex. A necessary and sufficient condition for simultaneously diagonalizability of two matrices is given in Jiang and Li (2016).

B. Preprocessing via Fourier-Motzkin Elimination (Zhen et al. 2017)

One can use Fourier-Motzkin elimination as a preprocessing procedure to eliminate the wait-and-see variables in two-stage robust optimization problems, and establish guaranteed optimality in finite number of steps. In this way, problems with few wait-and-see variables can be solved to optimality. For problems with many wait-and-see variables, one could eliminate a subset of the wait-and-see variables and then apply linear decision rules to approximate the resulting problems.

Algorithm 3 Fourier-Motzkin elimination for two-stage robust linear problems.

1: For some $l \in [m_y]$, rewrite each constraint in ($\mathrm{RP}_{\setminus\emptyset}$) in the form: there exists $\boldsymbol{z} \in \mathcal{R}_+^{n_x,m_y}$,

$$A_{il} z_l(\boldsymbol{x}) \le \boldsymbol{B}_i^\top \boldsymbol{x} + \tau b_i - \sum_{j \in [m_y] \setminus \{l\}} A_{ij} z_j(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{X} \quad \forall i \in [n_y + 1];$$

if $A_{il} \neq 0$, divide both sides by A_{il}. We obtain an equivalent representation of the feasible region of ($\mathrm{RP}_{\setminus\emptyset}$) involving the following constraints: there exists $\boldsymbol{z} \in \mathcal{R}_+^{n_x,m_y}$,

$$z_l(\boldsymbol{x}) \le \boldsymbol{G}_i^\top \boldsymbol{x} + \tau g_i + \boldsymbol{H}_i^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \in \mathcal{X} \quad \text{if } A_{il} > 0, \tag{20}$$

$$\boldsymbol{G}_j^\top \boldsymbol{x} + \tau g_j + \boldsymbol{H}_j^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \le z_l(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \in \mathcal{X} \quad \text{if } A_{jl} < 0, \tag{21}$$

$$0 \le \boldsymbol{G}_k^\top \boldsymbol{x} + \tau g_k + \boldsymbol{H}_k^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \in \mathcal{X} \quad \text{if } A_{kl} = 0. \tag{22}$$

Here, each $\boldsymbol{H}_i, \boldsymbol{H}_j, \boldsymbol{H}_k$ is a vector in $\mathbb{R}^{m_y - 1}$, each g_i, g_j, g_k is a scalar, and each $\boldsymbol{G}_i, \boldsymbol{G}_j, \boldsymbol{G}_k$ is a vector in $\mathbb{R}^{n_x}$.

2: After the wait-and-see variable z_l is eliminated, the feasible set of ($\mathrm{RP}_{\setminus\emptyset}$) becomes: there exists $\boldsymbol{z} \in \mathcal{R}_+^{n_x,m_y}$,

$$\boldsymbol{G}_j^\top \boldsymbol{x} + \tau g_j + \boldsymbol{H}_j^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \le \boldsymbol{G}_i^\top \boldsymbol{x} + \tau g_i + \boldsymbol{H}_i^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{X} \quad \text{if } A_{jl} < 0 \text{ and } A_{il} > 0, \tag{23}$$

$$0 \le \boldsymbol{G}_k^\top \boldsymbol{x} + \tau g_k + \boldsymbol{H}_k^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{X} \quad \text{if } A_{kl} = 0. \tag{24}$$

For simplicity, we consider the following equivalent formulation of (RO) throughout the rest of this section:

$$\max_{\tau} \ \left\{ \tau \ \middle| \ \exists \boldsymbol{z} \in \mathcal{R}_+^{n_x,m_y} : \boldsymbol{A}\boldsymbol{z}(\boldsymbol{x}) \le \boldsymbol{B}^\top \boldsymbol{x} + \tau \boldsymbol{b} \ \ \forall \boldsymbol{x} \in \mathcal{X} \ \right\}, \tag{$\mathrm{RP}_{\setminus\emptyset}$}$$

where $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_y^\top \\ -\boldsymbol{b}_y^\top \end{bmatrix}$, $\boldsymbol{B} = \begin{bmatrix} \boldsymbol{Q}^\top \\ \boldsymbol{0}^\top \end{bmatrix}$ and $\boldsymbol{b} = -\boldsymbol{e}_{n_y+1}$. We use Algorithm 3 to eliminate z_l, $l \in [m_y]$, in ($\mathrm{RP}_{\setminus\emptyset}$). After eliminating the wait-and-see variable z_l, let us denote the resulting reformulation of ($\mathrm{RP}_{\setminus\emptyset}$) as:

$$\max_{\tau} \ \left\{ \tau \ \middle| \ \exists \boldsymbol{z}_{\setminus\{l\}} \in \mathcal{R}_+^{n_x,m_y-1} : \boldsymbol{H}^\top \boldsymbol{z}_{\setminus\{l\}}(\boldsymbol{x}) \le \boldsymbol{G}^\top \boldsymbol{x} + \tau \boldsymbol{g} \ \ \forall \boldsymbol{x} \in \mathcal{X} \ \right\}, \tag{$\mathrm{RP}_{\setminus\{l\}}$}$$

where $\boldsymbol{H} \in \mathbb{R}^{(m_y-1)\times m}, \boldsymbol{G} \in \mathbb{R}^{n_x \times m}$, and $\boldsymbol{g} \in \mathbb{R}^m$ are the resulting coefficients of $\boldsymbol{z}_{\setminus\{l\}}$, $\boldsymbol{x}$ and τ, respectively. Zhen et al. (2017, Theorem 2) shows that the more wait-and-see variables are eliminated via Fourier-Motzkin elimination, the better the approximation from decision rules becomes; if all the wait-and-see variables are eliminated, the resulting equivalent reformulation is a static robust optimization problem. However, it is well-known that Fourier-Motzkin elimination may produce many redundant constraints. In order to keep the resulting formulation at its minimal size, after eliminating a wait-and-see variable via Fourier-Motzkin elimination, we execute an LO-based

procedure to detect and remove the redundant constraints. We test whether the j-th inequality, $j \in [m]$, in $(\mathrm{RP}_{\setminus\{l\}})$ is implied by the rest via the following problem:

$$\begin{aligned} \max_{\substack{\tau,\boldsymbol{x}\in\mathcal{X}\\ \boldsymbol{z}\geq\boldsymbol{0}}} \quad & \boldsymbol{H}_j^\top \boldsymbol{z} - \boldsymbol{G}_j^\top \boldsymbol{x} - \tau g_j \\ \text{s.t.} \quad & \boldsymbol{H}_i^\top \boldsymbol{z} \leq \boldsymbol{G}_i^\top \boldsymbol{x} + \tau g_i \qquad i \in [m] \setminus \{j\}, \end{aligned}$$

where $\boldsymbol{H}_i$ and $\boldsymbol{G}_i$ the i-th column of $\boldsymbol{H}$ and $\boldsymbol{G}$, and g_i is the i-th element of the vector $\boldsymbol{g}$, $i \in [m]$. Then, the j-th inequality is redundant if the optimal value is less than or equal to 0. By successively applying this LO-based procedure for each untested inequality against the remaining, one could efficiently remove the redundant constraints in $(\mathrm{RP}_{\setminus\{l\}})$. We refer to Zhen et al. (2017) for more details.

Remark 9. Algorithm 3 often leads to a quadratic increase in the number of inequalities in $(\mathrm{RP}_{\setminus\{l\}})$ with respect to $(\mathrm{RP}_{\setminus\emptyset})$. Dualizing $(\mathrm{RP}_{\setminus\{l\}})$ with respect to $\boldsymbol{z}$ would result in a disjoint bilinear problem with a higher dimensional (dual) $\widehat{\boldsymbol{y}}$ than the $\boldsymbol{y}$ in the original (BO). Therefore, Algorithm 3 can also be interpreted as a systematic lifting procedure for the decision variable $\boldsymbol{y}$ in the original (BO). Furthermore, if all $\boldsymbol{z}$ in $(\mathrm{RP}_{\setminus\emptyset})$ are eliminated, the resulting $(\mathrm{RP}_{\setminus[m_y]})$ is simply a static robust optimization problem possibly with many constraints, which can be reformulated into a convex optimization problem using robust optimization techniques.

C. Numerical experiment on unconstrained bimatrix games

In this appendix, we provide a numerical comparison of six methods in finding a Nash equilibrium of unconstrained bimatrix games: LH (Lemke and Howson 1964), R4B, SDP (Ahmadi and Zhang 2019, Algorithm 1), SCIP, CPLEX, and Gurobi. This comparison has been made with applying the methods to solve 50 randomly generated bimatrix games with payoff matrices in $[0,1]^{20\times 20}$.

To compare the quality of the obtained solutions, we report $\epsilon = \frac{|fval|}{|fval+10^{-4}|}$ as the optimality gap, where $fval$ is the objective value of the obtained solution. Since we know the optimal value of (10) is zero, ϵ indicates how far the solutions are from optimality. Figure 3 provides us with the performance of the methods with respect to the termination time and the optimality gap.

As one can see in Figure 3, LH and R4B obtained an optimal solution with optimality guarantee within less than 0.5 seconds for all 50 randomly generated instances. Among the other four methods, SDP can obtain optimal solutions for only four instances with optimality guarantees, while the other three methods have difficulty in either finding an optimal solution or guaranteeing the optimality of the found solution for all instances. CPLEX obtains optimal solutions for 39 instances without optimality gaurantee while Gurobi obtains optimal solutions for only 9 instances. SCIP fails to even obtain a near optimal solution.

To have a better understanding of Figure 3, we report the summary statistics on the termination time and the optimality gap in Tables 3 and 4, respectively. Considering both quality (the optimality

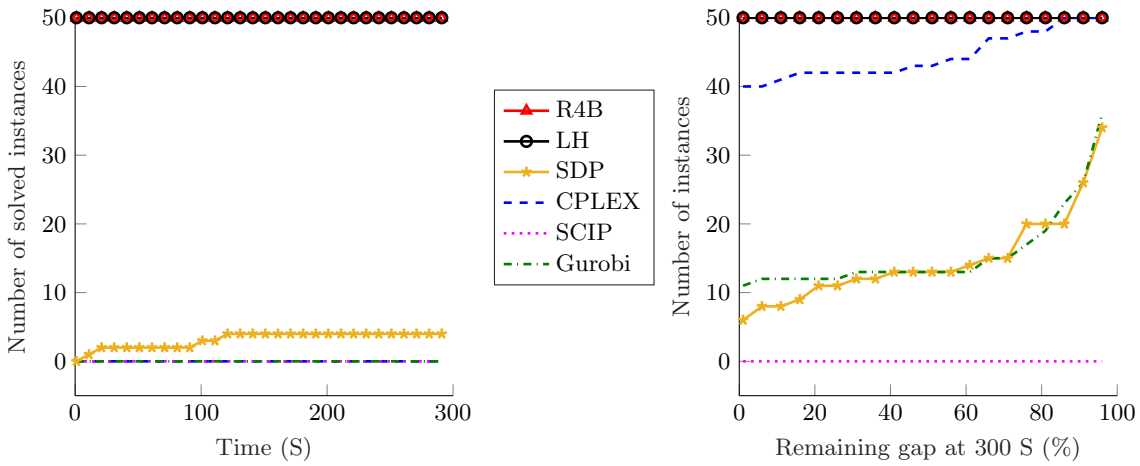

Figure 3 **Comparison of LH, R4B, SDP, CPLEX, SCIP, and Gurobi with solving 50 randomly generated bimatrix games. The comparison is done in two different ways: the left figure plots the termination time for the methods with the time limit of 300 Seconds. The right figure plots the optimality gap of the solutions found by the methods.**

gap) and speed (time of computation), LH and R4B outperform the other four methods in every instance. Codenotti et al. (2008) show that the average computation time of LH grows exponentially as the size of the games increases. Note that LH is only applicable to unconstrained bimatrix games.

Table 3 **Summary statistics of termination time (in seconds) for 50 randomly generated bimatrix games.**

	mean	StDev	max	min
LH	0.00	0.00	0.00	0.00
R4B	0.23	0.08	0.40	0.04
SDP	280.85	67.06	300.00	9.09
CPLEX	300.00	0.00	300.00	300.00
SCIP	300.00	0.00	300.00	300.00
Gurobi	300.00	0.00	300.00	300.00

Table 4 **Summary statistics of optimality gap (ϵ in %) for 50 randomly generated bimatrix games. We use best sol. and optimal sol. to denote the number of the instances that the corresponding solver/solution method finds the lowest upper bound and solution with optimality guarantee, respectively.**

	mean	StDev	max	min	best sol.	optimal sol.
LH	0.00	0.00	0.00	0.00	50	50
R4B	0.00	0.00	0.00	0.00	50	50
SDP	69.31	37.60	99.22	0.00	4	4
CPLEX	10.95	24.81	85.59	0.00	39	0
SCIP	99.95	0.01	99.96	99.91	0	0
Gurobi	67.40	40.03	99.28	0.00	9	0

D. Results on randomly generated bilinear problems

n_x	n_y	m_x	m_y	d_Q	d_A	R4B		Gurobi		CPLEX		SCIP	
						mean	st. dev.	mean	st. dev.	mean	st. dev.	mean	st. dev.
50	50	20	20	0.3	0.3	0.65	0.09	0.97	0.24	139.82	16.09	1200.00	0.00
					0.8	0.62	0.10	0.76	0.15	143.14	54.93	1200.00	0.00
				0.8	0.3	0.60	0.13	3.37	1.34	1200.00	0.00	1200.00	0.00
					0.8	0.50	0.05	1.62	0.65	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	37.29	4.55	2.30	0.56	398.12	89.69	1200.00	0.00
					0.8	25.50	6.53	1.33	0.27	188.22	114.41	1200.00	0.00
				0.8	0.3	32.53	5.10	20.31	7.97	1200.00	0.00	1200.00	0.00
					0.8	27.67	2.27	4.76	2.68	1200.00	0.00	1200.00	0.00
		300	20	0.3	0.3	3.33	0.55	3.53	1.40	352.64	65.74	1200.00	0.00
					0.8	3.07	2.11	1.77	0.63	213.35	35.84	1200.00	0.00
				0.8	0.3	1.76	0.87	15.35	5.06	1200.00	0.00	1200.00	0.00
					0.8	1.92	2.02	4.02	1.06	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	29.86	5.13	958.26	482.93	1060.48	178.89	1200.00	0.00
					0.8	34.20	2.14	11.67	9.56	632.57	367.41	1200.00	0.00
				0.8	0.3	27.32	2.99	1068.16	294.80	1200.00	0.00	1200.00	0.00
					0.8	30.66	1.36	44.28	35.01	1200.00	0.00	1200.00	0.00
	100	20	20	0.3	0.3	0.92	0.04	2.41	0.93	1200.00	0.00	1200.00	0.00
					0.8	1.05	0.09	1.40	0.40	1200.00	0.00	1200.00	0.00
				0.8	0.3	1.00	0.09	9.53	7.57	1200.00	0.00	1200.00	0.00
					0.8	0.95	0.02	3.26	1.62	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	54.38	13.54	14.55	6.50	1200.00	0.00	1200.00	0.00
					0.8	58.14	8.18	4.28	1.91	1200.00	0.00	1200.00	0.00
				0.8	0.3	65.39	7.24	37.56	20.40	1200.00	0.00	1200.00	0.00
					0.8	58.75	2.09	9.40	7.24	1200.00	0.00	1200.00	0.00
		300	20	0.3	0.3	4.89	0.61	12.41	10.00	1200.00	0.00	1200.00	0.00
					0.8	3.87	3.17	7.96	6.37	1200.00	0.00	1200.00	0.00
				0.8	0.3	5.20	3.39	120.73	80.87	1200.00	0.00	1200.00	0.00
					0.8	10.95	9.34	9.26	3.59	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	81.70	6.65	651.84	319.20	1200.00	0.00	1200.00	0.00
					0.8	74.71	6.57	19.62	7.05	1200.00	0.00	1200.00	0.00
				0.8	0.3	77.85	5.30	1200.00	0.00	1200.00	0.00	1200.00	0.00
					0.8	81.40	14.67	46.58	34.30	1200.00	0.00	1200.00	0.00
	50	20	20	0.3	0.3	0.98	0.16	2.52	1.22	1200.00	0.00	1200.00	0.00
					0.8	1.01	0.15	1.11	0.25	1200.00	0.00	1200.00	0.00
				0.8	0.3	1.12	0.28	5.62	1.67	1200.00	0.00	1200.00	0.00
					0.8	0.93	0.02	2.73	0.89	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	30.30	6.88	8.60	2.07	1200.00	0.00	1200.00	0.00
					0.8	40.61	13.48	4.87	1.79	1200.00	0.00	1200.00	0.00
				0.8	0.3	31.18	3.57	67.65	20.12	1200.00	0.00	1200.00	0.00
					0.8	39.01	14.21	13.37	10.47	1200.00	0.00	1200.00	0.00
		300	20	0.3	0.3	5.28	3.16	4.03	1.20	1200.00	0.00	1200.00	0.00
					0.8	2.08	0.32	3.40	0.61	1200.00	0.00	1200.00	0.00
				0.8	0.3	3.39	0.36	80.63	66.07	1200.00	0.00	1200.00	0.00
					0.8	4.09	4.46	7.82	3.86	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	61.54	4.19	531.15	612.69	1200.00	0.00	1200.00	0.00
					0.8	54.12	12.90	14.50	6.04	1200.00	0.00	1200.00	0.00
				0.8	0.3	57.78	2.53	1012.75	418.71	1200.00	0.00	1200.00	0.00
					0.8	48.11	16.96	29.04	9.68	1200.00	0.00	1200.00	0.00
			20	0.3	0.3	2.49	0.35	4.46	1.48	1200.00	0.00	1200.00	0.00
					0.8	2.99	0.19	2.25	0.24	1200.00	0.00	1200.00	0.00
				0.8	0.3	2.60	0.25	17.69	9.53	1200.00	0.00	1200.00	0.00
					0.8	2.45	0.26	9.82	3.66	1200.00	0.00	1200.00	0.00
				0.3	0.3	66.28	12.64	43.64	18.03	1200.00	0.00	1200.00	0.00
					0.8	73.71	16.30	9.44	1.07	1200.00	0.00	1200.00	0.00

			300	0.8	0.3	68.35	6.61	176.52	90.70	1200.00	0.00	1200.00	0.00
					0.8	79.83	7.77	37.83	19.77	1200.00	0.00	1200.00	0.00
	100		20	0.3	0.3	14.59	6.31	30.70	21.54	1200.00	0.00	1200.00	0.00
					0.8	13.36	11.99	8.50	0.92	1200.00	0.00	1200.00	0.00
				0.8	0.3	10.60	4.52	174.94	99.08	1200.00	0.00	1200.00	0.00
		300			0.8	7.84	7.42	16.49	4.72	1200.00	0.00	1200.00	0.00
			300	0.3	0.3	274.57	14.87	393.38	460.25	1200.00	0.00	1200.00	0.00
					0.8	158.73	80.45	23.87	13.78	1200.00	0.00	1200.00	0.00
				0.8	0.3	218.26	37.59	1139.07	136.25	1200.00	0.00	1200.00	0.00
					0.8	119.47	85.61	68.45	39.24	961.54	533.21	1200.00	0.00

E. Convex relaxation of problem (14)

By Theorem 1 we obtain the following convex relaxation of problem (14) with $\mathcal{X} = \mathcal{X}_2$

$$\max_{\boldsymbol{y}\geq 0,\boldsymbol{\gamma},\boldsymbol{x}} \quad \sum_{i\in[n_x]}\sum_{j\in[n_y]} Q_{ij}\gamma_{ij} + \boldsymbol{c}^\top \boldsymbol{y} \tag{25a}$$

$$\text{s.t.} \quad \log\left(\sum_{i\in[n_x]} \exp(x_i)\right) \leq \Gamma \tag{25b}$$

$$\sum_{j\in\mathcal{J}_k} y_j - 1 = 0 \qquad k\in[K] \tag{25c}$$

$$\boldsymbol{A}_x^\top \boldsymbol{\gamma}_j - y_j \boldsymbol{b}_x \leq \boldsymbol{0} \qquad j\in[n_y] \tag{25d}$$

$$y_j \log\left(\sum_{i\in[n_x]} \exp\left(\frac{\gamma_{ij}}{y_j}\right)\right) \leq \Gamma \qquad j\in[n_y] \tag{25e}$$

$$\boldsymbol{\gamma} \geq 0 \tag{25f}$$

$$\boldsymbol{x} - \sum_{j\in\mathcal{J}_k} \boldsymbol{\gamma}_j = \boldsymbol{0} \qquad k\in[K]. \tag{25g}$$

Observe that by applying Theorem 1 we also obtain the following constraints

$$\boldsymbol{A}_x^\top \boldsymbol{x} - \boldsymbol{b}_x \leq \boldsymbol{0},$$

$$\boldsymbol{x} \geq \boldsymbol{0},$$

$$\left(1 - \sum_{j\in\mathcal{J}_k} y_j\right)\left(\boldsymbol{A}_x^\top \left(\frac{\boldsymbol{x} - \sum_{j\in\mathcal{J}_k}\boldsymbol{\gamma}_j}{1 - \sum_{j\in\mathcal{J}_k} y_j}\right) - \boldsymbol{b}_x\right) \leq \boldsymbol{0},$$

$$\left(1 - \sum_{j\in\mathcal{J}_k} y_j\right) \log\left(\sum_{i\in[n_x]} \exp\left(\frac{\boldsymbol{x} - \sum_{j\in\mathcal{J}_k}\boldsymbol{\gamma}_j}{1 - \sum_{j\in\mathcal{J}_k} y_j}\right) - \Gamma\right) \leq 0.$$

It can be easily checked that the first two constraints are redundant, as are the last two constraints by Zhen et al. (2020a, Theorem 1).

In the same way, for $\mathcal{X}_1$ we obtain the convex relaxation that consists of (25a), (25c), (25d), (25f) and (25g).

	Q_{ij}	c_j	$(A_x)_{ij}$	b_i	n_x	$\lvert\mathcal{J}_k\rvert$	$\lvert\mathcal{K}\rvert$	Γ	M
#1	$\sim[-5,5]$		-	-	5	5	1	3	100
#2	$\sim[-5,5]$		-	-	5	5	10	3	1000
#3	$\sim[-5,5]$		-	-	20	10	10	11	1000
#4	$\sim[-5,5]$		-	-	30	20	20	16	10000
#5	$\sim[-5,5]$		-	-	100	40	30	50	10000
#6	$\sim[-4,4]$		-	-	200	50	50	100	100000
#7	$\sim[-5,5]$	$\sim[-10,10]$	$\sim[0,1]$	$\sim[5,15]$	10	5	2	3	1000
#8	$\sim[-5,5]$	$\sim[-10,10]$	$\sim[0,1]$	$\sim[5,15]$	10	50	50	3	10000
#9	$\sim[-5,5]$	$\sim[-10,10]$	$\sim[0,1]$	$\sim[5,15]$	30	50	50	4	10000
#10	$\sim[-5,5]$	$\sim[-10,10]$	$\sim[0,1]$	$\sim[5,15]$	50	60	60	4	10000
#11	$\sim[-5,10]$	$\sim[-10,10]$	$(A_x)'_{ij}$	b'_i	20	10	10	5	10000
#12	$\sim[-5,10]$	$\sim[-10,10]$	$(A_x)^*_{ij}$	b^*_i	20	50	10	5	10000
#13	$\sim[-5,10]$	$\sim[-10,10]$	$(A_x)'_{ij}$	b'_i	20	100	50	6	100000

Table 6 **Values for the parameters and coefficients of each distinct convex maximization problem considered.**

F. Data for convex maximization problem of Section 5.2

The instances are generated using the same numerical setting as in Selvi et al. (2020). In each instance, every max-term is generated with the same number of elements, i.e., $|\mathcal{J}_k| = |\mathcal{J}_{k'}|$ for all $k, k' \in \mathcal{K}$. For instance, when $\mathcal{X} = \mathcal{X}_2$ in (14), the instances 1 to 6 are defined by:

$$\begin{aligned} \max_{\boldsymbol{x}\geq\boldsymbol{0}} \quad & \sum_{k\in\mathcal{K}} \max_{j\in\mathcal{J}_k} \{\boldsymbol{Q}_j^\top \boldsymbol{x} + c_j\} \\ \text{s.t.} \quad & \frac{i}{n_x} \leq x_i \leq \frac{n_x}{i} \qquad\qquad i \in [n_x] \\ & \log\left(\sum_{i\in[n_x]} \exp(x_i)\right) \leq \Gamma, \end{aligned}$$

and problems 7 up to 13 are defined by:

$$\begin{aligned} \max_{\boldsymbol{x}\geq\boldsymbol{0}} \quad & \sum_{k\in\mathcal{K}} \max_{j\in\mathcal{J}_k} \{\boldsymbol{Q}_j^\top \boldsymbol{x} + c_j\} \\ \text{s.t.} \quad & \boldsymbol{A}_x^\top \boldsymbol{x} \leq \boldsymbol{b}_x \\ & \log\left(\sum_{i\in[n_x]} \exp(x_i)\right) \leq \Gamma, \end{aligned}$$

where $\boldsymbol{Q},\ \boldsymbol{c},\ \boldsymbol{A}_x,\ \boldsymbol{b}_x, n_x,\ |\mathcal{J}_k|,\ |\mathcal{K}|$, and Γ are given in Table 6, and $\boldsymbol{A}'_x$ and $\boldsymbol{b}'_x$ are given by

$$\boldsymbol{A}'_x = \begin{bmatrix} -3 & 7 & 0 & -5 & 1 & 1 & 0 & 2 & -1 & 1 \\ 7 & 0 & -5 & 1 & 1 & 0 & 2 & -1 & -1 & 1 \\ 0 & -5 & 1 & 1 & 0 & 2 & -1 & -1 & -9 & 1 \\ -5 & 1 & 1 & 0 & 2 & -1 & -1 & -9 & 3 & 1 \\ 1 & 1 & 0 & 2 & -1 & -1 & -9 & 3 & 5 & 1 \\ 1 & 0 & 2 & -1 & -1 & -9 & 3 & 5 & 0 & 1 \\ 0 & 2 & -1 & -1 & -9 & 3 & 5 & 0 & 0 & 1 \\ 2 & -1 & -1 & -9 & 3 & 5 & 0 & 0 & 1 & 1 \\ -1 & -1 & -9 & 3 & 5 & 0 & 0 & 1 & 7 & 1 \\ -1 & -9 & 3 & 5 & 0 & 0 & 1 & 7 & -7 & 1 \\ -9 & 3 & 5 & 0 & 0 & 1 & 7 & -7 & -4 & 1 \\ 3 & 5 & 0 & 0 & 1 & 7 & -7 & -4 & -6 & 1 \\ 5 & 0 & 0 & 1 & 7 & -7 & -4 & -6 & -3 & 1 \\ 0 & 0 & 1 & 7 & -7 & -4 & -6 & -3 & 7 & 1 \\ 0 & 1 & 7 & -7 & -4 & -6 & -3 & 7 & 0 & 1 \\ 1 & 7 & -7 & -4 & -6 & -3 & 7 & 0 & -5 & 1 \\ 7 & -7 & -4 & -6 & -3 & 7 & 0 & -5 & 1 & 1 \\ -7 & -4 & -6 & -3 & 7 & 0 & -5 & 1 & 1 & 1 \\ -4 & -6 & -3 & 7 & 0 & -5 & 1 & 1 & 0 & 1 \\ -6 & -3 & 7 & 0 & -5 & 1 & 1 & 0 & 2 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{b}'_x = \begin{bmatrix} -5 \\ 2 \\ -1 \\ -3 \\ 5 \\ 4 \\ -1 \\ 0 \\ 9 \\ 40 \end{bmatrix}, \text{ respectively.}$$

References

Ahmadi A, Zhang J (2019) Semidefinite programming and Nash equilibria in bimatrix games. *INFORMS Journal on Computing* Published online: `https://pubsonline.informs.org/doi/abs/10.1287/ijoc.2020.0960`.

Ahmed S, Guan Y (2005) The inverse optimal value problem. *Mathematical Programming* 102(1):91–110.

Al-Khayyal F, Larsen C, Van Voorhis T (1995) A relaxation method for nonconvex quadratically constrained quadratic programs. *Journal of Global Optimization* 6(3):215–230.

Alarie S, Audet C, Jaumard B, Savard G (2001) Concavity cuts for disjoint bilinear programming. *Mathematical Programming* 90(2):373–398.

Anstreicher K (2012) On convex relaxations for quadratically constrained quadratic programming. *Mathematical Programming* 136(2):233–251.

Ardestani-Jaafari A, Delage E (2016) Robust optimization of sums of piecewise linear functions with application to inventory problems. *Operations Research* 64(2):474–494.

Ardestani-Jaafari A, Delage E (2017) The value of flexibility in robust location–transportation problems. *Transportation Science* 52(1):189–209.

Ardestani-Jaafari A, Delage E (2019) Linearized robust counterparts of two-stage robust optimization problems with applications in operations management. *INFORMS Journal on Computing,* Ahead of Print.

Atamtürk A, Zhang M (2007) Two-stage robust network flow and design under demand uncertainty. *Operations Research* 55(4):662–673.

Audet C, Hansen P, Jaumard B, Savard G (1999) A symmetrical linear maxmin approach to disjoint bilinear programming. *Mathematical Programming* 85(3):573–592.

Avis D, Rosenberg G, Savani R, von Stengel B (2010) Enumeration of Nash equilibria for two-player games. *Economic Theory* 42(1):9–37.

Ayoub J, Poss M (2016) Decomposition for adjustable robust linear optimization subject to uncertainty polytope. *Computational Management Science* 13:219–239.

Beck A, Ben-Tal A (2009) Duality in robust optimization: Primal worst equals dual best. *Operations Research Letters* 37(1):1–6.

Ben-Tal A, den Hertog D (2014) Hidden conic quadratic representation of some nonconvex quadratic optimization problems. *Mathematical Programming* 143(1-2):1–29.

Ben-Tal A, den Hertog D, Vial JP (2015) Deriving robust counterparts of nonlinear uncertain inequalities. *Mathematical Programming* 149(1):265–299.

Ben-Tal A, El Ghaoui L, Nemirovski A (2009) *Robust Optimization* (Princeton Series in Applied Mathematics. Princeton University Press, Princeton, NJ).

Ben-Tal A, Golany B, Nemirovski A, Vial JP (2005) Retailer-supplier flexible commitments contracts: a robust optimization approach. *Manufacturing & Service Operations Management* 7(3):248–271.

Ben-Tal A, Goryashko A, Guslitzer E, Nemirovski A (2004) Adjustable robust solutions of uncertain linear programs. *Mathematical Programming* 99(2):351–376.

Bertsimas D, Bidkhori H (2015) On the performance of affine policies for two-stage adaptive optimization: a geometric perspective. *Mathematical Programming* 153(2):577–594.

Bertsimas D, de Ruiter F (2016) Duality in two-stage adaptive linear optimization: faster computation and stronger bounds. *INFORMS Journal on Computing* 28(3):500–511.

Bertsimas D, Dunning I (2016) Multistage robust mixed integer optimization with adaptive partitions. *Operations Research* 64(4):980–998.

Bertsimas D, Dunning I, Lubin M (2015) Reformulation versus cutting-planes for robust optimization. *Computational Management Science* 13(2):195–217.

Bertsimas D, Goyal V (2012) On the power and limitations of affine policies in two-stage adaptive optimization. *Mathematical Programming* 134(2):491–531.

Bertsimas D, Iancu D, Parrilo P (2010) Optimality of affine policies in multistage robust optimization. *Mathematics of Operations Research* 35(2):363–394.

Bertsimas D, Iancu D, Parrilo P (2011) A hierarchy of near-optimal policies for multistage adaptive optimization. *IEEE Transactions on Automatic Control* 56(12):2809–2824.

Bertsimas D, Litvinov E, Sun X, Zhao J, Zheng T (2013) Adaptive robust optimization for the security constrained unit commitment problem. *IEEE Transactions on Power Systems* 28(1):52–63.

Bertsimas D, Sim M, Zhang M (2019) Adaptive distributionally robust optimization. *Management Science* 65(2):604–618.

Bonami P, Lodi A, Tramontani A, Wiese S (2015) On mathematical programming with indicator constraints. *Mathematical Programming* 151(1):191–223.

Calafiore G (2008) Multi-period portfolio optimization with linear control policies. *Automatica* 44(10):2463–2473.

Calafiore G (2009) An affine control method for optimal dynamic asset allocation with transaction costs. *SIAM Journal on Control and Optimization* 48(4):2254–2274.

Ceria S, Soares J (1999) Convex programming for disjunctive convex optimization. *Mathematical Programming* 86(3):595–614.

Chen X, Deng X (2006) Settling the complexity of two-player Nash equilibrium. *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, 261–272.

Chen X, Sim M, Sun P, Zhang J (2008) A linear decision-based approximation approach to stochastic programming. *Operations Research* 56(2):344–357.

Chen X, Zhang Y (2009) Uncertain linear programs: extended affinely adjustable robust counterparts. *Operations Research* 57(6):1469–1482.

Codenotti B, De Rossi S, Pagan M (2008) An experimental analysis of Lemke-Howson algorithm. *arXiv preprint arXiv:0811.3247* .

Combettes P (2018) Perspective functions: properties, constructions, and examples. *Set-Valued and Variational Analysis* 26(2):247–264.

Ćustić A, Sokol V, Punnen AP, Bhattacharya B (2017) The bilinear assignment problem: Complexity and polynomially solvable special cases. *Mathematical Programming* 1(2):185–205.

Dantzig G (1963) *Linear Programming and Extensions* (Princeton University Press, Princeton, NJ).

de Ruiter F, Ben-Tal A (2017) Tractable nonlinear decision rules for robust optimization. Working paper.

de Ruiter F, Zhen J, den Hertog D (2019) Dual approach to two-stage nonlinear robust optimization. *Optimization Online* URL `http://www.optimization-online.org/DB_HTML/2018/03/6522.html`.

El Housni O, Goyal V (2020) On the optimality of affine polices for budgeted uncertainty sets. *Mathematics of Operations Research,* Articles in Advance.

Firouzbakht K, Noubir G, Salehi M (2016) Linearly constrained bimatrix games in wireless communications. *IEEE Transactions on Communications* 64(1):429–440.

Floudas C, Pardalos P (2008) *Encyclopedia of Optimization* (Springer Science & Business Media).

Fukuda K (2013) *Polyhedral Computation* (Lecture notes, ETH Zurich).

Gabrel V, Lacroix M, Murat C, Remli N (2014) Robust location transportation problems under uncertain demands. *Discrete Applied Mathematics* 164(1):100–11.

Gallo G, Ülkücü A (1977) Bilinear programming: an exact algorithm. *Mathematical Programming* 12(1):173–194.

Geoffrion A (1972) Generalized Benders decomposition. *Journal of Optimization Theory and Applications* 10(4):237–260.

Georghiou A, Tsoukalas A, Wiesemann W (2019) A primal-dual lifting scheme for two-stage robust optimization. *Operations Research* 68(2):1–19.

Golub G, van Loan C (1989) *Matrix Computations* (London, UK: Johns Hopkins University Press), 2 edition.

Gorissen B, Ben-Tal A, Blanc H, den Hertog D (2014) Technical note – Deriving robust and globalized robust solutions of uncertain linear programs with general convex uncertainty sets. *Operations Research* 62(3):672–679.

Gorissen B, den Hertog D (2013) Robust counterparts of inequalities containing sums of maxima of linear functions. *European Journal of Operational Research* 227(1):30–43.

Grossmann I, Lee S (2003) Generalized convex disjunctive programming: nonlinear convex hull relaxation. *Computational Optimization and Applications* 26(1):83–100.

Günlük O, Linderoth J (2010) Perspective reformulations of mixed integer nonlinear programs with indicator variables. *Mathematical Programming* 124(1-2):183–205.

Guslitzer E (2002) *Uncertainty-Immunized Solutions in Linear Programming*. Master's thesis, Technion-Israel Institute of Technology.

Hadjiyiannis M, Goulart P, Kuhn D (2011) A scenario approach for estimating the suboptimality of linear decision rules in two-stage robust optimization. *Proceedings IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 7386–7391.

Hanasusanto G, Kuhn D (2018) Conic programming reformulations of two-stage distributionally robust linear programs over Wasserstein balls. *Operations Research* 66(3):1–21.

Hanasusanto G, Kuhn D, Wiesemann W (2014) K-adaptability in two-stage robust binary programming. *Operations Research* 63(4):877–891.

Hijazi H, Bonami P, Cornuéjols G, Ouorou A (2012) Mixed-integer nonlinear programs featuring "on/off" constraints. *Computational Optimization and Applications* 52:537–558.

Iancu D, Sharma M, Sviridenko M (2013) Supermodularity and affine policies in dynamic robust optimization. *Operations Research* 61(4):941–956.

Jaillet P, Jena S, Ng T, Sim M (2016) Satisficing awakens: models to mitigate uncertainty. *Optimization Online* URL `http://www.optimization-online.org/DB_HTML/2016/01/5310.html`.

Jiang R, Li D (2016) Simultaneous diagonalization of matrices and its applications in quadratically constrained quadratic programming. *SIAM Journal on Optimization* 26(3):1649–1668.

Jiang R, Li D (2019) Convex relaxations with second order cone constraints for nonconvex quadratically constrained quadratic programming. *Journal of Global Optimization* 75:461–494.

Konno H (1971a) Bilinear programming: Part I. algorithm for solving bilinear programs. Technical report, DTIC Document.

Konno H (1971b) Bilinear programming: Part II. application of bilinear programming. Technical report, DTIC Document.

Konno H (1976) A cutting plane algorithm for solving bilinear programs. *Mathematical Programming* 11(1):14–27.

Lemke CE, Howson JT Jr (1964) Equilibrium points of bimatrix games. *Journal of the Society for Industrial and Applied Mathematics* 12(2):413–423.

Lim C, Smith J (2007) Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions* 39(1):15–26.

Lipp T, Boyd S (2016) Variations and extension of the convex–concave procedure. *Optimization and Engineering* 17(2):263–287.

Löfberg J (2004) YALMIP : A toolbox for modeling and optimization in MATLAB. *In Proceedings of the CACSD Conference* (Taipei, Taiwan).

Maher J, Fischer T, Gally T, Gamrath G, Gleixner A, Gottwald R, Hendel G, Koch T, Lübbecke M, Miltenberger M, Müller B, Pfetsch M, Puchert C, Rehfeldt D, Schenker S, Schwarz R, Serrano F, Shinano Y, Weninger D, Witt J, Witzig J (2017) The SCIP optimization suite 4.0. Technical Report 17-12, ZIB, Takustr.7, 14195 Berlin.

Mangasarian O, Stone H (1964) Two-person nonzero-sum games and quadratic programming. *Journal of Mathematical Analysis and Applications* 9(3):348–355.

McCormick G (1976) Computability of global solutions to factorable nonconvex programs: Part I–convex underestimating problems. *Mathematical Programming* 10(1):147–175.

Moehle N, Boyd S (2015) A perspective-based convex relaxation for switched-affine optimal control. *Systems & Control Letters* 86:34–40.

MOSEK ApS (2017) *The MOSEK optimization toolbox for MATLAB manual. Version 8.1.* URL `http://docs.mosek.com/8.1/toolbox.pdf`.

Nahapetyan AG (2008) Bilinear programming. *Encyclopedia of Optimization*, 279–282 (Springer).

Ng T, Sy C (2014) An affine adjustable robust model for generation and transmission network planning. *International Journal of Electrical Power & Energy Systems* 60:141–152.

Ordóñez F, Zhao J (2007) Robust capacity expansion of network flows. *Networks* 50(2):136–145.

Park J, Boyd S (2017) General heuristics for nonconvex quadratically constrained quadratic programming. *arXiv preprint arXiv:1703.07870* .

Postek K, den Hertog D (2016) Multi-stage adjustable robust mixed-integer optimization via iterative splitting of the uncertainty set. *INFORMS Journal on Computing* 28(3):553–574.

Rebennack S, Nahapetyan A, Pardalos P (2009) Bilinear modeling solution approach for fixed charge network flow problems. *Optimization Letters* 3:347–355.

Rocha P, Kuhn D (2012) Multistage stochastic portfolio optimisation in deregulated electricity markets using linear decision rules. *European Journal of Operational Research* 216(2):397–408.

Rockafellar R (1970) *Convex Analysis.* (Princeton University Press).

Roos E, den Hertog D, Ben-Tal A, de Ruiter F, Zhen J (2018) Tractable approximation of hard uncertain optimization problems. *Optimization Online* URL `http://www.optimization-online.org/DB_HTML/2018/06/6679.html`.

Sahinidis N, Grossmann E (1991) Convergence properties of generalized benders decomposition. *Computers and Chemical Engineering* 15(7):481–491.

See CT, Sim M (2009) Robust approximation of multiperiod inventory management. *Operations Research* 58(3):583–594.

Selvi A, Ben-Tal A, Brekelmans R, den Hertog D (2020) Convex maximization via adjustable robust optimization. *Optimization Online* .

Sherali H, Alameddine A (1992) A new reformulation-linearization technique for bilinear programming problems. *Journal of Global Optimization* 2(4):379–410.

Sherali H, Liberti L (2008) Reformulation-linearization methods for global optimization. Pardalos P, Floudas C, eds., *Encyclopedia of Optimization*, 3263–3268 (Springer, Berlin).

Sherali H, Shetty C (1980) A finitely convergent algorithm for bilinear programming problems using polar cuts and disjunctive face cuts. *Mathematical Programming* 19(1):14–31.

Simchi-Levi D, Wang H, Wei Y (2019) Constraint generation for two-stage robust network flow problems. *INFORMS Journal on Optimization* 1(1):49–70.

Soland R (1974) Optimal facility location with concave costs. *Operations Research* 22(2):373–382.

Stursberg O, Panek S (2002) *Hybrid Systems: Computation and Control. HSCC 2002.*, volume 2289 of *Lecture Notes in Computer Science*, chapter Control of Switched Hybrid Systems Based on Disjunctive Formulations, 421–435 (Springer, Berlin, Heidelberg).

Thieu T (1988) A note on the solution of bilinear programming problems by reduction to concave minimization. *Mathematical Programming* 41(1-3):249–260.

Vaish H (1974) *Nonconvex Programming with Applications to Production and Location Problems*. Phd thesis, Georgia Institute of Technology, URL `https://smartech.gatech.edu/bitstream/handle/1853/23351/vaish_harish_197412_phd_61551.pdf`.

Vaish H, Shetty C (1976) The bilinear programming problem. *Naval Research Logistics (NRL)* 23(2):303–309.

Wiesemann W, Kuhn D, Rustem B (2012) Robust resource allocations in temporal networks. *Mathematical Programming* 135(1):437–471.

Xu G, Burer S (2018) A copositive approach for two-stage adjustable robust optimization with uncertain right-hand sides. *Computational Optimization and Applications* 70(1):33–59.

Zeng B, Zhao L (2013) Solving two-stage robust optimization problems using a column-and-constraint generation method. *Operations Research Letters* 41(5):457–461, ISSN 0167-6377.

Zhen J, de Moor D, den Hertog D (2020a) An extension of the reformulation-linearization technique to nonlinear optimization problems. *Working Paper* .

Zhen J, de Ruiter F, Roos E, den Hertog D (2020b) Robust optimization for models with uncertain SOC and SDP constraints. *INFORMS Journal on Computing,* Articles in Advance.

Zhen J, den Hertog D (2017) Computing the maximum volume inscribed ellipsoid of a polytopic projection. *INFORMS Journal on Computing* 30(1):31–42.

Zhen J, den Hertog D, Sim M (2017) Adjustable robust optimization via Fourier-Motzkin elimination. *Operations Research* 66(4):1086–1100.

Zhen J, Kuhn D, Wisemann W (2020c) Distributionally robust nonlinear optimization. *Working Paper* .

Zheng X, Sun X, Li D (2011) Nonconvex quadratically constrained quadratic programming: best DC decompositions and their SDP representations. *Journal of Global Optimization* 50(4):695–712.