Research Reports on Mathematical and Computing Sciences
Series B : Operations Research

Department of Mathematical and Computing Sciences
Tokyo Institute of Technology
2-12-1 Oh-Okayama, Meguro-ku, Tokyo 152-8552 Japan

Branch-and-Cut Algorithms for the Bilinear Matrix Inequality Eigenvalue Problem

Mituhiro Fukuda †
e-mail: *mituhiro@is.titech.ac.jp*

Masakazu Kojima
e-mail: *kojima@is.titech.ac.jp*

April 1999, B-351

Abstract The optimization problem with the Bilinear Matrix Inequality (BMI) is one of the problems which have greatly interested researchers of the control and system theory in the last few years. This inequality permits to reduce in a elegant way various problems of robust control into its form. However, on the contrary of the Linear Matrix Inequality (LMI) which can be solved by interior-point-methods, the BMI is a computationally difficult object in theory and in practice.

This article improves the branch-and-bound algorithm of Goh, Safonov and Papavassilopoulos (1995) by applying a better convex relaxation of the BMI Eigenvalue Problem (BMIEP), and proposes new Branch-and-Bound and Branch-and-Cut Algorithms. Numerical experiments were conducted in a systematic way over randomly generated problems, and they show the robustness and the efficiency of the proposed algorithms.

Keywords: Bilinear Matrix Inequality, Branch-and-Cut Algorithm, Convex Relaxation, Cut Polytope.

† Author supported by the Ministry of Education, Science, Sports and Culture of Japan.

1 Introduction

Problems involving *Bilinear Matrix Inequalities* (BMIs) have been recently a focus of attention in the robust control theory as well as in mathematical programming [13, 14, 21]. This paper analyses the inherent bilinear structure of BMIs, and proposes Branch-and-Bound and Branch-and-Cut Algorithms to solve a specific problem in the robust control theory known as the *Bilinear Matrix Inequality Eigenvalue Problem* (BMIEP) [14]. Numerical comparisons between the new algorithms and some previously known algorithms are also provided over a large number of randomly generated test problems.

In the robust control theory, problems involving BMIs come out naturally when controllers for dynamical physical systems, *e.g.*, satellites, aircrafts or disk drivers, are designed in order to guarantee worst-case stability and performance in face of modeling uncertainties and disturbance inputs. This modeling process is known as the *robust controller synthesis*. It has been shown that a wide range of difficult control problems are reducible to problems involving BMIs [2, 12, 13]. Considering some examples, we have the μ/K_m-synthesis, the decentralized control, the constrained order $\mathcal{H}^\infty$ synthesis, the synthesis of one controller for multiple plants, *etc.*. Whereas the synthesis of robust controllers can be modeled using BMIs, the analysis of controllers, *i.e.*, the performance and stabilization degrade under modeling uncertainties or parameter variations, is reducible to relatively easy problems involving LMIs [14]. BMIs can be view as bilinear extensions of LMIs. LMIs are convex constraints [5], and they turn out to be constraints of Semidefinite Programs (SDPs) [32] which can be efficiently solved by interior-point methods [22].

Let us now define our problem. Given the symmetric matrices $\boldsymbol{B}_{ij} \in \mathbb{R}^{k\times k}$ $(i = 0, 1, \cdots, n;\ j = 0, 1, \cdots, m)$, we call the *Bilinear Matrix Inequality* an inequality for which the biaffine combination of the given matrices is a negative (or positive) semidefinite (or definite) matrix.

$$\mathbb{R}^{k\times k} \ni \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n}\sum_{j=1}^{m} x_i y_j \boldsymbol{B}_{ij} + \sum_{i=1}^{n} x_i \boldsymbol{B}_{i0} + \sum_{j=1}^{m} y_j \boldsymbol{B}_{0j} + \boldsymbol{B}_{00} \preceq \boldsymbol{0} \tag{1}$$

This inequality is considerably general, since once $\boldsymbol{B}_{ij}$'s are diagonals, it becomes a bilinear constraint. Since any nonconvex quadratic constraint can be reduced to a bilinear one [25], it includes quadratic inequalities as well. Thus, BMIs are nonconvex constraints. Furthermore, BMIs have a structure similar to bilinear programs known as biconvexity. The BMI constraint becomes convex once either of the variable $\boldsymbol{x}$ or $\boldsymbol{y}$ is fixed (See [14] for some numerical examples). Unfortunately, BMIs are really complex, and hard to be solved computationally. It is known that even the *BMI Feasibility Problem*, *i.e.*, the problem to find a point $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^n \times \mathbb{R}^m$ which satisfies (1), is an $\mathcal{NP}$-hard problem [29]. In this paper, we deal with a slight general formulation of the BMI Feasibility Problem known as the *BMI Eigenvalue Problem* (BMIEP) in the literature [14]:

$$\text{(BMIEP)} \quad \left\{ \begin{array}{ll} \min & \lambda \\ \text{s.t.} & \lambda \boldsymbol{I} - \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{y}) \succeq \boldsymbol{0} \\ & (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{H} \end{array} \right. \tag{2}$$

where

$$\mathcal{H} = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^n \times \mathbb{R}^m : \underline{\boldsymbol{x}} \le \boldsymbol{x} \le \overline{\boldsymbol{x}},\quad \underline{\boldsymbol{y}} \le \boldsymbol{y} \le \overline{\boldsymbol{y}}\}$$

with $\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}} \in \mathbb{R}^n$, $\underline{\boldsymbol{y}}, \overline{\boldsymbol{y}} \in \mathbb{R}^m$ being given constant vectors. The minimum of the maximum eigenvalue of the biaffine combination of the matrices $\boldsymbol{B}_{ij}$'s is sought in the BMIEP. As a

consequence, the BMI Feasibility Problem is solvable if and only if the optimal value of the BMIEP is non-positive if we restrict our domain to the hyper-rectangle $\mathcal{H}$. In our understanding, the BMIEP is fundamental to comprehend the hardness of BMIs, and their particular bilinear structure. In general, some problems arising from the robust control theory are formulated in a more general context as an optimization problem with a linear objective function and a restriction of the form (1) [2, 3]. We preferred, though, to analyze the above BMIEP, because we always can assume its feasibility on the contrary of the former problem which we can not (the feasible region could be disconnected also).

Some previous research related to BMIs are briefly described. It does not cover all recent works in this topic, but gives an idea about what has been already done. The first paper that formally introduced the term BMI in the control theory is probably due to Safonov, Goh and Ly [24] in 1994. Later, Goh *et al.* [14] presented the first implemented algorithm for the BMIEP. They proposed a convex relaxation of the problem, which becomes an SDP, and a branch-and-bound algorithm to obtain a global optimum. The same authors also proposed two algorithms to approximate local optima. Other branch-and-bound algorithms were proposed by VanAntwerp [31], and Fujioka and Hoshijima [8] who essentially improved the lower bound through a better convex relaxation of the BMIEP. Kawanishi *et al.* [17] and Takano *et al.* [27] gave results which reduce the feasible region of the problem using informations of local optimal solutions.

The D. C. optimization techniques were used by Tuan *et al.* [30] on the BMI Feasibility Problem. Further, a sequence of concave minimization problems and D. C. programs were employed by Liu and Papavassilopoulos [20] to solve the same problem. They also reported some numerical results using serial and parallel implementations of their algorithms.

Theoretical aspects of the BMI were studied by Mesbahi and Papavassilopoulos [21]. They established equivalent formulations for the BMI Feasibility Problem by generalizations of the Linear Programming and Linear Complementarity Problem over certain conic spaces of matrices.

More recently, the generalized Benders decomposition for bilinear and biconvex programmings [7] has been extended to BMI problems with linear objective functions. Essentially, lower bounds of a minimization problem were obtained by solving several subproblems coming from the Lagrangian relaxation of the BMI problem, which can be LP or SDP [2, 3, 33].

Although several algorithms have been proposed for BMI problems so far, they are either too theoretical to be implemented or only small instances could be solved. This paper's main concern is to report an improvement of some previous algorithms mostly by improving the convex relaxations of the BMI added to heuristic techniques for branch-and-bound and branch-and-cut algorithms. Computational results for each instance of the BMIEP are also reported.

Section 2 starts discussing convex relaxations of the BMIEP. Some theorems are proved by using results of a well-known polytope in graph theory, and the details are left for the Appendix. The second part of Section 2 is devoted to describe two algorithms to approximate local optimal solutions of the BMIEP. Section 3 explains the proposed Branch-and-Bound Algorithm, the Branch-and-Cut Algorithm and the heuristic procedures incorporated in them. Finally, Section 4 gives comparative numerical results using these algorithm, and Section 5 the conclusion.

1.1 Notation

$\mathcal{S}^k$	space of $k \times k$ symmetric matrices
$\boldsymbol{a} \geq \boldsymbol{b}$	component-wise inequality between vectors
$\boldsymbol{A} \succeq \boldsymbol{0}$	$\boldsymbol{A}$ is a positive semidefinite matrix
$\boldsymbol{A} \succeq \boldsymbol{B}$	$\boldsymbol{A} - \boldsymbol{B}$ is a positive semidefinite matrix
$\mathbf{vec}(\boldsymbol{A})$	vector formed by stacking the columns of matrix $\boldsymbol{A}$
$\lambda_{\max}[\boldsymbol{A}]$	maximum eigenvalue of matrix $\boldsymbol{A}$
$[\mathcal{A}]$	convex hull of set $\mathcal{A}$
$\mathcal{I}_n$	$\{1, 2, \cdots, n\}$
$\mathcal{J}_m$	$\{1, 2, \cdots, m\}$

2 Preliminaries

We start discussing some theoretical aspects of convex relaxations of the BMIEP (2) which will be employed for the lower bound computation. In Section 2.2, some heuristic algorithms to approximate local optima will be explained, and they will provide the upper bounds for the BMIEP. Lower and upper bounds of the BMIEP will be utilized in the Branch-and-Bound and Branch-and-Cut Algorithms proposed later in Section 3. All the discussion will consider the BMIEP, though it will also be valid for any subproblem of the branch-and-bound algorithms.

2.1 Convex Relaxation of the BMIEP

Convex (linear) relaxation is one of the techniques frequently utilized in 0-1 integer programming and nonlinear programming [18, 19, 23, 25, 34]. Several authors have utilized this technique to obtain an LMI (SDP) relaxation of the BMIEP [8, 14, 31]. Their approach requires the following equivalent formulation for the BMIEP.

Consider the BMIEP (2) and the matrix function (1). Letting $w_{ij} = x_i y_j \quad (i \in \mathcal{I}_n; \quad j \in \mathcal{J}_m)$, we obtain:

$$(\text{BMIEP}_{\mathcal{F}_W}) \quad \left\{ \begin{array}{ll} \min & \lambda \\ \text{s.t.} & \lambda \boldsymbol{I} - \boldsymbol{B}_L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \succeq \boldsymbol{0} \\ & (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathcal{F}_W \end{array} \right. \tag{3}$$

where

$$\boldsymbol{B}_L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) = \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \boldsymbol{B}_{ij} + \sum_{i=1}^{n} x_i \boldsymbol{B}_{i0} + \sum_{j=1}^{m} y_j \boldsymbol{B}_{0j} + \boldsymbol{B}_{00},$$

$$\mathcal{F}_W = \{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n \times m} : (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{H}, \quad \boldsymbol{W} = \boldsymbol{x}\, \boldsymbol{y}^T\}. \tag{4}$$

From now on, we denote by $\text{BMIEP}_{\mathcal{A}}$ the above problem when we restrict the domain of $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W})$ to the set $\mathcal{A} \subseteq \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n \times m}$, and by $v(BMIEP_{\mathcal{A}})$ the optimal value of the $BMIEP_{\mathcal{A}}$. Hence, we have $\text{BMIEP} \equiv \text{BMIEP}_{\mathcal{F}_W}$ in our notation.

Notice that writing our problem in the form (3), we can interpret it as a minimization problem of a linear function with a linear semidefinite constraint $(\lambda \boldsymbol{I} - \boldsymbol{B}_L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \succeq \boldsymbol{O})$ over the nonconvex set $\mathcal{F}_W$.

A natural approach to relax this problem is to search for a convex set which includes and closely approximates $\mathcal{F}_W$. Replacing $\mathcal{F}_W$ by such a set, the optimization problem becomes convex and therefore tractable [22]. Moreover, if this set is described by linear inequalities, the convex relaxed problem becomes an SDP [32], and hence, efficiently solvable by currently available software [4, 9, 26, 28].

Goh *et al.* [14] proposed the (convex) polytope $\widetilde{\mathcal{G}}_W$ to approximate $\mathcal{F}_W$:

$$\widetilde{\mathcal{G}}_W = \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n\times m} : \begin{array}{l} (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{H}, \\ \underline{w}_{ij} \le w_{ij} \le \overline{w}_{ij}, \quad i \in \mathcal{I}_n, \quad j \in \mathcal{J}_m \end{array} \right\}$$

where

$$\begin{array}{l} \underline{w}_{ij} = \min\{\underline{x}_i \underline{y}_j, \underline{x}_i \overline{y}_j, \overline{x}_i \underline{y}_j, \overline{x}_i \overline{y}_j\} \\ \overline{w}_{ij} = \max\{\underline{x}_i \underline{y}_j, \underline{x}_i \overline{y}_j, \overline{x}_i \underline{y}_j, \overline{x}_i \overline{y}_j\} \end{array} , \quad i \in \mathcal{I}_n, \quad j \in \mathcal{J}_m.$$

Later, Fujioka and Hoshijima [8] and VanAntwerp [31] improved this result introducing the polytope $\widetilde{\mathcal{F}}_W$ to obtain another convex relaxation of the BMIEP.

$$\widetilde{\mathcal{F}}_W = \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n\times m} : \begin{array}{l} w_{ij} \le x_i \overline{y}_j + \underline{x}_i y_j - \underline{x}_i \overline{y}_j, \\ w_{ij} \le x_i \underline{y}_j + \overline{x}_i y_j - \overline{x}_i \underline{y}_j, \\ w_{ij} \ge x_i \underline{y}_j + \underline{x}_i y_j - \underline{x}_i \underline{y}_j, \\ w_{ij} \ge x_i \overline{y}_j + \overline{x}_i y_j - \overline{x}_i \overline{y}_j, \\ i \in \mathcal{I}_n, \quad j \in \mathcal{J}_m \end{array} \right\}. \tag{5}$$

The relation between these sets is stated as follows:

Proposition 2.1 [8] *$\mathcal{F}_W \subset \widetilde{\mathcal{F}}_W \subset \widetilde{\mathcal{G}}_W$, and therefore, $v(BMIEP_{\mathcal{F}_W}) \ge v(BMIEP_{\widetilde{\mathcal{F}}_W}) \ge v(BMIEP_{\widetilde{\mathcal{G}}_W})$. Also, the $BMIEP_{\widetilde{\mathcal{G}}_W}$ and the $BMIEP_{\widetilde{\mathcal{F}}_W}$ are SDPs.*

In fact, the inequalities (5) were first proposed by Al-Khayyal and Falk [1] when they derived the expressions for the convex envelop of the function $\boldsymbol{b}(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{vec}(\boldsymbol{x}\ \boldsymbol{y}^T)$ over the hyper rectangle $\mathcal{H} \subset \mathbb{R} \times \mathbb{R}$ (with $n = m = 1$). Also, the same inequalities can be obtained by the "new reformulation-linearization technique" (RLT) [25] proposed by Sherali and Alameddine for bilinear programming.

Notice that, although $\widetilde{\mathcal{F}}_W$ gives a better approximation of $\mathcal{F}_W$ than $\widetilde{\mathcal{G}}_W$, the number of inequalities to represent the polytope $\widetilde{\mathcal{F}}_W$ increases by a constant factor.

An interesting result follows. It shows that we can not obtain a better convex approximation of $\mathcal{F}_W$ whenever $\min\{n, m\} = 1$.

Theorem 2.2 *$\widetilde{\mathcal{F}}_W = [\mathcal{F}_W]$ whenever $\min\{n, m\} = 1$.*

This theorem follows from the proof of Theorem 2.5. See also [11] for an alternative proof using arguments of totally unimodular matrices.

From Theorem 2.2, it remains only to analyze the case $n, m \ge 2$. We propose a new polytope $\widetilde{\mathcal{E}}_W$ to approximate the nonconvex set $\mathcal{F}_W$:

$$\widetilde{\mathcal{E}}_W = \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \widetilde{\mathcal{F}}_W : \begin{array}{l} a_{ikjl} \le e(x_i, x_k, y_j, y_l, w_{ij}, w_{il}, w_{kj}, w_{kl}) \le b_{ikjl} \\ i,k \in \mathcal{I}_n, \quad i \ne k, \quad j,l \in \mathcal{J}_m, \quad j \ne l \end{array} \right\} \tag{6}$$

where

$$\begin{aligned} a_{ikjl} &= (\overline{y}_j \underline{y}_l \overline{x}_i - \underline{y}_j \overline{y}_l \underline{x}_i)(\overline{x}_k - \underline{x}_k) + (\overline{y}_j \overline{y}_l \underline{x}_k - \underline{y}_j \underline{y}_l \overline{x}_k)(\overline{x}_i - \underline{x}_i), \\ b_{ikjl} &= (\overline{y}_j \underline{y}_l \underline{x}_i - \underline{y}_j \overline{y}_l \overline{x}_i)(\overline{x}_k - \underline{x}_k) + (\overline{y}_j \overline{y}_l \overline{x}_k - \underline{y}_j \underline{y}_l \underline{x}_k)(\overline{x}_i - \underline{x}_i), \\ e(x_i, x_k, y_j, y_l, w_{ij}, w_{il}, w_{kj}, w_{kl}) &= (\overline{y}_j \underline{y}_l - \underline{y}_j \overline{y}_l)(\overline{x}_k - \underline{x}_k) x_i + (\overline{y}_j \overline{y}_l - \underline{y}_j \underline{y}_l)(\overline{x}_i - \underline{x}_i) x_k \\ &+ (\overline{x}_i \underline{x}_k - \underline{x}_i \overline{x}_k)(\overline{y}_l - \underline{y}_l) y_j + (\overline{x}_i \overline{x}_k - \underline{x}_i \underline{x}_k)(\overline{y}_j - \underline{y}_j) y_l \\ &+ (\overline{x}_k - \underline{x}_k)(\overline{y}_l - \underline{y}_l) w_{ij} - (\overline{x}_k - \underline{x}_k)(\overline{y}_j - \underline{y}_j) w_{il} \\ &- (\overline{x}_i - \underline{x}_i)(\overline{y}_l - \underline{y}_l) w_{kj} - (\overline{x}_i - \underline{x}_i)(\overline{y}_j - \underline{y}_j) w_{kl}. \end{aligned} \tag{7}$$

Notice further that

$$b_{ikjl} - a_{ikjl} = (\overline{x}_i - \underline{x}_i)(\overline{x}_k - \underline{x}_k)(\overline{y}_j - \underline{y}_j)(\overline{y}_l - \underline{y}_l), \qquad \begin{array}{l} i,k \in \mathcal{I}_m, \quad i \ne k, \\ j,l \in \mathcal{J}_m, \quad j \ne l. \end{array}$$

The analog of Proposition 2.1 is given, and it follows from the definition of $\widetilde{\mathcal{E}}_W$ and Theorem 2.5.

Proposition 2.3 *Suppose that* $n, m \ge 2$. *Then* $\mathcal{F}_W \subset \widetilde{\mathcal{E}}_W \subset \widetilde{\mathcal{F}}_W \subset \widetilde{\mathcal{G}}_W$, *and therefore,* $v(BMIEP_{\mathcal{F}_W}) \ge v(BMIEP_{\widetilde{\mathcal{E}}_W}) \ge v(BMIEP_{\widetilde{\mathcal{F}}_W}) \ge v(BMIEP_{\widetilde{\mathcal{G}}_W})$. *Also, the* $BMIEP_{\widetilde{\mathcal{E}}_W}$ *is an SDP.*

Remark 2.4 *The inclusion* $\widetilde{\mathcal{E}}_W \subset \widetilde{\mathcal{F}}_W$ *is strict. For example, consider* $n = m = 2$ *and* $\mathcal{H} = [0,1]^4$. *Then* $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}) \in \widetilde{\mathcal{F}}_W$, *but* $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \notin \widetilde{\mathcal{E}}_W$.

Theoretically, the set $\widetilde{\mathcal{E}}_W$ closely approximates the set $\mathcal{F}_W$ compared to $\widetilde{\mathcal{G}}_W$ and $\widetilde{\mathcal{F}}_W$. In fact, we can derive the following result.

Theorem 2.5 $\widetilde{\mathcal{E}}_W = [\mathcal{F}_W]$ *whenever* $\min\{n, m\} = 2$.

The proof follows from the lemma given by Yajima *et al.* [34] and results from the well-known cut polytope [6] in graph theory. See Appendix. Although the sets $\widetilde{\mathcal{F}}_W$ and $\widetilde{\mathcal{E}}_W$ are closely related to cut polytope as the proof of the theorem indicates, the inequalities (6) and (7) were originally found thanks to the insights given by the polyhedron facets-vertices generator program CDD [10].

Table 1 summarizes the number of inequalities to represent each polytope $\widetilde{\mathcal{G}}_W$, $\widetilde{\mathcal{F}}_W$ and $\widetilde{\mathcal{E}}_W$.

Table 1: Number of inequalities to represent each convex polytope

set	$\widetilde{\mathcal{G}}_W$	$\widetilde{\mathcal{F}}_W$	$\widetilde{\mathcal{E}}_W$
# inequalities	$2nm + 2(n+m)$	$4nm$	$2nm(n-1)(m-1) + 4nm$

From this table, we notice that it is impractical to directly utilize the set $\widetilde{\mathcal{E}}_W$ to relax our problem $\text{BMIEP}_{\mathcal{F}_W}$ since the total number of inequalities is of order $\mathcal{O}(n^2 m^2)$, and SDPs

of such a size are computationally expensive to be solved. Considering a numerical example with $n = m = 10$, we need 240 inequalities to define $\widetilde{\mathcal{G}}_W$, 400 to $\widetilde{\mathcal{F}}_W$, and 16600 to $\widetilde{\mathcal{E}}_W$. However, we will utilize part of the inequalities which describe the set $\widetilde{\mathcal{E}}_W$ in our Branch-and-Cut Algorithm proposed in Section 3. These cuts (inequalities) are efficient to improve the performance of the algorithm as demonstrated in the numerical experiments.

Remark 2.6 *The analysis of the convex relaxations of the BMIEP was made all about how to approximate the nonconvex set* $\mathcal{F}_W$, *and it is independent of the semidefinite constraint* $\lambda \boldsymbol{I} - \boldsymbol{B}_L(\boldsymbol{x}, \boldsymbol{y}) \succeq \boldsymbol{O}$. *In one sense, these results are interesting because they can be straightly applied to other optimization problems involving bilinear terms. However, a difficult problem of finding the convex hull of the feasible region of the BMIEP,* $\{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n \times m} : (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{H}, \lambda \boldsymbol{I} - \boldsymbol{B}_L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \succeq \boldsymbol{O}, \boldsymbol{W} = \boldsymbol{x}\,\boldsymbol{y}^T\}$ *is left as a further topic. Notice that if we could find the convex hull of the feasible region, we can obtain the optimal value of the BMIEP once we solve this convex relaxed problem since the objective function is linear.*

2.2 Approximations of the BMIEP local optima

Some heuristic algorithms to compute local optima for problems involving BMIs have been proposed in the literature. In particular, Goh *et al.* [14] addressed two algorithms. The first is based on the *Method of Centers* [15]. Although this algorithm guarantees convergence to a local optimal solution, there is no result concerning its convergence rate and it might give some numerical instability in latter iterations [12]. The second algorithm exploits the biconvex structure of BMIs. Notice that the BMIEP (2) becomes a convex problem (an SDP) once the variable $\boldsymbol{x}$ or $\boldsymbol{y}$ is fixed. We call their heuristic algorithm the *Alternating SDPs Method* and we transcribe it in Algorithm 2.7.

Algorithm 2.7 : `Alternating SDPs Method [14]`

0 Let $\delta > 0$ and $(\boldsymbol{x}^0, \boldsymbol{y}^0) \in \mathcal{H}$ given;
1 Set $\lambda^0 = \lambda_{\max}[\boldsymbol{B}(\boldsymbol{x}^0, \boldsymbol{y}^0)]$; $t = 0$;
2 **repeat**
3 $(\lambda^{t+1}, \boldsymbol{x}^{t+1}) = \arg\min\{\lambda : \lambda \boldsymbol{I} - \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{y}^t) \succeq \boldsymbol{O}, \quad (\boldsymbol{x}, \boldsymbol{y}^t) \in \mathcal{H}\}$;
4 $(\lambda^{t+1}, \boldsymbol{y}^{t+1}) = \arg\min\{\lambda : \lambda \boldsymbol{I} - \boldsymbol{B}(\boldsymbol{x}^{t+1}, \boldsymbol{y}) \succeq \boldsymbol{O}, \quad (\boldsymbol{x}^{t+1}, \boldsymbol{y}) \in \mathcal{H}\}$;
5 $t := t + 1$;
6 **until** $(\lambda^{t-1} - \lambda^t < \delta|\lambda^t|)$;

The Alternating SDPs Method will be utilized in our algorithms to compute approximations of local optima of the BMIEP. On the contrary to the Method of Centers adapted for the BMIEP, the Alternating SDPs Method does not necessary converge to a local optimal solution. See [14] for numerical example. Also, notice that the steps 3 and 4 can be swapped. For our implementation, we adopted $(\boldsymbol{x}^0, \boldsymbol{y}^0) = (\boldsymbol{x}^*, \boldsymbol{y}^*) \in \mathcal{H}$ as the initial point where $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*) \in \mathcal{H} \times \mathbb{R}^{n \times m}$ is the optimal solution obtained from one of the relaxed BMIEPs explained in Section 2.1. This choice is not so rashness considering that the feasible regions of the subproblems shrink along the branch-and-bound algorithm, and then, the optimal solutions of the relaxed subproblems give close approximations of a global optimal solution.

For small instances of the BMIEP, we can even utilize a rough procedure to compute an approximation of a local optimum. Fujioka and Hoshijima [8] utilized the value $\lambda_{\max}[\boldsymbol{B}(\boldsymbol{x}^*, \boldsymbol{y}^*)]$ which is simply the maximum eigenvalue of the matrix function $\boldsymbol{B}(\cdot, \cdot)$ (1) at $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, where $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ is the optimal solution of one of the relaxed problems.

The numerical experiments, given in Section 4, show that the Alternating SDPs Method is a good heuristic to obtain an approximation of a local optimum. However, we have utilized either the Alternating SDPs Method or the Fujioka and Hoshijima's heuristic procedure to compute upper bounds of our problems depending on the instance size, since the former is relatively time consuming.

3 Branch-and-Bound and Branch-and-Cut Algorithms for the BMIEP

Branch-and-bound algorithms have been utilized by several authors to solve the BMIEP in the last few years [8, 14, 17, 27, 31]. We propose a new Branch-and-Bound Algorithm which mainly differs from the previously proposed one [8] in the branching rule (Section 3.2 - (i)), and a Branch-and-Cut Algorithm which is a variation of the former and utilizes valid cuts defined by the set $\widetilde{\mathcal{E}}_W$ (6). We start describing the general framework of the Branch-and-Cut Algorithm (or the Branch-and-Bound Algorithm which is a particular case) in Section 3.1. Section 3.2 gives the details of the algorithms such as lower and upper bounds computation, branching rule, and criterion to select a subproblem along their subroutines.

3.1 Description of the Algorithms

We begin by briefly describing the Branch-and-Cut Algorithm which is a combination of the branch-and-bound and cutting plane algorithms. Let $\epsilon > 0$ be a given admissible relative error for the algorithm stopping criterion, and P_0 be the original BMIEP (2). The Branch-and-Cut Algorithm decomposes P_0 into several subproblems, and tries to solve each of them in a prescribed order. These subproblems are represented by nodes in a rooted tree called *branching tree*. P_0 is the root node and the subproblems are posted in a hierarchical way in this tree. At each step, a subproblem P_r, which is a leaf node in the branching tree, is chosen and divided into two subproblems P_{r_1} and P_{r_2} according to the *branching rule* (Section 3.2 - (i)). This decomposition is recursively applied until each unexamined subproblem is decomposed, solved, or shown not containing an optimal solution for the problem P_0. The crucial computational task is the evaluation of the *lower bounds* $l(P_r)$ and $l_2(P_r)$, and the *upper bound* $u(P_r)$ of each subproblem P_r. Using the values of the lower bounds and upper bounds, we can eliminate subproblems which do not contain a global optimal solution. The lower bound $l(P_r)$ is the optimal value of the convex relaxed problem BMIEP$_{\widetilde{\mathcal{F}}_W^r}$ (Section 2.1) for the corresponding subproblem P_r with feasible set $\widetilde{\mathcal{F}}_W^r$; the lower bound $l_2(P_r)$ is obtained solving the same relaxed problem with a narrow feasible region (with some additional cutting planes coming from the set $\widetilde{\mathcal{E}}_W^r$ for the corresponding subproblem P_r); the upper bound $u(P_r)$ is computed according to the algorithms given in Section 2.2. In order to describe the general steps of the algorithm, the following notation is adopted. $\mathcal{A}_t$ denotes the set of *active nodes* (subproblems), *i.e.*, the nodes which were not deleted by the lower bound elimination until the iteration t. LB_t and UB_t denote the minimum lower bound among the active nodes, and the minimum upper bound found until the iteration t, respectively.

Some minor remarks should be mentioned about Algorithm 3.1. Since we are dealing with a continuous minimization problem instead of a discrete one, we can only obtain an approximate solution in finite time. $\epsilon > 0$ is the relative error parameter which is guaranteed for the approximate solution. Further, we are omitting the case when an optimal solution of the relaxed problem

Algorithm 3.1: Branch-and-Cut

```
0    Let P_0 and ε > 0 given; t = 0 ;
1    LB_t = max{l(P_t), l_2(P_t)} ;  UB_t = u(P_t) ;
2    A_t = {P_t} ;
3    while ( (UB_t − LB_t > ε|UB_t|) and (A_t ≠ ∅) )
4       Select P_r ∈ A_t ;  A_t = A_t \ {P_r} ;
5       Split P_r into P_{r1} and P_{r2} ;
6       for q = 1 to 2
7          if ( UB_t − l(P_{r_q}) > ε|UB_t| )
8             if ( UB_t − l_2(P_{r_q}) > ε|UB_t| )
9                UB_{t+1} = min{ UB_t, u(P_{r_q})} ;
10               A_t = A_t ∪ {P_{r_q}} ;
11            else
12               UB_{t+1} = UB_t ;
13            endif
14         else
15            UB_{t+1} = UB_t ;
16         endif
17      endfor
18      A_{t+1} = {P_r ∈ A_t : UB_{t+1} − max{l(P_r), l_2(P_r)} > ε|UB_{t+1}|} ;
19      LB_{t+1} = min{max{l(P_r), l_2(P_r)} : P_r ∈ A_t} ;  t = t + 1 ;
20   endwhile
```

is in fact an optimal solution of the subproblem because it is a rare phenomenon for the set of the problems solved here, and its inclusion could complicate Algorithm 3.1 unnecessary.

In Steps 18 and 19, it is possible that $l_2(\cdot)$ does not exist for some subproblems since it is not always computed. The description of the Branch-and-Bound Algorithm is omitted since it just differs in few steps compared to Algorithm 3.1: Step 8 is omitted and $l_2(P_r)$ is suppressed from Steps 1 and 19.

The convergence of Algorithm 3.1 follows since the objective function for all the convex relaxed problems given in Section 2.1, BMIEP$_{\widetilde{\mathcal{G}}_W}$, BMIEP$_{\widetilde{\mathcal{F}}_W}$ or BMIEP$_{\widetilde{\mathcal{E}}_W}$, are uniformly continuous in the feasible regions. See [14] for a formal proof.

3.2 Technicalities of the Algorithms

(i) Branching Rule

Branching rule is an important element for the performance of branch-and-bound algorithms in global optimization. Different branching rules result in different computational outcomes in practice [23]. Usually, the bisection branching rule is utilized whenever the feasible region of the subproblem is a hyper rectangle which is the case of the BMIEP (2). In this rule, the longest edge of the feasible region is split at the middle generating two equivolumetric hyper rectangles for the child problems. Previously proposed branch-and-bound algorithms for the BMIEP have all applied the bisection branching rule [8, 14, 17, 27]. The proposed new branching rule has the bisection branching rule as a special case.

Let P_r be a subproblem with the corresponding feasible region $\mathcal{H}^r = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^n \times \mathbb{R}^m : \underline{\boldsymbol{x}}^r \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}}^r, \quad \underline{\boldsymbol{y}}^r \leq \boldsymbol{y} \leq \overline{\boldsymbol{y}}^r\}$, see (2). A coordinate of the variables x_i or y_j is then chosen, and its range is divided at the middle as in the bisection rule, generating two equivolumetric hyper rectangles $\mathcal{H}^{r_1}$ and $\mathcal{H}^{r_2}$ for the child problems P_{r_1} and P_{r_2}, respectively.

The selection of the coordinate to be split is made as follows. First, in order to avoid generating narrow feasible regions for the subproblems, which lead to numerical instability, a parameter $\alpha \in (0, 1]$ is introduced. The coordinate that will be partitioned must have then its range at least α times the longest range among all variable coordinates. That is, defining $R_{\max} = \max\{ \overline{x}_i - \underline{x}_i, \overline{y}_j - \underline{y}_j : i \in \mathcal{I}_n, \quad j \in \mathcal{J}_m\}$, the selected variable coordinate must satisfy $\overline{x}_i - \underline{x}_i \geq \alpha R_{\max}, \quad i \in \mathcal{I}_n$ (or $\overline{y}_j - \underline{y}_j \geq \alpha R_{\max}, \quad j \in \mathcal{J}_m$). Notice further that the bisection branching rule is a special case of this rule for $\alpha = 1$.

Suppose now that $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ is an optimal solution of the relaxed problem BMIEP$_{\widetilde{\mathcal{F}}^r_W}$ for P_r (Figure 1 - left represents the feasible set $\widetilde{\mathcal{F}}^r_W$ of the BMIEP$_{\widetilde{\mathcal{F}}^r_W}$ when $(\boldsymbol{x}, \boldsymbol{y}) \in [-1, 1] \times [-1, 1]$). Notice that choosing an appropriate variable coordinate to be split, $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ might not be feasible for the convex relaxed problems of the child problems (Figure 1). In this case, we can expect an increase in the lower bound values for them compared to the lower bound obtained for the parents problem P_r. We adopt the following heuristic to determine one coordinate to be split. We over estimate the lower bounds for the possible child problems performing an orthogonal projection of $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ in relation to the space $\boldsymbol{x}$ - $\boldsymbol{y}$ over the feasible set of the convex relaxed problem of one of the possible child problems, $\widetilde{\mathcal{F}}^{r_1}_W$ or $\widetilde{\mathcal{F}}^{r_2}_W$. More concretely, suppose that we want to compute the estimation of the lower bound for the index $\widehat{i} \in \mathcal{I}_n$ (variable $\boldsymbol{x}$). We can consider without loss of generality that $x^*_{\widehat{i}} < x^M_{\widehat{i}} = \frac{\underline{x}^r_{\widehat{i}} + \overline{x}^r_{\widehat{i}}}{2}$. Notice that, in this case, $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ can not be a feasible point for $\widetilde{\mathcal{F}}^{r_2}_W$ since $(\boldsymbol{x}^*, \boldsymbol{y}^*) \notin \mathcal{H}^{r_2}$. See Figure 1. If $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*) \in \widetilde{\mathcal{F}}^{r_1}_W$, the lower bound for the BMIEP$_{\widetilde{\mathcal{F}}^{r_1}_W}$ will have the same value of the BMIEP$_{\widetilde{\mathcal{F}}^r_W}$ since $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ is already the optimal solution for the BMIEP$_{\widetilde{\mathcal{F}}^{r_1}_W}$. Otherwise, the projection $(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}}) \in \widetilde{\mathcal{F}}^{r_1}_W$ of $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*) \in \widetilde{\mathcal{F}}^r_W$ is computed by:

$$\widehat{\boldsymbol{W}} = \begin{cases} w^*_{kl}, & k \in \mathcal{I}_m, \quad k \neq \widehat{i}, \quad l \in \mathcal{J}_m, \\ \min\{w^*_{\widehat{i}l}, \quad x^*_{\widehat{i}} \underline{y}_l{}^r + x^M_{\widehat{i}} y^*_l - x^M_{\widehat{i}} \underline{y}_l{}^r\}, & \text{if} \quad w_{\widehat{i}l} \geq x^*_{\widehat{i}} y^*_l, \quad l \in \mathcal{J}_m, \\ \max\{w^*_{\widehat{i}l}, \quad x^*_{\widehat{i}} \overline{y}_l{}^r + x^M_{\widehat{i}} y^*_l - x^M_{\widehat{i}} \overline{y}_l{}^r\}, & \text{if} \quad w_{\widehat{i}l} < x^*_{\widehat{i}} y^*_l, \quad l \in \mathcal{J}_m. \end{cases}$$

Then,

$$\lambda_{\max}(\boldsymbol{B}_L(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}})) \quad (\geq v(\text{BMIEP}_{\widetilde{\mathcal{F}}^{r_1}_W})) \tag{8}$$

gives an estimation of the lower bound for the child problem P_{r_1} since the objective function was evaluated for a feasible point $(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}}) \in \widetilde{\mathcal{F}}^{r_1}_W$. The case $x^*_{\widehat{i}} \geq x^M_{\widehat{i}}$ is derived in a similar way.

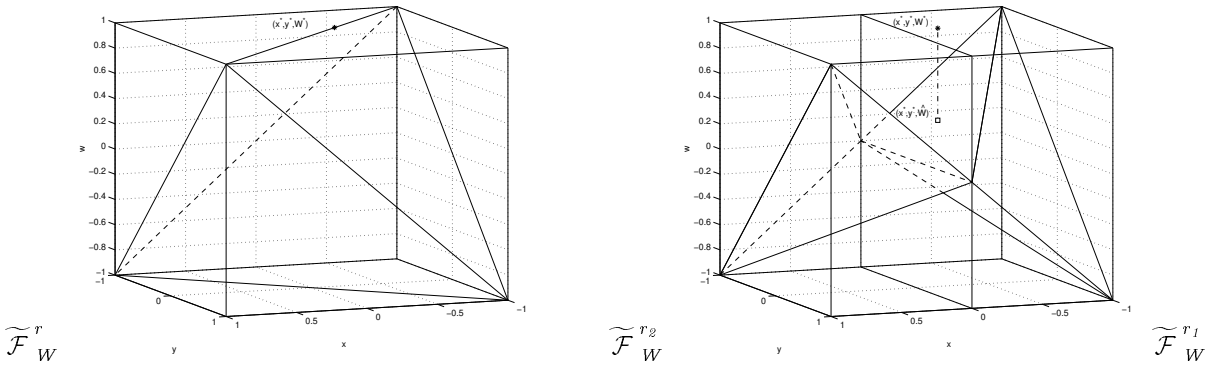
We select an index $i \in \mathcal{I}_n$ or $j \in \mathcal{J}_m$ which satisfies $\overline{x}_i - \underline{x}_i \geq \alpha R_{\max}$ or $\overline{y}_j - \underline{y}_j \geq \alpha R_{\max}$, respectively, and gives the greatest value for (8) to split at its middle point of the corresponding range.

Observe from Figures 1 that using the set $\widetilde{\mathcal{F}}^r_W$ to approximate the feasible region $\mathcal{F}^r_W$, the sum of the volumes of the child problem feasible sets is always less than the volume of the parents problem feasible set. This is another advantage of using the set $\widetilde{\mathcal{F}}^r_W$ instead of $\widetilde{\mathcal{G}}^r_W$ to approximate $\mathcal{F}^r_W$.

(ii) Lower Bound Computation

According to the branching rule adopted for our algorithms, all the subproblems P_r's have the same structure as the original BMIEP (2) only differing at the lower and upper bounds of each

Figure 1: Splitting process of the feasible set $\widetilde{\mathcal{F}}^{r}_{W}$ into $\widetilde{\mathcal{F}}^{r_1}_{W}$ and $\widetilde{\mathcal{F}}^{r_2}_{W}$



variable $\boldsymbol{x}$ and $\boldsymbol{y}$ (*i.e.*, $\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}$ and $\underline{\boldsymbol{y}}, \overline{\boldsymbol{y}}$). The lower bound $lb(P_r)$ for the subproblem P_r is obtained solving the corresponding convex relaxed problem BMIEP $_{\widetilde{\mathcal{F}}^{r}_{W}}$ which is an SDP.

The lower bound $lb_2(P_r)$ is obtained solving the BMIEP $_{\widetilde{\mathcal{F}}^{r}_{W}}$ with some additional cuts (inequalities) coming from the convex set $\widetilde{\mathcal{E}}^{r}_{W} (\subset \widetilde{\mathcal{F}}^{r}_{W})$ which are valid for the optimal solution $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ of the BMIEP $_{\widetilde{\mathcal{F}}^{r}_{W}}$. More precisely, let us denote by Φ the set of valid inequality indices for the point $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*)$ related to the set $\widetilde{\mathcal{E}}^{r}_{W}$ (6) as defined below.

Definition 3.4 *Let $\rho > 0$ be a small positive number. Then*

$$\Phi = \left\{ (i,j,k,l,s) \in \mathcal{I}_n \times \mathcal{J}_m \times \mathcal{I}_n \times \mathcal{J}_m \times \{0,1\} : \begin{array}{l} a_{ikjl} - \rho(b_{ikjl} - a_{ikjl}) > e(x_i^*, x_k^*, y_j^*, y_l^*, w_{ij}^*, w_{il}^*, w_{kj}^*, w_{kl}^*), \\ s = 0 \\ \qquad\qquad or \\ e(x_i^*, x_k^*, y_j^*, y_l^*, w_{ij}^*, w_{il}^*, w_{kj}^*, w_{kl}^*) < b_{ikjl} + \rho(b_{ikjl} - a_{ikjl}), \\ s = 1 \end{array} \right\}$$

The inequalities $a_{ikjl} \le e(\cdot) \le b_{ikjl}$ comes from $\widetilde{\mathcal{E}}^{r}_{W}$ (6), and the term $\pm\rho(b_{ikjl} - a_{ikjl})$ is a small scaling number close to zero to avoid numerical errors.

Then, we have the following feasible set for the relaxed problem with the cutting planes:

$$appr\,\widetilde{\mathcal{E}}^{r}_{W} = \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \widetilde{\mathcal{F}}^{r}_{W} : \begin{array}{ll} a_{ikjl} \le e(x_i, x_k, y_j, y_l, w_{ij}, w_{il}, w_{kj}, w_{kl}), & \forall (i,j,k,l,0) \in \Phi \\ e(x_i, x_k, y_j, y_l, w_{ij}, w_{il}, w_{kj}, w_{kl}) \le b_{ikjl}, & \forall (i,j,k,l,1) \in \Phi \end{array} \right\}$$

which obviously satisfies the inclusion $\widetilde{\mathcal{E}}^{r}_{W} \subset appr\,\widetilde{\mathcal{E}}^{r}_{W} \subset \widetilde{\mathcal{F}}^{r}_{W}$.

Remark 3.5 *One might expect that the cardinality of Φ is large, being of the same order $\mathcal{O}(n^2 m^2)$ as the number of facets of $\widetilde{\mathcal{E}}^{r}_{W}$. However, the numerical experiments have shown that the number of valid cuts per subproblem is of order $\mathcal{O}(n)$ or $\mathcal{O}(m)$ in average [11].*

Further, we might not just solve the BMIEP $_{appr\,\widetilde{\mathcal{E}}^{r}_{W}}$ for each subproblem P_r, since the improvement of the lower bound using cutting planes must compensate the cost of its computational time (which is of same order to obtain $l(P_r)$. See Remark 3.5). Therefore, we make an estimation a priori for the improvement obtained by solving the relaxed problem with the cutting planes, and we solve the subproblem P_r with the cuts if and only if it permit its own elimination by

the lower bound elimination (Step 8 of Algorithm 3.1). The estimation is evaluated as follows. First, we project $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*) \in \widetilde{\mathcal{F}}^r_W$ over the set $appr\,\widetilde{\mathcal{E}}^r_W$ along the orthogonal direction of the original variable space $\boldsymbol{x}$ - $\boldsymbol{y}$. That is, along the line $\gamma(\boldsymbol{0}, \boldsymbol{0}, \boldsymbol{x}^*(\boldsymbol{y}^*)^T - \boldsymbol{W}^*)$ $(\gamma \in \mathbb{R})$. Then, we obtain the point $(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}}) \in appr\,\widetilde{\mathcal{E}}^r_W$ such that

$$(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}}) = (\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{W}^*) + \widehat{\gamma}(\boldsymbol{0}, \boldsymbol{0}, \boldsymbol{x}^*(\boldsymbol{y}^*)^T - \boldsymbol{W}^*)$$

where $\widehat{\gamma}$ is the smallest positive number such that $(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}})$ lies at the boundary of $appr\,\widetilde{\mathcal{E}}^r_W$. Since $(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}}) \in appr\,\widetilde{\mathcal{E}}^r_W$, and it is not an optimal solution of the BMIEP$_{app\,\widetilde{\mathcal{E}}^r_W}$, we compensate this fact using a parameter $\theta \in [0, 1]$ to obtain an estimative for $l_2(P_r)$.

$$est[l_2(P_r)] = l(P_r) + \theta(\lambda_{\max}[\boldsymbol{B}_L(\boldsymbol{x}^*, \boldsymbol{y}^*, \widehat{\boldsymbol{W}})] - l(P_r))$$

Finally, we decide to solve the BMIEP$_{app\,\widetilde{\mathcal{E}}^r_W}$ if and only if

$$UB_t - est[l_2(P_r)] \leq \epsilon |UB_t| \qquad \text{(Step 8 of Algorithm 3.1)} \tag{9}$$

$l_2(P_r)$ is then given by the optimal solution of the BMIEP$_{appr\,\widetilde{\mathcal{E}}^r_W}$.

(iii) Upper Bound Computation

We utilized either the Alternating SDPs Method (Algorithm 2.7) or the heuristic procedure mentioned by Fujioka and Hoshijima [8] (Section 2.2) to compute the upper bound $ub(P_r)$ of the subproblem P_r. We introduce a parameter β which switches from one algorithm to the other during the Algorithm 3.1. Basically, the Fujioka and Hoshijima's procedure was adopted for the upper bound computation, however Algorithm 2.7 was performed at Step 1 and every β iterations of the while loop 3-20 of Algorithm 3.1. The parameter β was settled empirically to a value between *2* to *50* depending on the problem instance for the numerical experiments given in the next section.

(iv) Selection of a subproblem in the branching tree

Best-bound and depth-first search are generally used in practice for branch-and-bound algorithms. While depth-first search chooses one of the deepest nodes of the branching tree, the best-bound search chooses the node with the minimum lower bound. Here, we adopted the best-bound search which minimizes the number of subproblems prior to the termination, although it has a tendency to consume an amount of memory which is an exponential function of the problem size [23].

4 Numerical Experiments

This section regards the main part of this paper. It gives the computational results for the proposed Branch-and-Bound and Branch-and-Cut Algorithms. They show that these algorithms are promising for medium-sized optimization problems involving BMIs. Furthermore, our algorithms seem more efficient compared to some previous branch-and-bound based algorithms, specially when the dimensions of the problems increase.

As mentioned in Section 1, problems involving BMIs are extremely difficult and only few real problems were solved until the present. For example, one of the largest BMIEP which was completely solved has dimensions $n = 4, m = 6$ and $k = 6$ [8] where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{y} \in$

$\mathbb{R}^m$, $\boldsymbol{B}_{ij} \in \mathcal{S}^k$. Thus, randomly generated BMIEPs were utilized for testing and comparing the Branch-and-Bound and the Branch-and-Cut Algorithms with some previously proposed algorithms. For each n and m fixed, we generated $7 \sim 20$ BMIEPs. Each element of the matrices $\boldsymbol{B}_{ij}$ $(0 \le i \le n; \quad 0 \le j \le m)$ uniformly took a value in the range $[-100, 100]$, and the lower and upper bounds of the variables, $\underline{\boldsymbol{x}}$, $\overline{\boldsymbol{x}}$, $\underline{\boldsymbol{y}}$ and $\overline{\boldsymbol{y}}$, a value in the range $[-50, 50]$. Generating the test problems in this way, the matrices $\boldsymbol{B}_{ij}$'s become fully dense, and therefore, they become computationally more expensive compared to the sparse ones usually found in real applications in the control theory.

The new algorithms, Branch-and-Bound and Branch-and-Cut, were implemented in C++, compiled using the DEC C++ compiler cxx with the option -O3. The software SDPA 4.20 [9] was incorporated in the subroutines of the algorithms to solve the SDPs using the search direction HRVW/KSH/M and relative error parameter for the duality gap ϵ^* fixed to 10^{-4}.

The parameters for the new Branch-and-Bound and Branch-and-Cut Algorithms are summarized in Table 2.

Table 2: Parameters for the Branch-and-Bound and Branch-and-Cut Algorithms

parameter	value	algorithm	description
ϵ	0.001	B & B and B & C	relative error - stopping criterion (Algorithm 3.1)
α	0.3	B & B and B & C	controlling parameter for the feasible regions in the branching rule (Section 3.2 - (i))
θ	0.4	B & C	estimation parameter for $l_2(\cdot)$ (Section 3.2 - (ii))
δ	0.02	B & B and B & C	stopping criterion for the Alternating SDPs Method (Algorithm 2.7)
ρ	0.05	B & C	scaling parameter (Section 3.2 - (ii))
ϵ_{abs}	10^{-8}	B & B and B & C	computational zero

All the computational experiments were conducted on Digital AlphaPC 164LX (CPU Alpha 21140 - 599 MHz with 1 GB of memory) under Digital UNIX V4.0D.

First, we compare the performance of the Branch-and-Bound and the Branch-and-Cut Algorithms to the Goh *et al.*'s algorithm (1995) [14] and Fujioka and Hoshijima's algorithms (1997) [8]. From now on, we will denote these algorithms by the B & B, the B & C, the GSP and the FH, respectively. Both the GSP and the FH were rewritten in C++, and we have utilized the SDPA 4.20 to evaluate the SDPs instead of using the LMI Control Toolbox or other SDP solver. Also, we changed the stopping criterion from the absolute error to the relative error between the upper and lower bounds of the problem. The Alternating SDPs Method (Algorithm 2.7) was utilized to evaluate the upper bounds in the GSP, since the Method of Centers is computationally expensive, and the authors did not clearly mention its use [14].

We summarize our results in Tables 4 - 6. The test problems are as follows. "gsp", "twy", "goh" and "fh" are BMIEPs reported in the literature. "rand1 $\sim$ rand10" denote the pseudo-randomly generated problems. The parameter β (Section 2.2) was fixed to 50 for the problems found in the literature and according to Table 3 for the randomly generated problems.

Table 4 shows the total number of subproblems generated during the whole algorithms and the CPU time consumed for them. The GSP seems not efficient from our numerical results. It basically reflects the poor convex relaxation obtained by the BMIEP$_{\widetilde{\mathcal{G}}_W}$. Recall that the relaxation BMIEP$_{\widetilde{\mathcal{F}}_W}$ was utilized for the algorithms FH, B & B and B & C to compute the

Table 3: Parameter β for the test problems

$n = m$	$1 \sim 4$	5	6	7	8	9	10
β	50	25	15	10	6	4	2

lower bounds instead. The B & B and the B & C have similar behaviors, however it seems that the B & C is slightly superior when the dimensions of the problem increase. Comparing the FH with the B & B in particular, it seems that the new branching rule adopted in the B & B (and B & C) worked efficiently to diminish the total number of subproblems. Notice that the CPU time is directly proportional to the number of subproblems (or to the number of subproblems plus the number of subproblems solved with the cutting planes (column w/c) for the B & C).

Table 4: Performance comparison between the GSP, FH and the B & B, B & C algorithms (Total number of subproblems in the branching tree and CPU time) on problems from literature and pseudo-randomly generated problems

Problem	Dimension			Total number of subproblems					CPU time (sec.)			
	n	m	k	GSP	FH	B & B	B & C	(w/c)	GSP	FH	B & B	B & C
gsp [14]	1	1	3	97	15	15	-	-	0.9	0.1	0.1	-
twy [27]	2	3	4	*	387	363	363	(24)	*	4.5	4.5	4.8
goh [12]	2	4	6	*	229	171	173	(9)	*	4.6	3.5	3.7
fh [8]	4	6	6	*	4677	1805	1805	(8)	*	272.0	107.5	108.1
rand1	1	1	4	209	47	49	-	-	2.0	0.2	0.3	-
rand2	2	2	8	7231	121	125	125	(0)	219.5	2.4	2.6	2.6
rand3	3	3	12	*	519	531	519	(10)	*	30.6	32.8	32.7
rand4	4	4	16	*	2199	1717	1473	(175)	*	355.9	285.4	279.2
rand5	5	5	20	*	3915	3595	3267	(237)	*	1580.9	1580.0	1538.0
rand6	6	6	24	*	2475	2287	2033	(133)	*	2587.0	2527.6	2422.5
rand7	7	7	28	*	3783	2687	2283	(213)	*	8542.4	6229.1	5858.2
rand8	8	8	32	*	1785	1421	1177	(147)	*	7989.4	6447.6	6090.8
rand9	9	9	36	*	2437	2131	1879	(161)	*	20488.2	19687.7	18523.4
rand10	10	10	40	*	1473	1251	1067	(103)	*	20945.1	19886.2	18710.4

* algorithm stopped after generating 10000 subproblems

w/c: number of subproblems solved with the cutting planes among the total number of subproblems

Table 5 gives the maximum number of active subproblems for the branch-and-bound algorithms. The smallest values were reached by the B & C and B & B which reflects the fact that they require less memory compared to the GSP and the FH.

Finally, Table 6 shows that almost all the computational time is spent to evaluate the lower bounds for the FH, the B & B and the B & C. Also, it is clear that the Alternating SDPs Method (Algorithm 2.7) utilized in the GSP is quite expensive if compared to the Fujioka and Hoshijima's procedure (Section 2.2) utilized in the FH, and in the B & B and the B & C for small instances. See parameter β.

Now, Figures 2 and 3 show the effectiveness of the new branching rule proposed in Section 3.2 - (i). It compares the performance of the Branch-and-Bound Algorithm which incorporates the new branching rule to the same algorithm using the bisection branching rule instead. These algorithms are refereed now as the B & B and the B - R, respectively. Notice further that the B - R is extremely similar to the FH. The FH always utilizes the Fujioka and Hoshijima's heuristic procedure to compute an approximate solution of the BMIEP, while the B - S (and therefore the

Table 5: Performance comparison between the GSP, FH and the B & B, B & C algorithms (Maximum number of active subproblems)

Problem	Dimension			Maximum nr. of active subproblems			
	n	m	k	GSP	FH	B & B	B & C
gsp	1	1	3	5	4	2	-
twy	2	3	4	*	65	56	56
goh	2	4	6	*	26	23	23
fh	4	6	6	*	262	149	151
rand1	1	1	4	18	3	3	-
rand2	2	2	8	850	6	8	8
rand3	3	3	12	*	36	35	33
rand4	4	4	16	*	210	178	123
rand5	5	5	20	*	258	233	190
rand6	6	6	24	*	138	146	119
rand7	7	7	28	*	209	145	119
rand8	8	8	32	*	118	88	67
rand9	9	9	36	*	128	120	99
rand10	10	10	40	*	92	65	53

* algorithm stopped after generating 10000 subproblems

Table 6: Performance comparison between the GSP, FH and the B & B, B & C algorithms (Percentage of the time consumed to evaluate $lb(\cdot)$, $lb_2(\cdot)$ and $ub(\cdot)$)

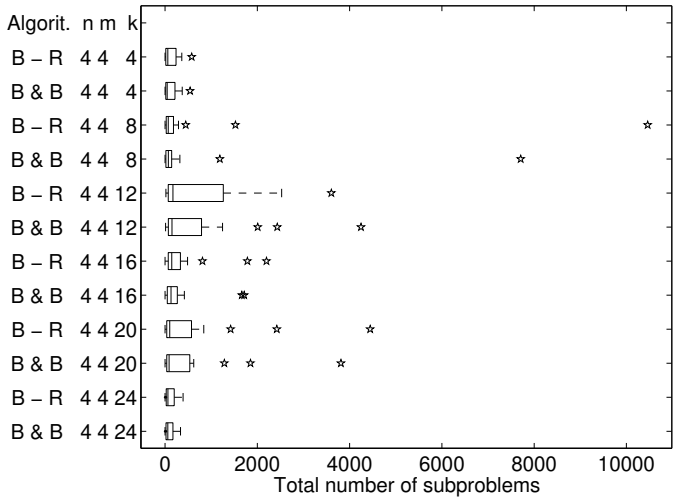
Algorithm	GSP		FH		B & B		B & C		
	$lb(\cdot)$	$ub(\cdot)$	$lb(\cdot)$	$ub(\cdot)$	$lb(\cdot)$	$ub(\cdot)$	$lb(\cdot)$	$lb_2(\cdot)$	$ub(\cdot)$
%	$42 \sim 62$	$38 \sim 58$	$93 \sim 100$	$0 \sim 1$	$94 \sim 100$	$0 \sim 6$	$86 \sim 98$	$0 \sim 12$	$0 \sim 4$

B & B) uses the Alternating SDPs Method, and the heuristic procedure with the parameter β fixed to 50 for this series of comparison. Therefore, the computational results we obtain for the B - S become almost the same as we obtain for the FH. The dimensions of the variables $\boldsymbol{x}$ and $\boldsymbol{y}$ were fixed to 4, and the dimensions of the matrices $\boldsymbol{B}_{ij}$'s was changed from 4×4 to 24×24 for the test problems. 20 randomly generated BMIEPs were solved for each dimension.

Figure 2 shows the total number of subproblems generated by each algorithm for each test problem. The results are summarized by using the box plot representation where each box includes the results of roughly 50% of the problems, the vertical line in the box represents the median and the stars the far out values.

Both Figure 2 and Figure 3 indicate that the new branching rule can decrease the corresponding factor (the total number of subproblems or the CPU time) in 20% for some cases compared to the bisection branching rule. Notice that some small instances of the problem are harder to be solved than the larger ones (*e.g.*, see the maximum for $k = 8$ and $k = 24$). This is not a surprising fact since we are dealing with randomly generated problems, and it is common to obtain wide variations in the computational results when branch-and-bound methods are employed [23].

Figure 2: Performance comparison between the B - R and the B & B algorithms (Total number of subproblems in the branching tree)



Finally, Figures 4 and 5 give comparative results for the Branch-and-Bound and Branch-and-Cut Algorithms proposed in this paper. We fixed $n = m$ and $k = 4n$ since we wanted to solve BMIEPs with $k \geq n, m$. 20 randomly generated problems were solved for $1 \leq n = m \leq 6$ and 7 problems for $7 \leq n = m \leq 10$. The parameter β (Section 2.2) was fixed as given in Table 3. Notice that fixing the parameter β to 2 for $n = m = 10$, we mainly utilized the Alternating SDPs Method (Algorithm 2.7) to evaluate the upper bounds in the B & B and the B & C.

Excepting small instances which $2 \leq n = m \leq 3$, the B & C is slightly superior than the B & B in terms of the total number of subproblems and computation time. As we could expect, the total number of subproblems generated during the whole algorithms is always fewer for the B & C. Reflecting this, the average CPU time decreased $0.5\% \sim 6.6\%$ for $n, m \geq 4$. These facts demonstrate that the cuts, which we introduced in the B & C by heuristic procedures (Section 3.2 - (ii)), are effective to diminish the total number of subproblems.

The largest problem we solved has dimension $n = m = 10$ and $k = 40$, and we needed 12

Figure 3: Performance comparison between the B - R and the B & B algorithms (CPU time)

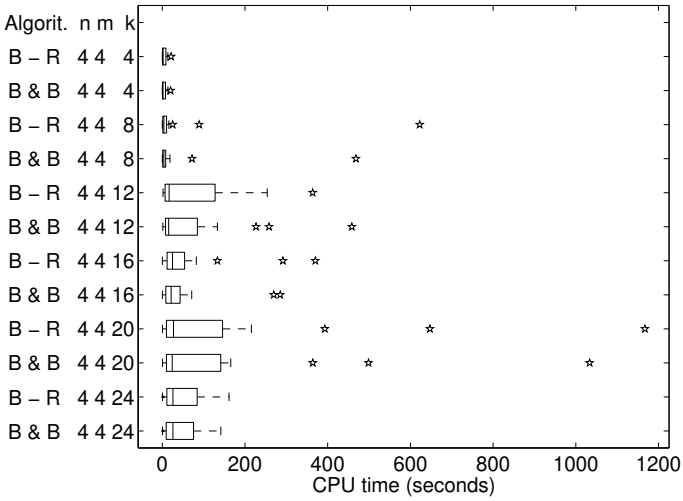


Figure 4: Performance comparison between the B & B and the B & C algorithms (Total number of subproblems in the branching tree)

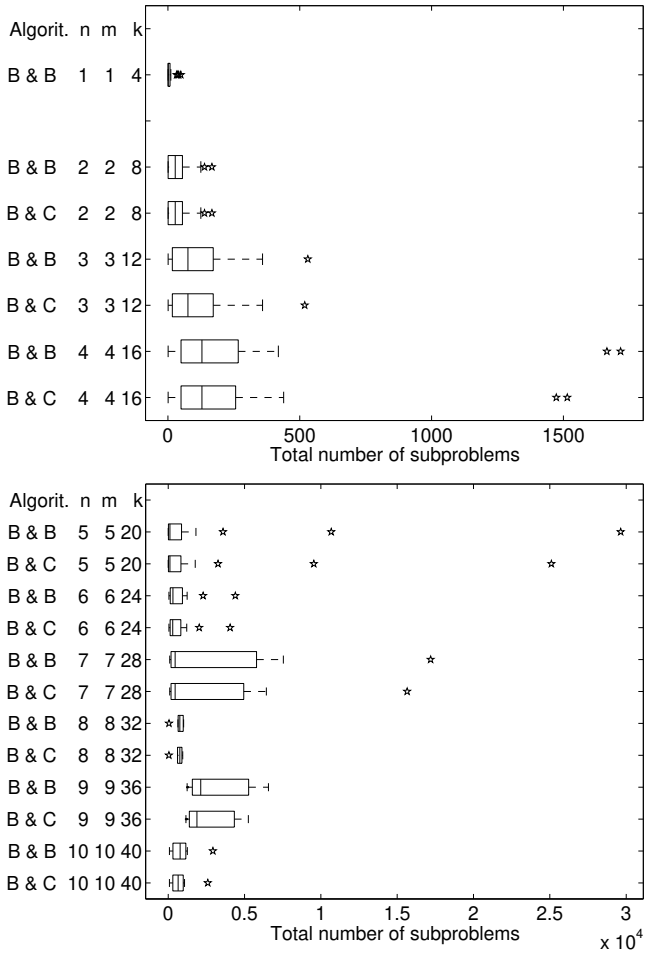
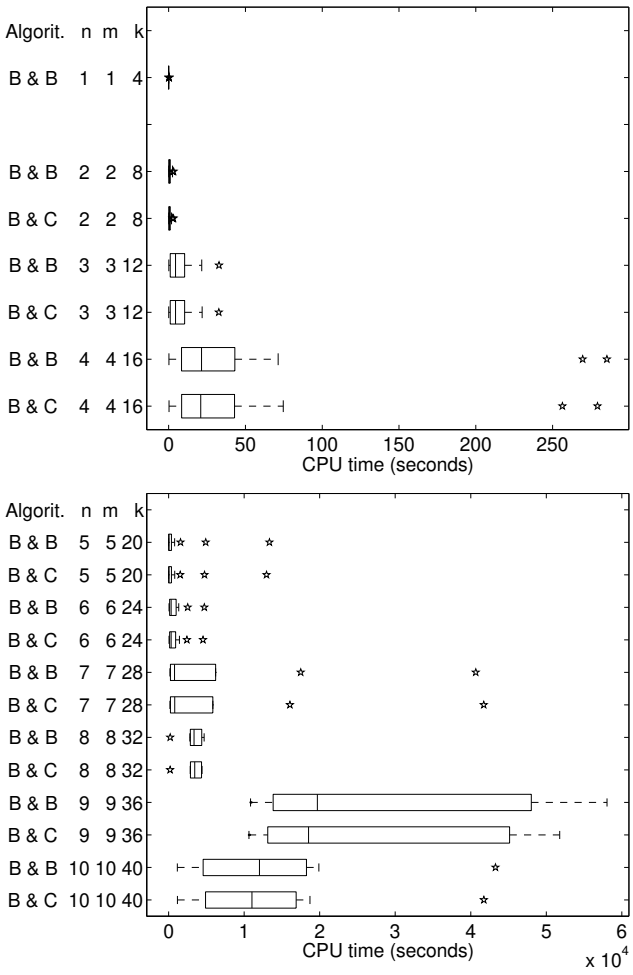


Figure 5: Performance comparison between the B & B and the B & C algorithms (CPU time)



hours (41742 sec.) and approximately 14MB of memory to solve it. Though, the most difficult problem has dimension $n = m = 9$ and $k = 36$ for which we needed 15 hours of computation.

The numerical experiments show that the B & C is the most efficient algorithm to solve the BMIEP among the algorithms considered in this paper. We can order the tested algorithms in increasing order of efficiency as the GSP [14], the FH [8], the B & B and the B & C. The Branch-and-Bound and Branch-and-Cut Algorithms are very robust since they solved all the randomly generated problems. We emphasize that our main result we obtained in this series of numerical experiments is that problems involving BMIs of medium size (variables with dimensions 10 each and matrices with dimensions 40×40) were solved. The following factors can be attribute for this result:

1. The algorithms were written in C++, instead of MATLAB;
2. A very efficient SDP solver, the SDPA 4.20, was incorporated in the subroutines;
3. An efficient branching rule was proposed and implemented in the B & B and the B & C (Section 3.2 - (i));
4. The cuts based on the convex relaxation BMIEP$_{\widetilde{\mathcal{E}}_W}$ (Section 2) were cleverly introduced in the B & C accelerating its convergence, specially for large instances (Section 3.2 - (ii)).

5 Concluding Remarks

The main contribution of this article is to have shown that BMIEPs of medium size (variables $\boldsymbol{x}$ and $\boldsymbol{y}$ with dimensions 10 each and matrices $\boldsymbol{B}_{ij}$'s with dimensions 40×40) can be solved on the currently available computers.

Convex relaxations of the BMIEP were studied in details following the approach of Goh *et al.* [14] and Fujioka and Hoshijima [8]. A theoretically better convex relaxation BMIEP$_{\widetilde{\mathcal{E}}_W}$ was proposed through the connection with the well-known correlation polytope (boolean quadratic polytope) of the graph theory.

The Branch-and-Bound Algorithm and the Branch-and-Cut Algorithm were proposed based on the above results. The numerical experiments over randomly generated problems demonstrated that these algorithms are very robust. They showed also that the proposed Branch-and-Cut Algorithm is the most efficient compared to the tested branch-and-bound based algorithms in terms of the computational time and the amount of memory for high dimension instances.

The convex relaxations and the algorithms exposed here can be naturally extended to optimization problems with a bilinear objective function and BMI constraints. On the contrary of the BMIEPs which are always feasible, these problems can be infeasible. However, the currently available SDP packages [9, 26, 28] can detect infeasibility of the SDPs, and therefore, only minor modifications are necessary in the proposed algorithms to solve these problems.

A different approach to solve problems involving BMIs was recently proposed by Kojima and Tunçel. They announced two conceptual algorithms based on LP and SDP relaxations to solve nonconvex quadratic programs [18] and their implementable variants [19]. In their paper, they showed that their framework is quite general to be applied to *Quadratic Matrix Inequalities* (QMIs), which includes BMIs. Jarre [16] proposed an algorithm based on interior-point method, SQP method and trust regions to solve SDP with nonlinear equalities. Some preliminary computations have shown that this method is also efficient to approximate local optima of problems involving BMIs.

Acknowledgment

The authors want to thank M. Shida of Kanagawa University and A. Deza of Tokyo Institute of Technology for several useful comments, and also K. Fujisawa of Kyoto University for making the SDPA available.

References

[1] F. A. Al-Khayyal and J. E. Falk, "Jointly constrained biconvex programming," *Mathematics of Operational Research* **8** (1983) 273–286.

[2] E. B. Beran, "Methods for optimization-based fixed-order control design," PhD thesis, Department of Mathematics and Department of Automation, Technical University of Denmark, Denmark, September 1997.

[3] E. B. Beran, L. Vandenberghe and S. Boyd, "A global BMI algorithm based on the generalized Benders decomposition," in: *Proceedings of the European Control Conference*, Brussels, Belgium, July 1997.

[4] B. Borchers, "CSDP, a C library for semidefinite programming," Department of Mathematics, New Mexico Institute of Mining and Technology, Socorro, NM, November 1998. Available at http://www.nmt.edu/~borchers/csdp.html.

[5] S. Boyd, L. El Ghaoui, E. Feron and V. Balakrishnan, *Linear matrix inequalities in system and control theory* (SIAM, Philadelphia, 1994).

[6] M. M. Deza and M. Laurent, *Geometry of cuts and metrics* (Springer-Verlag, Berlin, 1997).

[7] C. A. Floudas and V. Visweswaran, "A primal-relaxed dual global optimization approach," *Journal of Optimization Theory and Applications* **78** (1993) 187–225.

[8] H. Fujioka and K. Hoshijima, "Bounds for the BMI eingenvalue problem - a good lower bound and a cheap upper bound," *Transactions of the Society of Instrument and Control Engineers* **33** (1997) 616–621.

[9] K. Fujisawa, M. Kojima and K. Nakata, "SDPA (SemiDefinite Programming Algorithm) - user's manual - version 4.10," Research Report B-308, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan, December 1995, revised May 1998. Available at ftp://ftp.is.titech.ac.jp/pub/OpRes/software/SDPA.

[10] K. Fukuda, "Cdd/Cdd+ reference manual," Institute for Operations Research, ETH-Zentrum, Zurich, Switzerland, December 1997. Available at http://www.ifor.math.ethz.ch/ifor/staff/fukuda/cdd_home/cdd.html.

[11] M. Fukuda, "Branch-and-cut algorithms for bilinear matrix inequality problems," Master thesis, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan, February 1999.

[12] K.-C. Goh, "Robust control synthesis via bilinear matrix inequalities," PhD thesis, University of Southern California, Los Angeles, CA, May 1995.

[13] K.-C. Goh, M. G. Safonov, and J. H. Ly, "Robust synthesis via bilinear matrix inequalities," *International Journal of Robust and Nonlinear Control* **6** (1996) 1079–1095.

[14] K.-C. Goh, M. G. Safonov, and G. P. Papavassilopoulos, "Global optimization for the biaffine matrix inequality problem," *Journal of Global Optimization* **7** (1995) 365–380.

[15] P. Huard, "Resolution of mathematical programming with nonlinear constraints by the method of centres," in: J. Abadie, ed., *Nonlinear Programming* (North-Holland Publishing Company, Amsterdam, 1967).

[16] F. Jarre, "A QQP-minimization method for semidefinite and smooth nonconvex programs," Working Paper, Abteilung Mathematik, Universität Trier, Trier, Germany, August 1998.

[17] M. Kawanishi, T. Sugie and H. Kanki, "BMI global optimization based on branch and bound method taking account of the property of local minima," in: *Proceedings of the Conference on Decision and Control*, San Diego, CA, December 1997.

[18] M. Kojima and L. Tunçel, "Cones of matrices and successive convex relaxation of nonconvex sets," Research Report B-338, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan, March 1998.

[19] M. Kojima and L. Tunçel, "Discretization and localization in successive convex relaxation method for nonconvex quadratic optimization problems," Research Report B-341, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan, July 1998.

[20] S.-M. Liu and G. P. Papavassilopoulos, "Numerical experience with parallel algorithms for solving the BMI problem," in: *13th Triennial World Congress of IFAC*, San Francisco, CA, July 1996.

[21] M. Mesbahi and G. P. Papavassilopoulos, "A cone programming approach to the bilinear matrix inequality problem and its geometry," *Mathematical Programming* **77** (1997) 247–272.

[22] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming* (SIAM, Philadelphia, 1994).

[23] K. G. Ramakrishnan, M. G. C. Resende and P. M. Pardalos, "A branch and bound algorithm for the quadratic assignment problem using a lower bound based on linear programming," in: C. A. Floudas and P. M. Pardalos ed., *State of the Art in Global Optimization* (Kluwer Academic Publishers, Dordrecht, 1996).

[24] M. G. Safonov, K.-C. Goh and J. H. Ly, "Control system synthesis via bilinear matrix inequalities," in: *Proceedings of the American Control Conference*, Baltimore, MD, June 1994.

[25] H. D. Sherali and A. R. Alameddine, "A new reformulation-linearization technique for bilinear programming problems," *Journal of Global Optimization* **2** (1992) 379–410.

[26] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," Department of Quantitative Economics, Maastricht University, Maastricht, The Netherlands, August 1998. Available at http://www.unimaas.nl/~sturm/software/sedumi.html.

[27] S. Takano, T. Watanabe and K. Yasuda, "Branch and bound technique for global solution of BMI," *Transactions of the Society of Instrument and Control Engineers* **33** (1997) 701–708 (in Japanese).

[28] K. C. Toh, M. J. Todd and R. H. Tütüncü, "SDPT3 - a MATLAB software package for semidefinite programming," Department of Mathematics, National University of Singapore, Singapore, November 1998. Available at http://www.math.nus.sg/~mattohkc/index.html.

[29] O. Toker and H. Özbay, "On the NP-hardness of solving bilinear matrix inequalities and simultaneous stabilization with static output feedback," in: *Proceedings of the American Control Conference*, Seattle, WA, June 1995.

[30] H. D. Tuan, S. Hosoe and H. Tuy, "D. C. optimization approach to robust control: feasibility problems," Technical Report 9601, Nagoya University, Aichi, Japan, October 1996.

[31] J. G. VanAntwerp, "Globally optimal robust control for systems with time-varying nonlinear perturbations," Master thesis, University of Illinois at Urbana-Champaign, Urbana, IL, 1997.

[32] L. Vandenberghe and S. Boyd, "Semidefinite Programming," *SIAM Review* **38** (1996) 49–95.

[33] Y. Wakasa, M. Sasaki and T. Tanino, "A primal-relaxed dual global optimization approach for the BMI problem," in: *Proceedings of the 26th Symposium of Control Theory*, Chiba, Japan, May 1997 (in Japanese).

[34] Y. Yajima, M. V. Ramana and P. M. Pardalos, "Cuts and semidefinite relaxations for nonconvex quadratic problems," Research Report 98-1, Department of Industrial Engineering and Management, Tokyo Institute of Technology, Tokyo, Japan, January 1998.

A Appendix

Theorem 2.5 is proved in this section using the concept of correlation polytope which is closely related to the well-known cut polytope [6] in graph theory.

Lemma A.1 *Let*

$$\begin{array}{rcl} T : \mathcal{H} \times \mathbb{R}^{n\times m} & \longrightarrow & [0,1]^n \times [0,1]^m \times \mathbb{R}^{n\times m} \\ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) & \longmapsto & (\boldsymbol{x}', \boldsymbol{y}', \boldsymbol{W}') \end{array}$$

be the affine transformation defined by

$$\begin{array}{rcl} x_i' & = & \dfrac{x_i - \underline{x}_i}{\overline{x}_i - \underline{x}_i}, \qquad y_j' \;=\; \dfrac{y_j - \underline{y}_j}{\overline{y}_j - \underline{y}_j}, \\ w_{ij}' & = & \dfrac{-\underline{y}_j x_i - \underline{x}_i y_j + w_{ij} + \underline{x}_i \underline{y}_j}{(\overline{x}_i - \underline{x}_i)(\overline{y}_j - \underline{y}_j)}, \end{array} \qquad i \in \mathcal{I}_n, \quad j \in \mathcal{J}_m.$$

Then, T is a homeomorphism, and $\boldsymbol{W} = \boldsymbol{x}\,\boldsymbol{y}^T$ if and only if $\boldsymbol{W}' = \boldsymbol{x}'\,\boldsymbol{y}'^T$.

Proof: By a simple arithmetic inspection. ∎

The above lemma shows that we can restrict our discussion of approximating the nonconvex set $\mathcal{F}_W$ (4) to the particular case $\mathcal{F}_W = \mathcal{F}_W^{0,1}(n,m) = \{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n\times m} : (\boldsymbol{x}, \boldsymbol{y}) \in [0,1]^n \times [0,1]^m, \quad \boldsymbol{W} = \boldsymbol{x}\,\boldsymbol{y}^T\}$ hereafter.

In order to prove Theorem 2.5, we need some results from graph theory. Most of the following definitions and results can be found in the book of Deza and Laurent [6].

Let $K_n = (V_n, E_n)$ denote the complete graph with n nodes $V_n = \{1, 2, \cdots, n\}$, and $n(n-1)/2$ edges $E_n = \{ij : \quad i, j \in V_n, \quad i \neq j\}$ (the symbol ij denotes the unordered pair of the integers i, j).

Remark A.2 *This section will follow some notation of [6]. For instance, $\mathbb{R}^{E_n}$ denotes the $|E_n|$-dimensional real space. Also, we implicitly understand one-to-one correspondences between isometric spaces in order to keep the notation compact. For instance, a coordinate of $u \in \mathbb{R}^{E_n}$ ($|E_n| = n(n-1)/2$) will be denoted by u_{ij} ($1 \leq i < j \leq n$).*

Given a subset $S \subseteq V_n$, let denote by $\pi(S) \in \mathbb{R}^{V_n \cup E_n}$ the *correlation vector* defined by:

$$\pi(S)_{ij} = \begin{cases} 1, & \text{if} \quad i, j \in S, \\ 0, & \text{otherwise,} \end{cases} \qquad 1 \leq i \leq j \leq n.$$

Definition A.3 *The polytope in $\mathbb{R}^{V_n \cup E_n}$ which is defined as the convex hull of the correlation vectors $\pi(S)$ for all the subsets S of V_n is called the correlation polytope, and it is denoted by $COR^{\square}(K_n) = COR_n^{\square}$. In other words,*

$$COR_n^{\square} = \{ \sum_{S \subseteq V_n} \lambda_S \pi(S) : \sum_{S \subseteq V_n} \lambda_S = 1, \quad \lambda_S \geq 0, \quad \forall S \subseteq V_n \}.$$

The correlation polytope is also known as the *boolean quadratic polytope*. It is well-known that the correlation polytope is in one-to-one correspondence to the cut polytope via the *covariance mapping* [6]:

$$\begin{array}{rccc} \xi : \mathbb{R}^{E_{n+1}} & \longrightarrow & \mathbb{R}^{V_n \cup E_n} \\ \boldsymbol{d} & \longmapsto & \boldsymbol{p} \end{array}$$

where

$$\begin{cases} p_{ii} &= d_{i,n+1}, & 1 \leq i \leq n, \\ p_{ij} &= \frac{1}{2}(d_{i,n+1} + d_{j,n+1} - d_{ij}), & 1 \leq i < j \leq n. \end{cases} \tag{10}$$

The cut polytope is a widely studied geometric object in graph theory, specially in how to determine all its facets [6]. However, it is known that even the problem to check if a given vector belongs or not in the cut polytope is NP-hard [6]. Since cut and correlation polytopes are mathematically equivalent objects under a linear bijection, we mostly center our discussion in the correlation polytope.

The next defined metric polytope is closely related to the cut polytope and therefore to the correlation polytope as well.

Definition A.4 *The polytope in $\mathbb{R}^{E_n}$ defined by all triangle inequalities is called the metric polytope $MET^{\square}(K_n) = MET_n^{\square}$. That is,*

$$MET_n^{\square} = \left\{ \boldsymbol{u} \in \mathbb{R}^{E_n} : \begin{array}{lll} u_{ij} - u_{ik} - u_{jk} \leq 0, & i, j, k \in V_n, & i \neq j, j \neq k, k \neq i, \\ u_{ij} + u_{ik} + u_{jk} \leq 2, & i, j, k \in V_n, & i \neq j, j \neq k, k \neq i \end{array} \right\}.$$

The notion of the correlation polytope and metric polytope can be extended for any arbitrary graph.

Definition A.5 *Let* $G = (V_n, E)$ *be any graph of* n *nodes with the edges given by* $E \subseteq E_n$. *Then the* $COR^{\square}(G)$ *and the* $MET^{\square}(G)$ *are defined as the projections of the* $COR^{\square}$ *and the* $MET_n^{\square}$ *on the spaces* $\mathbb{R}^{V_n \cup E}$ *and* $\mathbb{R}^{E}$, *respectively.*

The following lemma is a reformulation of the result given by Yajima *et al.* [34] for quadratic problems which we specialized for the bilinear case. It essentially proves that any point in $\mathcal{F}_W^{0,1}(n,m)$ can be expressed as a linear combination of the points of the form $(\boldsymbol{z}, \boldsymbol{t}, \boldsymbol{z}\ \boldsymbol{t}^T)$ such that $(\boldsymbol{z}, \boldsymbol{t}) \in \mathcal{D}_{n,m} = \{0,1\}^n \times \{0,1\}^m$. This lemma is the key result which permits to apply the relaxation of 0-1 quadratic programs [6] to continuous bilinear optimization problems.

Lemma A.6 [34] *Consider* $(\boldsymbol{x}, \boldsymbol{y}) \in [0,1]^n \times [0,1]^m$ *and for each* $(\boldsymbol{z}, \boldsymbol{t}) \in \mathcal{D}_{n,m}$, *let* $\lambda(\boldsymbol{z}, \boldsymbol{t}) \in \mathbb{R}$ *defined by*

$$\lambda(\boldsymbol{z}, \boldsymbol{t}) = \prod_{z_i=0} (1 - x_i) \prod_{z_i=1} x_i \prod_{t_i=0} (1 - y_i) \prod_{t_i=1} y_i.$$

Then,

$$\sum_{(\boldsymbol{z}, \boldsymbol{t}) \in \mathcal{D}_{n,m}} \lambda(\boldsymbol{z}, \boldsymbol{t}) = 1, \qquad \lambda(\boldsymbol{z}, \boldsymbol{t}) \geq 0, \tag{11}$$

and

$$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}\ \boldsymbol{y}^T) = \sum_{(\boldsymbol{z}, \boldsymbol{t}) \in \mathcal{D}_{n,m}} \lambda(\boldsymbol{z}, \boldsymbol{t})(\boldsymbol{z}, \boldsymbol{t}, \boldsymbol{z}\ \boldsymbol{t}^T). \tag{12}$$

Consider now the complete bipartite graph $K_{n,m} = (V_n, V_m, E_{n,m})$ of n and m nodes, $V_n = \{1_n, 2_n, \cdots, n_n\}$, $V_m = \{1_m, 2_m, \cdots, m_m\}$, and edges $E_{n,m} = \{i_n j_m : i_n \in V_n, \quad j_m \in V_m\}$. According to Definitions A.3 and A.5, the correlation polytope for $K_{n,m}$ can be rewritten as:

$$COR^{\square}(K_{n,m}) = \left\{ \sum \lambda_{(\boldsymbol{z}, \boldsymbol{t})} (\boldsymbol{z}, \boldsymbol{t}, \boldsymbol{z}\ \boldsymbol{t}^T) : (\boldsymbol{z}, \boldsymbol{t}) \in \mathcal{D}_{n,m}, \begin{array}{l} \sum \lambda_{(\boldsymbol{z}, \boldsymbol{t})} = 1, \\ \lambda_{(\boldsymbol{z}, \boldsymbol{t})} \geq 0 \end{array} \right\}. \tag{13}$$

As an immediate consequence of Lemma A.6, we have the following proposition which shows that the convex hull of the set $\mathcal{F}_W^{0,1}$ is exactly the correlation polytope for the complete bipartite graph $K_{n,m}$ where n and m denote respectively the dimension of the variables $\boldsymbol{x}$ and $\boldsymbol{y}$ in our original context.

Proposition A.7 $[\mathcal{F}_W^{0,1}(n,m)] = COR^{\square}(K_{n,m})$.

Proof: From Lemma A.6 and (13), we have that $\mathcal{F}_W^{0,1}(n,m) \subseteq COR^{\square}(K_{n,m})$. Since $COR^{\square}(K_{n,m})$ is convex, $[\mathcal{F}_W^{0,1}(n,m)] \subseteq COR^{\square}(K_{n,m})$. On the other hand,

$$(\boldsymbol{z}, \boldsymbol{t}, \boldsymbol{z}\ \boldsymbol{t}^T) \in \mathcal{F}_W^{0,1}(n,m), \quad \forall (\boldsymbol{z}, \boldsymbol{t}) \in \mathcal{D}_{n,m},$$

and thus $COR^{\square}(K_{n,m}) \subseteq [\mathcal{F}_W^{0,1}(n,m)]$. ∎

The above proposition permits us to apply theoretical results of the correlation polytope to analyze the set $[\mathcal{F}_W^{0,1}(n,m)]$. It also shows that $[\mathcal{F}_W^{0,1}(n,m)]$ is always a *polytope* in $\mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n \times m}$.

Theorem A.8 **[6]** *For a graph* G, $COR^{\square}(G) = \xi(MET^{\square}(\nabla G))$ *if and only if* G *has no* K_4*-minor, where* ∇G *denotes the suspension graph of* $G = (V_n, E)$ *obtained by adding a new node, say* $n+1$, *to* G, *and making it adjacent to all other nodes in* V_n.

Observe that when $\min\{n, m\} = 2$, $K_{n,m}$ does not contain any K_4-minor. Therefore, Proposition A.7 and Theorem A.8 imply that if $\min\{n, m\} = 2$,

$$[\mathcal{F}_W^{0,1}(n, m)] = COR^{\square}(K_{n,m}) = \xi(MET^{\square}(\nabla K_{n,m})).$$

The following theorem characterizes the metric polytope of an arbitrary graph G.

Theorem A.9 **[6]** *Let* $G = (V_n, E)$ *be a graph.*

1. $MET^{\square}(G) = \{ \boldsymbol{u} \in \mathbb{R}_+^E : \boldsymbol{u}_{ij} \leq 1$ *for* $ij \in E$, $\boldsymbol{u}(F) - \boldsymbol{u}(C \backslash F) \leq |F| - 1$ *for* C *cycle of* G, $F \subseteq C$, $|F|$ *odd* $\}$.
2. *Let* C *be a cycle in* G *and* $F \subseteq C$ *with* $|F|$ *odd. The inequality* $\boldsymbol{u}(F) - u(C \backslash F) \leq |F| - 1$ *defines a facet of* $MET^{\square}(G))$ *if and only if* C *is a chordless circuit.*
3. *Let* $ij \in E$. *The inequality* $\boldsymbol{u}_{ij} \leq 1$ *defines a facet of* $MET^{\square}(G)$ *if and only if* $ij \in E$ *does not belong to any triangle of* G.

Finally, we apply the results presented so far to demonstrate Theorem 2.5.

Proof of Theorem 2.5: We can consider without loss of generality that $n = 2$, $\mathcal{F}_W = \mathcal{F}_W^{0,1}(2, m)$ (Lemma A.1), and $\widetilde{\mathcal{E}}_W$ given in the form:

$$\widetilde{\mathcal{E}}_W = \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}) \in \mathbb{R}^2 \times \mathbb{R}^m \times \mathbb{R}^{2\times m} : \begin{array}{l} w_{ij} \leq x_i, \quad w_{ij} \leq y_j, \\ w_{ij} \geq 0, \quad w_{ij} \geq x_i + y_j - 1, \\ 0 \leq x_k + y_l + w_{ij} - w_{il} - w_{kj} - w_{kl} \leq 1, \\ i, k \in \mathcal{I}_2, \quad i \neq k, \quad j, l \in \mathcal{J}_m, \quad j \neq l \end{array} \right\}.$$

The only fact we have to show is that $\widetilde{\mathcal{E}}_W = \xi(MET^{\square}(\nabla K_{2,m}))$. First, we describe $MET^{\square}(\nabla K_{2,m})$. In order to simplify our notation, we will utilize the indices i, k to indicate the nodes of V_2 and the indices j, l to indicate the nodes of V_m, instead of using i_2, k_2 and j_m, l_m, respectively. From Theorem A.9, we just need to consider all the chordless cycles in $\nabla K_{2,m}$. Analyzing Fig 6, we observe that there are only two types of such cycles:

$$x_i - d_{ij} - y_j - d_{j\star} - z - d_{\star i} - x_i, \quad i \in \mathcal{I}_2, \quad j \in \mathcal{J}_m, \tag{14}$$

and

$$\begin{array}{c} x_i - d_{ij} - y_j - d_{jk} - x_k - d_{kl} - y_l - d_{li} - x_i, \\ i, k \in \mathcal{I}_2, \quad i \neq k, \quad j, l \in \mathcal{J}_m, \quad j \neq l, \end{array} \tag{15}$$

where x_1, x_2 and $y_1, y_2, \cdots, y_m$ denote the nodes in $K_{2,m}$, z denotes the node created by the suspension, and d_{ij} denotes the edge between x_i and y_j. $\star$ referees to the index of node z.

Then, $MET^{\square}(\nabla K_{2,m})$ can be described by the inequalities:

$$\left.\begin{array}{l} d_{ij} - d_{j\star} - d_{i\star} \leq 0, \\ d_{j\star} - d_{i\star} - d_{ij} \leq 0, \\ d_{i\star} - d_{ij} - d_{j\star} \leq 0, \\ d_{ij} + d_{j\star} + d_{i\star} \leq 2, \end{array}\right\} \quad i \in \mathcal{I}_2, \quad j \in \mathcal{J}_m, \tag{16}$$

which comes from the cycles of length 3, and

$$0 \leq d_{ij} + d_{kj} + d_{kl} - d_{il} \leq 2, \quad i,k \in \mathcal{I}_2, \quad i \neq k, \quad j,l \in \mathcal{J}_m, \quad j \neq l, \tag{17}$$

which comes from the cycles of length 4. Applying now the operator ξ to the above inequalities, we obtain:

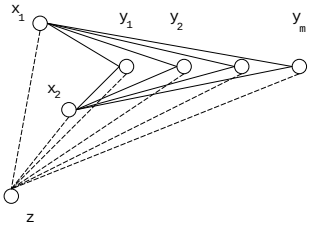
$$\left.\begin{array}{l} -p_{ii} \leq 0, \\ -p_{ii} + p_{ij} \leq 0, \\ -p_{jj} + p_{ij} \leq 0, \\ p_{ii} + p_{jj} - p_{ij} \leq 1, \end{array}\right\} \quad i \in \mathcal{I}_2, \quad j \in \mathcal{J}_m, \tag{18}$$

$$0 \leq p_{kk} + p_{ll} + p_{ij} - p_{il} - p_{kj} - p_{kl} \leq 2, \quad i,k \in \mathcal{I}_2, \quad i \neq k, \quad j,l \in \mathcal{J}_m, \quad j \neq l. \tag{19}$$

Finally, making the association $p_{ii} = x_i$, $p_{jj} = y_j$, and $p_{ij} = w_{ij}$ $(i \in \mathcal{I}_2, j \in \mathcal{J}_m)$, we observe that the above inequalities are precisely the inequalities of the set $\widetilde{\mathcal{E}}_W$, and the result follows. ∎

Essentially, Theorem 2.5 is saying that $\widetilde{\mathcal{E}}_W$ is exactly the metric polytope projected over the complete bipartite graph with n and m nodes.

Figure 6: Suspension graph of $K_{2,m}$



Remark A.10 *When* $\min\{n,m\} = 3$, *the complete bipartite graph* $K_{n,m}$ *has a* K_4 *-minor. Therefore, Theorems A.8 and 2.5 imply that the inclusion* $[\mathcal{F}_W^{0,1}(n,m)] \subset \widetilde{\mathcal{E}}_W$ *is strict in this case.*

The proof of Theorem 2.2 follows the same steps of the proof of Theorem 2.5 omitting the inequalities (15), (17) and (19) which are not present in the case $\min\{m,n\} = 1$.