# D.C. Optimization Approach to Robust Control: Feasibility Problems

H.D. Tuan[*], P. Apkarian[†], S. Hosoe[‡] and H. Tuy[§]

**Abstract.** The feasibility problem for constant scaling in output feedback control is considered. This is an inherently difficult problem [20, 21] since the set of feasible solutions is nonconvex and may be disconnected. Nevertheless, we show that this problem can be reduced to the global maximization of a concave function over a convex set, or alternatively, to the global minimization of a convex program with an additional reverse convex constraint. Thus this feasiblity problem belongs to the realm of d.c. optimization [14, 15, 32, 33], a new field which has recently emerged as an active promising research direction in nonconvex global optimization. By exploiting the specific d.c. structure of the problem, several algorithms are proposed which at every iteration require solving only either convex or linear subproblems.
Analogous algorithms with new characterizations are proposed for the Bilinear Matrix Inequality (BMI) feasibility problem.

## 1 Introduction

Consider the system given by Fig.1, where $G(s)$ is a generalized plant of order $n$ with the state space realization

$$G(s) := \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0 \end{array}\right], \tag{1}$$

and $\Sigma$ is a constant scaling matrix of the form

$$\begin{array}{l} \Sigma = \mathrm{diag}(\sigma_1 I_{r_1}, ..., \sigma_\kappa I_{r_\kappa}),\ \sigma_i > 0,\ i = 1, 2, ..., k,\ q_1 + ... + q_\kappa = q \\ (q \text{ is the dimension of disturbance } w \text{ and control output } z). \end{array} \tag{2}$$

Let $\mathcal{K}$ denote the set of proper controllers $K$ internally stabilizing the closed-loop system $F_l(G, K)$.
The feasibility problem (FP) for constant scaling in output feedback control can be stated as follows:

[*]Department of Control and Information, Toyota Technological Institute, Hisakata 2-12-1, Tenpaku, Nagoya 468-8511, Japan
[†]ONERA-CERT, 2 av. Edouard Belin, 31055 Toulouse, France
[‡]Department of Electronic-Mechanical Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-01, Japan
[§]Institute of Mathematics, P.O. Box 631 Bo ho, Hanoi, Vietnam

*For a fixed value $\gamma$, find a constant scaling matrix $\Sigma$ satisfying (2) and also a controller $K \in \mathcal{K}$ such that*

$$|\Sigma^{-1/2}F_l(G,K)\Sigma^{1/2}|_\infty < \gamma. \tag{3}$$

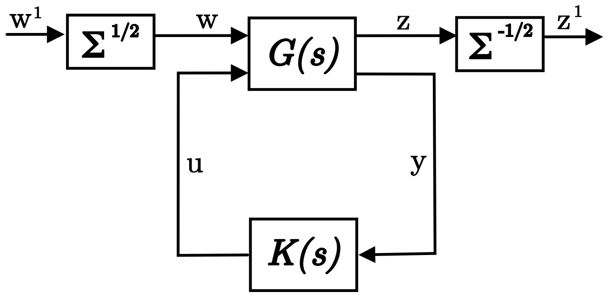

Figure 1: the feedback system

It is well known (see e.g. [21]) that the above FP plays an important role in robust control for systems with structred uncertainties. It is already well known (see e.g. [3]) that FP can be recast as to check the feasiblity of a system of inequality and quality

$$\mathcal{L}(Z,x,y) < 0, \tag{4}$$

$$x_iy_i = 1,\ x_i > 0,\ y_i > 0,\ i = 1,2,...,m = k-1, \tag{5}$$

where (4) is a linear matrix inequality (LMI) in respect to the decision variables $Z, x = (x_1, x_2, ..., x_m), y = (y_1, y_2, ..., y_m)$. In what follows we shall refer to the FP for (4)-(5) as LFP for short.
The difficulty of (4)-(5) is the nonconvex constraint (5) that makes LFP nonconvex.
On the other hand, starting from [25], it has became recognized nowadays that many challenging problems in control theory can be reduced to the so called Bilinear Matrix Inequality feasibility problem (BFP for short) which is to find $x \in [a,b] \subset R^N$ such that

$$F_{00} + \sum_{i=1}^{N} x_iF_{i0} + \sum_{1\le i\le j\le N} x_ix_jF_{ij} < 0 \tag{6}$$

where $F_{i0}, F_{ij}$, for $i,j = 1,\dots,N$ are symmetric matrices. The difficulty of (6) is the presence of nonlinear terms $x_ix_j$ which make BFP nonconvex.

The most existing algorithms for solving LFP and BFP (see e.g. [7, 24]) are heuristic which are able to locate at best a local solution. Only recently techniques of global optimization have begun to be applied for finding a global solution (e.g. [13] for solving a particular case of BFP by the general-purpose branch and bound scheme and [35] for solving LFP by a covering approach).

In this paper, we shall show that LFP and BFP belong to the class of so called *d.c. optimization* problems which have been extensively studied in the last decade (see e.g. [15, 33, 14, 32]). Several available methods and techniques using the concepts of outer approximation [33, 34], polyhedral annexation [32, 33], branch and bound, decomposition [15, Chapter 6] can then be tailored to the special structure of these problems which, even though nonconvex, are characterized by a fairly notable amount of partial convexity.

In the sequel, we shall develop methods for solving LFP and BFP based on the branch and bound approach. In light of recent developments, it seems that this approach is the most suitable for handling nonconvexities in a broad class of optimization problems. However, unlike the branch and bound algorithm in [13], our methods will incorporate a decomposition strategy to exploit the partial convexity present in LFP and BFP.

In fact, it turns out that these problems have a much *lower rank of nonconvexity* than their actual dimension, namely they become convex when a relatively small number of "complicating variables" are held fixed. By *branching* upon these complicating variables, the global search procedure is restricted to operate basically in a low dimensional space, thus mitigating the difficulties of the "curse of dimensionality" inherent to most nonconvex problems. On the other hand, the constraints in these problems are d.c. inequalities, i.e. can be described by $\underline{\text{d}}$ifferences of $\underline{\text{c}}$onvex functions, and consequently can easily be relaxed to linear or convex constraints. This allows an efficient lower bounding technique to be applied for handling the nonconvex feasible set. The resulting branch and bound algorithm should be practicable for small values of $m$ (in (5)) or $N_1$ (in (67)), even if the overall dimension of the problem may be fairly large.

The paper is organized as follows. In section 2 we will present the basic branch and bound (BB) algorithms for 2 classes of low rank nonconvex optimization problems. Based on this scheme, sections 3 develops algorithm for solving LFP, and section 4 algorithms for solving BFP. For completeness we will provide rigorous proof for all main propositions, though some of these might be derived as special cases from more general results of d.c. optimization. Finally, section 5 is devoted to some preliminary computational experience.

The notation of the paper is standard. In particular, $\partial g(x)$ denotes the subdifferential of a convex function $g(x)$, $Q_\perp$ is a matrix satisfying $Q'_\perp Q_\perp > 0$ such that the null space of $Q_\perp$ is equal to the range space of $Q$, $\lambda_{\max}[Q]$ is the maximum eigenvalue of a symmetric matrix $Q$, and $\langle .,. \rangle$ is the scalar product in a finite dimensional linear space, for $x \in R^n$ the notation $x^{-1}$ means $(x_1^{-1}, ..., x_n^{-1})'$ and conv$M$ for a set $M$ means its convex hull.

## 2 Branch and Bound Scheme for Global Optimization and convergence issues

### 2.1 Minization of a nonconvex function over a convex set: CPA

To solve LFP and BFP, one of our approaches is to convert them into checking whether the optimal value of the following optimization problem is not more than zero

$$(P) \qquad \min\{\omega(x,y)|\ (x,y) \in F \subset R^m \times R^n\}$$

where $F$ is a convex set (defined by LMIs), function $\omega(x,y)$ is *nonconvex* but becomes convex (in $y$) when $x$ is held fixed (so $x = (x_1,\ldots,x_m)'$ are "complicating variables"). A branch and bound (BB) method for solving $(P)$ is an iterative procedure in which the $x$-space is iteratively partitioned into smaller sets (*branching*) and the search over each partition set $M$ is carried out through estimating a lower bound $\beta(M)$ of $\omega(x,y)$ over all $(x,y) \in F$ such that $x \in M$ (*bounding*). At each iteration, a feasible solution is known which is the best among all feasible solutions so far obtained. The value of the objective function at this *current best solution* is the *current best value*. Clearly, those partition sets $M$ with $\beta(M)$ higher than the current best value cannot provide any better feasible solution than the current best. They are therefore discarded from further consideration. On the other hand, from the information so far obtained, the partition set with smallest $\beta(M)$ can be considered the most promising one. To concentrate further investigation on this set we subdivide it into more refined subsets. A lower bound is then computed for each of these newly generated partition sets, and the procedure goes to the next iteration. Thus, a BB method for solving $(P)$ involves two basic operations:

1. *Branching*: The space of complicating variables $x = (x_1,\ldots,x_m)$ is partitioned into finitely many polyhedrons of the same kind, e.g. (hyper)rectangles of the form $M = [p,q] = \{x \in R^m|\ p_i \le x_i \le q_i,\ i = 1,\ldots,m\}$. At each iteration, a partition set $M = [p,q]$ is selected and subdivided further into two subrectangles (*children* of $M$) via some hyperplane $x_{i_0} = \alpha$ where $i_0$ and $\alpha \in (p_{i_0}, q_{i_0})$ are chosen according to a specified rule. When $q_{i_0} - p_{i_0} = \max_{i=1,\ldots,m}\{q_i - p_i\}$ and $\alpha = \frac{p_{i_0}+q_{i_0}}{2}$ the subdivision is called the standard *bisection*.

2. *Bounding*: Given a partition set $M$, one has to compute a number $\beta(M)$ such that

$$\beta(M) \le \bar{\omega}(M) := \inf\{\omega(x,y)|\ (x,y) \in F_M\}, \tag{7}$$

where $F_M = \{(x,y) \in F|\ x \in M\}$. A general bounding method is to construct a *convex minorant* $\psi_M(x,y)$ of $\omega(x,y)$ over $M$, i.e. a convex function $\psi_M(x,y)$ satisfying

$$\psi_M(x,y) \le \omega(x,y) \forall y, x \in M. \tag{8}$$

Then, it is obvious that the optimal value $\beta(M)$ of the convex program

$$\min\{\psi_M(x,y)|(x,y) \in F_M\} \tag{9}$$

provides a lower bound for $\omega(x,y)$ over $M$.

Let $M_\kappa$ be the candidate for further partition at iteration $\kappa$ (as mentioned above, $M_\kappa$ is the partition set with smallest lower bound at iteration $\kappa$). To ensure convergence,

the operations of branching and bounding must be *consistent* in the following sense: as $\kappa \to +\infty$, the difference $\min \bar{\omega}(M_\kappa) - \beta(M_\kappa)$ must tend to zero, i.e. the smallest lower bound at iteration $\kappa$ must tend to the sought global minimum $\bar{\omega}(F)$. Of course, a sufficient consistency condition is that the difference between the current best value and the smallest lower bound tends to zero as $\kappa \to \infty$. In that case, by stopping the procedure when this difference has become less than a precribed tolerance $\varepsilon > 0$, the current best solution yields an approximate global optimal solution, called a (global) $\varepsilon$-optimal solution of the problem.
Obviously, the quality of bounding has much influence to the convergence speed of BB algorithm since the tighter $\psi_M(x,y)$ approximates $\omega(x,y)$ the closer $\beta(M_\kappa)$ is to $\bar{\omega}(M_\kappa)$ and the lesser iterations are needed to reach an $\varepsilon-$optimal solution.

Another fact worth mentioning is that the speed of convergence may critically depend on the subdivision strategy. For rectangular subdivisions, although the standard bisection method guarantees convergence under rather general conditions, a drawback of this subdivision strategy is that it does not take account of the problem conditions at the current stage of the algorithm. In many cases the convergence can be significantly sped up by using an adaptive subdivision strategy exploiting the information so far obtained in the course of the iterative process.

In the present paper, the function $\omega(x,y)$ in $(P)$ is shown having the following form

$$\omega(x,y) = f_1(y) + f_2(x) \tag{10}$$

where $f_1$ and $f_2$ have the following special structures:

$(i)$ *$f_1$ is linear on $y$.*

$(ii)$ *$f_2$ is separable concave in $x_i$, i.e.*

$$f_2(x) = \sum_{i=1}^{m} f_{2i}(x_i) \tag{11}$$

*where $f_{2i}$, $i = 1,\dots,m$ are concave functions.*

Note that $(i)$ also means $(P)$ becomes convex when $x$ is held fixed, while $(ii)$ means that for every rectangle $M \subset [a,b]$ there is an affine minorant of $\omega(x,y)$ which agrees with $f$ at the vertices of $M$ ([32, Prop. 5.7]). In fact, it can be easily seen that on $[p_i,q_i]$, the affine function which agrees with $f_{2i}$ at $p_i, q_i$ is

$$\psi_{2iM}(x_i) = f_{2i}(p_i) + \frac{f_{2i}(q_i) - f_{2i}(p_i)}{q_i - p_i}(x_i - p_i) \tag{12}$$

Then function $\tilde{f}_{2i} := f_{2i} - \psi_{2iM}$ is still concave on $[p_i,q_i]$ and thus its mimimum value over $M$ is attained for $x_i \in \{p_i, q_i\}$ which is obviously zero. Therefore $\tilde{f}_{2i}$ is nonnegative on $[p_i,q_i]$ which implies $\psi_{2iM}(x_i) \leq f_{2i}(x_i) \forall x_i \in [p_i,q_i]$, i.e. $\psi_{2iM}$ is an affine minorant of $f_{2i}$ on $[p_i,q_i]$. It can be shown moreover [32] that $\psi_{2iM}$ is the convex envelope of $f_{2i}$ on $[p_i,q_i]$, i.e. $\psi_{2iM}$ is the tightest convex minorant of $f_{2i}$ on $[p_i,q_i]$.

Thus, the tight convex minorant satisfying (8), which agrees with $\omega(x,y)$ at the vertices of $M=[p,q]$ is

$$\psi_M(x,y)=f_1(y)+\sum_{i=1}^{m}[f_{2i}(p_i)+\frac{f_{2i}(q_i)-f_{2i}(p_i)}{q_i-p_i}(x_i-p_i)] \tag{13}$$

Therefore, a lower bound $\beta(M)$ in (7) in a BB algorithm is provided by (9) with $\psi_M(x,y)$ defined by (13).
Let $(x(M),y(M))$ be the optimal solution of (13). Then is is obvious that $\omega(x(M),y(M))$ provides an upper bound for $(P)$ and thus can be used for updating the current best solution. Moreover, if $\psi_M(x(M),y(M))=\omega(x(M),y(M))$ (this happens for instance when $x_i(M)\in\{p_i,q_i\}$) then $\beta(M)$ is the *exact* minimum of $(P)$ over $M$ and thus will provide the optimal value of $(P)$ in our BB process. This implies the following subdivision rule, which makes such situation as fast as possible.

**Adaptive subdivision rule.** *For $M=[p,q]$ select*

$$i_M \quad\in\quad \operatorname{argmax}\{f_{2i}(x_i(M))-\psi_{2iM}(x_i(M))|\ i=1,2,...m\} \tag{14}$$

*Divide $M$ into two subrectangles by the line $x_{i_M}=x_{i_M}(M)$.*

An important property ensured by this subdivision process is that

**Lemma 1** *Let $M_\nu=[p^\nu,q^\nu],\nu=1,2,\ldots,$ be any infinite nested sequence such that every $M_{\nu+1}$ is a child of $M_\nu$ in the above subdivision process. Then there are $i_0\in\{1,2,...,m\}$ and a subsequence $\Delta\subset\{1,2,..\}$ such that $i_{M_\nu}=i_0\ \forall\nu\in\Delta$ and as $\nu\to\infty,\nu\in\Delta$,*

$$\psi_{M_\nu}(x_i(M_\nu),y_i(M_\nu))\to\omega(x_i(M_\nu),y_i(M_\nu)),\quad i=1,2,...m. \tag{15}$$

*Proof* First observe that, since $p^\nu\le p^{\nu+1}\le q^{\nu+1}\le q^\nu$, the limits $\bar p=\lim p^\nu,\bar q=\lim q^\nu$ exist. Next, since $i_{M_\nu}\in\{1,2,...,m\}$, there exist $i_0\in\{1,2,...,m\}$ and a subsequence $\Delta\subset\{1,2,\ldots\}$ such that $i_{M_\nu}=i_0\ \forall\nu\in\Delta$. Without loss of generality we may assume $i_0=1$. By (14) we thus have

$$0\le f_{2i}(x_i^\nu)-\psi_{2iM}(x_i^\nu)\le f_{21}(x_1^\nu)-\psi_{21M}(x_1^\nu),\ i=2,...,m, \tag{16}$$

where $x^\nu,y^\nu$ stand for $x(M_\nu),y(M_\nu)$. We can also assume that $x^\nu\to\bar x,y^\nu\to\bar y$. Now the subdivision rule ensures that $x_1^\nu\in\{p_1^{\nu+1},q_1^{\nu+1}\}$, so letting $\nu\to+\infty,\nu\in\Delta$, yields $\bar x_1\in\{\bar p_1,\bar q_1\}$, which gives $f_{21}(\bar x_1)=\psi_{21M}(\bar x_1)$. This in turn implies, by (16), $f_{2i}(\bar x_i)-\psi_{2iM}(\bar x_i)=0,\ i=2,...,m.$
□

The last thing we should not forget is that we are interested only in checking whether zero is the optimal solution of $(P)$. Thus, at iteration $\kappa$, all $M$ with $\beta(M)>0$ can be deleted from a futher consideration. Suming up, we are now in a position to state our algorithm.

**Concave programming algorithm (CPA).**
*Initialization.* Start with $M_0=[a,b]$. Set $\mathcal{S}_1=\mathcal{N}_1=\{M_0\}$. Set $\kappa=1$.

*Step 1.* For each $M \in \mathcal{N}_\kappa$ solve (13) to obtain $\beta(M)$ and an optimal solution $(x(M), y(M))$. Define $(x^\kappa, y^\kappa)$ to be the best among all the feasible $(x(M), y(M)), M \in \mathcal{N}_\kappa$. If $\omega(x(M), y(M)) \leq 0$ terminate: the optimal value of $(P)$ is not more than 0.

*Step 2.* In $\mathcal{S}_\kappa$ delete all $M$ such that $\beta(M) > 0$. Let $\mathcal{R}_\kappa$ be the set of remaining rectangles. If $\mathcal{R}_\kappa = \emptyset$, terminate: the optimal value of $(P)$ is more than 0.

*Step 3.* Choose $M_\kappa \in \operatorname{argmin}\{\beta(M) | \; M \in \mathcal{R}_\kappa\}$ and divide it into two smaller rectangles $M_{\kappa,1}, M_{\kappa,2}$ according to the above adaptive subdivision rule 1. Let $\mathcal{N}_{\kappa+1} = \{M_{\kappa,1}, M_{\kappa,2}\}$, $\mathcal{S}_{\kappa+1} = (\mathcal{R}_\kappa \setminus M_\kappa) \cup \mathcal{N}_{\kappa+1}$.

Set $\kappa \leftarrow \kappa + 1$ and go back to Step 1.

**Proposition 1** *Either CPA terminates after finitely many iterations, producing the evidence that the optimal value of $(P)$ is more than 0 or not. Or if it is infinite then the optimal value of problem $(P)$ is zero.*

*Proof* Suppose the algorithm is infinite. One of the two childrens of $M_0$ must have infinitely many descendants, hence must be split at some iteration $\kappa_1$, i.e. must be $M_{\kappa_1}$ for some $\kappa_1$ (following our notation, $M_\kappa$ is the candidate for further partition at iteration $\kappa$). Analogously, one of the two childrens of $M_{\kappa_1}$ must be $M_{\kappa_2}$ at some iteration $k_2 > \kappa_1$, and so on. Proceeding that way we see that there exists a nested sequence of rectangles $M_\nu := M_{\kappa_\nu}$, $\nu = 1, 2, \ldots.$ as in Lemma 1. Therefore, we may also assume that $\{M_\nu\} \to \bar{M} = [\bar{p}, \bar{q}]$, $(x^\nu, y^\nu) \to (\bar{x}, \bar{y})$ with $\psi_{\bar{M}}(\bar{x}, \bar{y}) = \omega(\bar{x}, \bar{y})$. Then $(\bar{x}, \bar{y})$ satisfies

$$\omega(\bar{x}, \bar{y})) \geq \min\{\omega(x, y) | \; (x, y) \in F\}. \tag{17}$$

But by the choice of $M_\kappa$ in Step 3, $\beta(M_{\kappa_\nu}) \leq \beta(M) \; \forall M \in \mathcal{R}_{\kappa_\nu}$, hence $\psi_{M_{\kappa_\nu}}(x^\nu, y^\nu) = \beta(M_{\kappa_\nu}) \leq \min\{\omega(x, y) | \quad (x, y) \in F\}$, and by letting $\nu \to \infty : \omega(\bar{x}, \bar{y}) \leq \min\{\omega(x, y) | \quad (x, y) \in F\}$. This, together with (17), implies

$$\omega(\bar{x}, \bar{y}) = \min\{\omega(x, y) | \quad (x, y) \in F\}.$$

Thus, $(\bar{x}, \bar{y})$ is an optimal solution of problem $(P)$. Now, again with

$$\psi_{M_{\kappa_\nu}}(x^\nu, y^\nu) = \beta(M_{\kappa_\nu}) \; \forall \nu \tag{18}$$

and $\beta(M_{\kappa_\nu}) \leq 0 \; \forall \nu$ it follows, by letting $\nu \to +\infty$, that

$$\omega(\bar{x}, \bar{y}) = \psi_{\bar{M}}(\bar{x}, \bar{y}) \leq 0. \tag{19}$$

On the other hand, for every $\nu$, the termination criterion in Step 1 implies

$$\omega(x^\nu, y^\nu) > 0. \tag{20}$$

As $\nu \to \infty$, since $(x^\nu, y^\nu) \to (\bar{x}, \bar{y})$, (20) yields

$$\omega(\bar{x}, \bar{y}) \geq 0, \tag{21}$$

and hence, in view of (19), $\omega(\bar{x}, \bar{y}) = 0$. □

Fortunately, in practice, we only need an approximate solution, within a given tolerance $\varepsilon > 0$. Therefore, in Step 3 we can delete all $M$ such that $\beta(M) \geq -\varepsilon$. With this deletion criterion, the algorithm will terminate after finitely many steps, and when $\mathcal{R}_\kappa = \emptyset$, we can conclude that the optimal value of $(P)$ is more than $-\varepsilon$.

Indeed, if the algorithm were infinite, we would have from (18), where $\beta(M_{\kappa_\nu}) < -\varepsilon$ : $\omega(\bar{x}, \bar{y}) \leq -\varepsilon$, contradicting (21).

## 2.2 Minimization of a convex function over a nonconvex set: DCPA

Alternatively, we also show that in many case, the problems of our interest can be reduced to checking whether the optimal value of $(P)$ is negative when $\omega(x,y)$ is *linear* function but the constraint $(x,y) \in F$ is *nonconvex* which becomes convex (in $y$) when $x$ is held fixed (so again only $x = (x_1, \ldots, x_m)'$ are "complicating variables"). For instance, this happens when $F$ has the representation

$$F = C \cap D \tag{22}$$

where $C$ is a convex set defined by LMIs and $D$ is a nonconvex set defined by d.c. constraint

$$D = \{(x,y) : \; f_1(y) + f_2(x) \le 0\} \tag{23}$$

with convex function $f_1$ and concave function $f_2$.
For developing a BB algorithm to solve $(P)$ is this case, with given a partition set $M$, one has to compute a number $\beta(M)$ satisfying (7). A general bounding method is to construct a convex set

$$H_M \supset F_M := \{(x,y) \in F | \; x \in M\}. \tag{24}$$

Then the optimal value $\beta(M)$ of the convex program:

$$\min\{\omega(x,y) | \; (x,y) \in H_M\}. \tag{25}$$

yields a lower bound for $\omega(x,y)$ over $F_M$, i.e. such $\beta(M)$ satisfies (7).

Again, it is clear that the tighter $H_M$ approximates $F_M$, the closer $\beta(M)$ is to $\omega(F_M)$, and the lesser iterations are needed to reach an optimal solution within a given tolerance, but, on the other hand, the larger will be the computational effort necessary for constructing $H_M$ and estimating the lower bounds. Therefore, for the efficiency of the procedure, a trade-off must be resolved between these two conflicting requirements. For instance, the tightest $H_M$ satisfying (24) is of course $\mathrm{conv} F_M$, which is however hardly described. Instead, one often takes

$$H_M := C \cap \mathrm{conv}\{(x,y) | \; (x,y) \in D, \; x \in M\} \supset \mathrm{conv} F_M \tag{26}$$

which is much easier to described, for instance in problems considered below.
Note however that we can not directly take the optimal solution $(x(M), y(M))$ of the relaxed problem (25) for updating the current best solution as done in the CPA since $(x(M), y(M))$ in general is not feasible to $F$. Instead, the upper bound of problem $(P)$ in $M$ can be provided by the following optimization programm

$$\min\{\omega(x(M), y) : (x(M), y) \in F_M\} \tag{27}$$

which become convex since $x$ is held fixed equal $x(M)$.

Clearly, branching for this nonconvex problem can be performed analogously to that for concave program.

When $F$ has the form (22)-(23) with $f_2(x)$ having the separated concave structure like (11), for every rectangle $M = [p, q]$, a set $H_M$ satisfying (24) can be taken as

$$H_M = C \cap \{(x, y) : \ \psi_M(x, y) \leq 0, \ x \in M\} \tag{28}$$

with $\psi_M(x, y)$ defined by (13). Moreover, it can be shown that $\{(x, y) : \ \psi_M(x, y) \leq 0, \ x \in M\} = \mathrm{conv}\{(x, y) : \ f_1(y) - f_2(x) \leq 0, \ x \in M\}$, i.e. $H_M$ defined by (28) satisfies (26).
If $(x(M), y(M)) \in F_M$ then $\beta(M)$ become the *exact* minimum of $(P)$ over $M$ and thus is the optimal value of $(P)$ in our BB process. Therefore again the adaptive subdivision rule is fit to speed up this situation.

Suming up, for problem $(P)$ where $\omega$ linear, $F$ has form (22), (23) with $f_2$ having structure (11), we can produce the d.c. optimization branch and bound algorithm (DCPA for short) which is different of CPA only by step 1: it uses (25),(28) for lower bound computing instead (13) of CPA and uses (27) for updating the best current solution. Analogously, we can prove that

**Proposition 2** *Either DCPA terminates after finitely many iterations, producing the evidence that the optimal value of $(P)$ is more than 0 or not. Or if it is infinite then the optimal value of problem $(P)$ is zero.*

## 3 The LFP Problem

First, we recall the following already known result (see e.g. [3]).

**Lemma 2** *There is a controller $K(s)$ satisfying (3) if and only if the following inequality system is feasible on symmetric matrices $Z_1, Z_2$ of dimension $n \times n$ and $\Sigma$ of the structure (2).*

$$\begin{array}{rcl}
\begin{bmatrix} B_2 \\ D_{12} \\ 0 \end{bmatrix}_{\perp} \begin{bmatrix} AZ_1 + Z_1A' & Z_1C_1' & B_1 \\ C_1Z_1 & -\gamma\Sigma & D_{11} \\ B_1' & D_{11}' & -\gamma\Sigma^{-1} \end{bmatrix} \begin{bmatrix} B_2 \\ D_{12} \\ 0 \end{bmatrix}_{\perp}' & < & 0 \\
\begin{bmatrix} C_2' \\ D_{21}' \\ 0 \end{bmatrix}_{\perp} \begin{bmatrix} Z_2A + A'Z_2 & Z_2B_1 & C_1' \\ B_1'Z_2 & -\gamma\Sigma^{-1} & D_{11}' \\ C_1 & D_{11} & -\gamma\Sigma \end{bmatrix} \begin{bmatrix} C_2' \\ D_{21}' \\ 0 \end{bmatrix}_{\perp}' & < & 0 \\
\begin{bmatrix} Z_1 & I_n \\ I_n & Z_2 \end{bmatrix} & \geq & 0.
\end{array} \tag{29}$$

□

Note that (29) is LMIs in $(Z_1, Z_2, \Sigma, \Sigma^{-1})$ and we always can set $\sigma_k = 1$. Then by setting $Z = (Z_1, Z_2)$, $x = (\sigma_1, \sigma_2, ..., \sigma_{k-1})$, $y = (1/\sigma_1, 1/\sigma_2, ..., 1/\sigma_{k-1})$ we see that (29) can be rewritten as (4)-(5).
In the sequel without loss of generality we shall restrict $x$ to the domain

$$a_i \leq x_i \leq b_i, \ a_i > 0, b_i > 0, \ i = 1, 2, ..., m, \tag{30}$$

and also $Z$ belongs to some closed bounded set of symmetric matrices. Such an assumption is automatically satisfied when the problem is solved on computer.

## 3.1 CPA and DCPA for LFP

The reduction of $m$ nonconvex constraints in (5) to just 1 nonconvex one is shown in the following lemma.

**Lemma 3** *(5) is equivalent to the following*

$$\begin{bmatrix} x_i & 1 \\ 1 & y_i \end{bmatrix} \geq 0, \ i = 1, 2, ..., m, \tag{31}$$

$$\sum_{i=1}^{m} y_i - \sum_{i=1}^{m} \frac{1}{x_i} \leq 0 \tag{32}$$

*where (31) is LMIs while (32) in nonconvex.*

*Proof* Since the implication (5) $\Rightarrow$ $\{(31), (32)\}$ is obvious, let us prove the inverse implication. Rewrite (32) as

$$\sum_{i=1}^{m} (y_i - \frac{1}{x_i}) \leq 0 \tag{33}$$

But by (31),

$$x_i y_i \geq 1, \ i = 1, 2, ..., m,$$

so (33) holds true if and only if $y_i = \frac{1}{x_i}$, $i = 1, 2, ..., m$, i.e. (5). □

By the above lemma 3, the feasibility of LFP (4)-(5) is equivalent to the feasibility of the system (4), (31), (32) which will be used from now on. Thus, the following result can be easily established.

**Lemma 4** *The system (4), (31), (32) is feasible if and only if either*
*(i) Zero is the optimal value of the problem*

$$\min \sum_{i=1}^{m} y_i - \sum_{i=1}^{m} \frac{1}{x_i} : \ (4), (31), \tag{34}$$

*or*
*(ii) The optimal value of the following optimization problem is negative*

$$\min \ t : \ (31), (32), \tag{35}$$

$$\mathcal{L}(Z, x, y) \leq tI, \tag{36}$$

*where $I$ is the identity matrix with the dimension conformed with that of $\mathcal{L}(Z, x, y)$.*

Since each function $-1/x_i$ in (34) is concave on $(0, +\infty]$, function

$$\omega(x, y) := \sum_{i=1}^{m} y_i - \sum_{i=1}^{m} \frac{1}{x_i} \tag{37}$$

belongs to the class (10)-(11). Therefore, defining

$$F := \{(x, y) | \ \exists Z := (Z_1, Z_2) \text{ s.t. } (4), (31)\}, \tag{38}$$

problem (34) is the concave programm $(P)$ considered in Subsection 2.1 and thus can be solved by CPA, with $\beta(M)$ is computed according (13), (12), (13) by

$$\beta(M) := \min \sum_{i=1}^{m} y_i + \sum_{i=1}^{m} [-\frac{1}{p_i} - \frac{1}{q_i} + \frac{x_i}{p_i q_i}] : \ (4), (31), \ x \in M, \tag{39}$$

which is just a LMI optimization program.

Analogously, (36) is a particular case of problem $(P)$ considered in Subsection 2.2 with $F$ having the structure (22), (23), (11). Thus DCPA is readily applied for solving (34). In this case the convex program (25) for computing the lower bound $\beta(M)$ is looked concretely as the following LMI program

$$\beta(M) := \min\{t : (31), \mathcal{L}(Z, x, y) \le tI, \sum_{i=1}^{m} y_i + \sum_{i=1}^{m} [-\frac{1}{p_i} - \frac{1}{q_i} + \frac{x_i}{p_i q_i}] \le 0\}. \tag{40}$$

Let $(Z(M), x(M), y(M))$ be the optimal solution of (40). Then obviously we can take $\lambda_{\max}[\mathcal{L}(Z(M), \tilde{x}, \tilde{x}^{-1})]$ with $\tilde{x} \in \{x(M), y(M)^{-1}\}$ for updating the upper bound instead of solving an additional optimization problem like (27).

**Remark 1.** For more general cases involving also full block scalings of LFP we refer the reader to [4]. Note that in such cases the separated concave structure is no longer present and the problem become high rank nonconvex ones, which require different BB methods. Although the separated concave structure also disappears in the optimal scaling problem (OSP) (finding the smallest $\gamma$ such that LFP is solvable by some controller $K(s) \in \mathcal{K}$), the solving efficient BB algorithms still are possible [30], which require cputime only about two times than that needed for solving LFP. The concave struture has been also exploited in [5, 29] for solving so called parameterized LMIs arising in many robust control problems.

**Remark 2.** Let as before $(a, b)$ satisfy (30). Denote

$$\mathcal{E}_n(\gamma) = \{(x, y) \in R^{2m} : \ x_i y_i \le \gamma^2, \ i = 1, 2, ..., m; \ x \in [a, b]\}.$$

A covering type algorithm for solving LFP has been proposed in [35] where LFP is transformed to checking the feasibility of the system of LMI (4), with the nonconvex constraint

$$(x, y) \in \mathcal{E}_n(1). \tag{41}$$

The covering algorithm [35] can not directly solve the feasibility of system (4), (41). Instead, for every $\epsilon > 0$, it constructs $N$ convex sets $\mathcal{C}(\theta_i, 1)$ satisfying

$$\mathcal{E}_n(1 - \epsilon) \subset \bigcup_{i=1}^{N} \mathcal{C}(\theta_i, 1) \subset \mathcal{E}_n(1) \tag{42}$$

where

$$N := \prod_{i=1}^{m} \frac{\ln(b_i/a_i)}{\ln(1/\mathcal{K})}, \mathcal{K} = \frac{1 - \sqrt{1 - (1 - \epsilon)^2}}{1 + \sqrt{1 - (1 - \epsilon)^2}}. \tag{43}$$

Thus the feasibility of (4) with

$$(x, y) \in \bigcup_{i=1}^{N} \mathcal{C}(\theta_i, 1) \tag{44}$$

is checked by solving $N$ convex (LMI) feasibility problem of (4) with $(x, y) \in \mathcal{C}(\theta_i, 1)$. For example, with $a_i = 0.001, b_i = 1000$ (as in the numerical examples in Section 5 below) and $\eta = 0.01$ then by (43) one can see that $N = 49$ so one needs to solve $49^m$ (i.e. 49, 2401, 117649,..., for m=1,2,3,..., respectively) convex (LMI) problems for checking the feasibility of (4), (44). Thus, such covering search process can not be practical for $m > 1$. Moreover, by (42), if system (4), (44) is feasible then system (4), (41) is so but *not* vice versa. Hence, even after solving $N$ LMIs feasibility problems one can not know whether (4), (41) is feasible or not. This also means that LFP using the covering technique of [35] may be even more difficult than OSP (the covering algorithm of [35] for OSP with tolerance $\epsilon$ requires solving $N$ LMI programs) that is of course unnatural. Note that the inefficiency of the covering method is quite well known in global optimization, so it is not surprising that the covering method of [35] for both LFP and OSP is not practical for problems with $m > 1$. The attempt in [35, Section 6] to improve the covering process not only does not work for the feasibility problem (4), (41) but seems to be not correct as it illegitimately leaves unexplored a significant portion of the feasible set (see [31] for more details).

## 3.2 Alternative linear programming based algorithm

Although there are a lot of great softwares for solving LMIs, most of them presently still require a high computational cost for LMIs of a large size or a system involving many LMI constraints. Thus DCPA computes lower bounds by solving LMI programs (40), which may be expensive at the first stage when very tight lower bounds are not worthwhile. We next describe an alternative Algorithm which computes lower bounds by using much less expensive linear programs. Since these lower bounds are less accurate than those provided by convex programs, more iterations will be required when approaching the optimum. Therefore, a reasonable compromise seems to be a hybrid algorithm which proceeds as this algorithm in a first stage, then switches to DCPA in the second stage.

Rewrite problem (4)-(5) in the form

$$\min\ t\ :\ f(Z, x, y) \leq t,\ Z \in \mathcal{Z},\ a \leq x \leq b,\ x_i y_i = 1, i = 1, 2, ..., m, \tag{45}$$

where $f(Z, x, y) = \lambda_{\max}[\mathcal{L}(Z, x, y)]$ and $\mathcal{Z}$ is a bounded set of symmetric matrices with the size conformed with that of $Z$.
Note that a subgradient of the convex function $f(Z, x, y)$ at any point $(Z, x, y)$ can be easily computed (see e.g. [22] and also the Appendix).

### 3.2.1 Bounding

At iteration $k$, we already have in hand a convex piecewise affine minorant $L_\kappa(Z, x, y)$ of $f(Z, x, y)$, of the form

$$L_\kappa(Z, x, y) = \max\{l_j(Z, x, y) |\ j \in J_\kappa\},$$

where $l_j(Z, x, y), j \in J_\kappa$, are affine functions resulting from the previous iterations. For each newly generated rectangle $M = [p, q]$ we approximate the arc

$$F_{M,i} = \{(x_i, y_i) :\ x_i y_i = 1,\ x_i \in M\}$$

by the polytope $H_{M,i}$ formed by the chord of this arc with the tangents to the arc at the two endpoints of the arc, and the tangent to the arc parallel to the chord (see Fig.2). It is easily verified that $H_{M,i}$ is determined by the inequalities

$$\left| \begin{array}{l} p_i q_i y_i + x_i \le p_i + q_i, \\ p_i^2 y_i + x_i \ge 2p_i, \;\; q_i^2 y_i + x_i \ge 2q_i, \\ p_i q_i y_i + x_i \ge 2\sqrt{p_i q_i}. \end{array} \right. \tag{46}$$

A lower bound $\beta(M)$ for

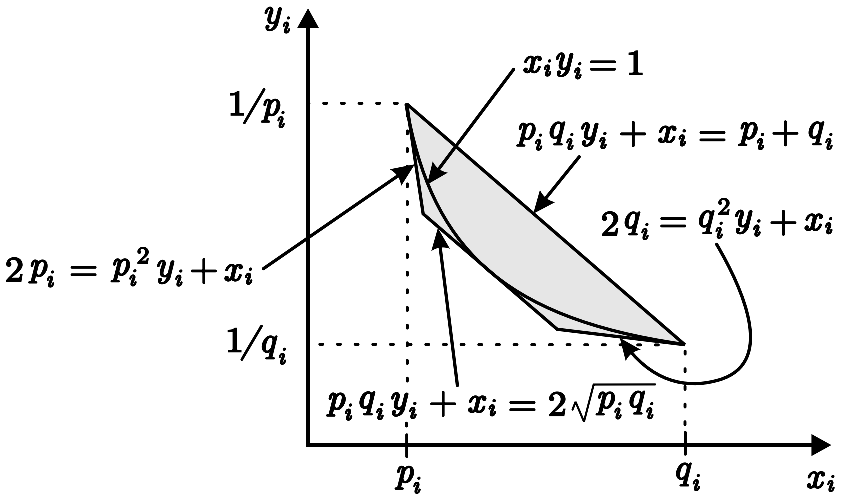

Figure 2: the set $H_{M,i}$

$$\min\{t|\; f(Z,x,y) \le t,\; Z \in \mathcal{Z},\; x_i y_i = 1, x \in M, i = 1,2,...,m\}, \tag{47}$$

is then provided by the optimal value of the linear program

$$\begin{array}{ll} LP(M) \qquad & \min\ t\ : \\ & l_j(Z,x,y) \le t,\; j \in J_\kappa \\ & p_i q_i y_i + x_i \le p_i + q_i\ (i = 1,2,...,m) \\ & p_i^2 y_i + x_i \ge 2p_i,\; q_i^2 y_i + x_i \ge 2q_i\ (i = 1,2,...,m) \\ & p_i q_i y_i + x_i \ge 2\sqrt{p_i q_i}\ (i = 1,2,...,m) \\ & Z \in \mathcal{Z}. \end{array} \tag{48}$$

**Updating** $L_\kappa(Z,x,y)$

At iteration $\kappa$ let $M_\kappa$ be the rectangle with smallest $\beta(M)$, i.e. the candidate for further splitting. Let $(Z(M_\kappa), x(M_\kappa), y(M_\kappa))$ be an optimal solution of $LP(M_\kappa)$. If $f(Z(M_\kappa), x(M_\kappa), y(M_\kappa)) = \beta(M_\kappa)$, then set $L_{\kappa+1}(x,y) = L_\kappa(x,y)$, $J_{\kappa+1} = J_\kappa$. Otherwise, $f(Z(M_\kappa),$

$x(M_\kappa), y(M_\kappa)) > \beta(M_\kappa)$, then $(\beta(M_\kappa), Z(M_\kappa), x(M_\kappa), y(M_\kappa))$ does not belong to the convex set $\{(t, Z, x, y) |\ f(Z, x, y) \leq t\}$ and we can separate it from the latter by the cutting plane

$$l_\kappa(Z, x, y) \leq t \tag{49}$$

where

$$\begin{aligned} l_\kappa(Z, x, y) &= \langle \pi_\kappa, (Z - Z(M_\kappa), x - x(M_\kappa), y - y(M_\kappa))\rangle \\ &\quad + f(Z(M_\kappa), x(M_\kappa), y(M_\kappa)) \end{aligned} \tag{50}$$

with $\pi_\kappa \in \partial f(Z(M_\kappa), x(M_\kappa), y(M_\kappa))$. Indeed, it is obvious that $(\beta(M_\kappa), Z(M_\kappa), x(M_\kappa), y(M_\kappa))$ does not satisfy (49) and if $f(Z, x, y) \leq t$ then by the definition of subdifferential one has

$$l_\kappa(Z, x, y) \leq f(Z, x, y) \leq t,$$

i.e. $(t, Z, x, y)$ satisfies (49).
So, when $f(Z(M_\kappa), x(M_\kappa), y(M_\kappa)) > \beta(M_\kappa)$, we define

$$J_{\kappa+1} = J_\kappa \cup \{\kappa\},\ \ L_{\kappa+1}(Z, x, y) = \sup_{j \in J_{\kappa+1}} \{l_j(Z, x, y)\} \tag{51}$$

### 3.2.2 Algorithm and Convergence

The branching rule is as follows. If

$$i_\kappa \in \text{argmax}\{|x_i(M_\kappa)y_i(M_\kappa) - 1|\ :\ i = 1, 2, ..., m\}$$

then we divide $M_\kappa$ into two smaller rectangles via the line $x_{i_\kappa} = \frac{1}{2}(x_{i_\kappa}(M_\kappa) + \frac{1}{y_{i_\kappa}(M_\kappa)})$.

**Linear Programming based algorithm (LPA)**

*Initialization.* Start with $M_0 = [a, b]$ and a point $(Z^0, x^0, y^0)$ such that $Z^0 \in \mathcal{Z}, a \leq x^0 \leq b, x_i^0 y_i^0 = 1, i = 1, 2, ..., m$ (see (47)). Compute $\pi_0 \in \partial f(Z^0, x^0, y^0)$. Let

$$\begin{gathered} l_0(x, y) := \langle \pi_0, (Z - Z^0, x - x^0, y - y^0)\rangle + f(Z^0, x^0, y^0), \\ J_1 = \{0\},\ \ L_1(x, y) = l_0(x, y). \end{gathered}$$

Define $\alpha_0 = f(Z^0, x^0, y^0)$. Set $\mathcal{S}_1 = \mathcal{N}_1 = \{M_0\}$. Set $\kappa = 1$.

*Step 1.* For each $M \in \mathcal{N}_\kappa$ solve $LP(M)$ to obtain $\beta(M)$ and $(Z(M), x(M), y(M))$. Define the current best solution $(Z^\kappa, x^\kappa, y^\kappa)$ by comparing $f(Z^{\kappa-1}, x^{\kappa-1}, y^{\kappa-1})$ with $f(Z(M), x(M), \tilde{y}(M)), f(Z(M), \tilde{x}(M), y(M))$, for $M \in \mathcal{N}_\kappa$, where $\tilde{y}_i(M)x_i(M) = 1,\ \tilde{x}_i(M)y_i(M) = 1,\ i = 1, 2$. If $\alpha_\kappa = f(Z^\kappa, x^\kappa, y^\kappa) < 0$, terminate.

*Step 2.* In $\mathcal{S}_\kappa$ delete all $M$ such that $\beta(M) \geq 0$. Let $\mathcal{R}_\kappa$ be the set of remaining rectangles. If $\mathcal{R}_\kappa = \emptyset$, terminate: the problem (45) has no solution.

*Step 3.* Choose $M_\kappa \in \text{argmin}\{\beta(M) |\ M \in \mathcal{R}_\kappa\}$ and divide it into two smaller rectangles $M_{\kappa,1}, M_{\kappa,2}$ according to the above specified rule. Let $\mathcal{N}_{\kappa+1} = \{M_{\kappa,1}, M_{\kappa,2}\}$, $\mathcal{S}_{\kappa+1} = (\mathcal{R}_\kappa \setminus M_\kappa) \cup \mathcal{N}_{\kappa+1}$.

*Step 4.* If $f(Z(M_\kappa), x(M_\kappa), y(M_\kappa)) = \beta(M_\kappa)$, then let $J_{\kappa+1} = J_\kappa, L_{\kappa+1}(Z, x, y) = L_\kappa(Z, x, y)$. Otherwise, $f(Z(M_\kappa), x(M_\kappa), y(M_\kappa)) > \beta(M_\kappa)$, then take $\pi_\kappa \in \partial f(Z(M_\kappa), x(M_\kappa), y(M_\kappa))$, and define $J_{\kappa+1} = J_\kappa \cup \{\kappa\}$ and $l_\kappa, L_{\kappa+1}$ by (50), (51).
Set $\kappa \leftarrow \kappa + 1$ and go back to Step 1.

An analogue to Proposition 2 holds for LPA, namely:

**Proposition 3** *If Algorithm 3 is infinite, then the system (4)-(5) is infeasible and the optimal value of problem (45) is zero.*

*Proof* Let $M_\nu := M_{\kappa_\nu} = [p^\nu, q^\nu]$, $\nu = 1, 2, \dots$ be an infinite nested sequence of rectangles as in Lemma 1. As in the proof of Lemma 1 we may assume $(Z(M_\nu), x(M_\nu), y(M_\nu)) \to (\bar{Z}, \bar{x}, \bar{y})$ with $\bar{x}_i\bar{y}_i = 1$, $i = 1, 2, ..., m$, i.e. $(\bar{Z}, \bar{x}, \bar{y})$ satisfies (5). Furthermore, by an outer approximation argument [34] we now show that

$$\beta(M_\nu) \to \bar{\beta} = f(\bar{Z}, \bar{x}, \bar{y}). \tag{52}$$

Let $u^\nu$ and $\bar{u}$ stand for $(Z(M_\nu), x(M_\nu), y(M_\nu))$ and $(\bar{Z}, \bar{x}, \bar{y})$, and $\beta^\nu$ stand for $\beta(M_\nu)$.

Note that

$$l_i(u^\nu) \le \beta^\nu < f(u^\nu) \ \forall i < \kappa_\nu. \tag{53}$$

Since $\{u^\nu\}$ is bounded, $\{\pi^\nu \in \partial f(u^\nu)\}$ is bounded, too [23, Theorem 24.7], so we may assume $\pi^\nu \to \bar{\pi} \in \partial f(\bar{u})$. Since $l_{\kappa_\nu}(u) = \langle \pi^\nu, u - u^\nu \rangle + f(u^\nu)$, it follows that $l_{\kappa_\nu}(u) \to \langle \bar{\pi}, u - \bar{u} \rangle + f(\bar{u})\ \forall u$. In particular

$$l_{\kappa_\nu}(\bar{u}) \to f(\bar{u}). \tag{54}$$

But from (53)

$$f(\bar{u}) \ge \bar{\beta}, \tag{55}$$

while by fixing $i$ we have $l_i(u^\nu) \le \beta^{\kappa_\nu}\ \forall \kappa_\nu > i$, so, by letting $\nu \to \infty$, we get $l_i(\bar{u}) \le \bar{\beta}$ , for any $i = 0, 1, \dots$. In particular

$$l_{\kappa_\nu}(\bar{u}) \le \bar{\beta}. \tag{56}$$

From (54) and (56) it follows that $f(\bar{u}) \le \bar{t}$, hence, in view of (55): $\bar{\beta} = f(\bar{u})$, proving (52). Now, since $t(M_\nu) = \beta(M_\nu) \le \min\{f(Z, x, y) | \quad (5)\ \}$, it follows that

$$f(\bar{Z}, \bar{x}, \bar{y}) \le \min\{f(Z, x, y) | \quad (5)\ \}$$

But, as shown above, $(\bar{Z}, \bar{x}, \bar{y})$ satisfies (5). Hence $f(\bar{Z}, \bar{x}, \bar{y}) = \min\{f(Z, x, y) | \quad (5)\ \}$ just as in the proof of Proposition 1. The rest is clear. □

## 3.3 Hybrid algorithms

The idea behind this hybrid algorithms is that in the first stage of the search procedure, it does not pay to compute very accurate bounds, so less computationally expensive algorithms like linear programming based or local search ones can be utilized, but at an advanced stage, as it becomes more and more important to have accurate bounds, better results could be obtained by using LMI based algorithm with a global search instead.

Take the problem (45). Since $LP(M)$ increases in size with $\kappa$, solving $LP(M)$ becomes more and more time-consuming. Furthermore, many of the constraints (48) may be redundant or may correspond to very poor approximations of the constraint $f(Z, x, y) \le t$ for $x \in M$. Therefore, in practical implementation, when $\kappa$ exceeds a certain number, say $\kappa > \kappa_0$, it may be better to replace (48) simply by the original constraint $f(Z, x, y) \le t$ and to compute a lower bound for $M = [p, q]$ by solving (40) rather than $LP(M)$.
An alternative way is to use an local search algorithm like Frank and Wolf algorithm (FW) [9] which is less computational expensive, at the first stage to obtain a good *upper bound* and then use a global search algorithm like CPA, DCPA, LPA at the final stage.

## 4 BMI Feasibility Problem

### 4.1 Concave programming reformulation of general BFP

Return to consider BFP (6) which can be written as

$$F_0 + \sum_{i=1}^{N} x_i + \sum_{i\le j\le N}^{N} w_{ij}F_{ij} < 0 \tag{57}$$

$$x \in [a,b],\ w_{ij} = w_{ji},\ i,j = 1,\dots,N, \tag{58}$$

$$w_{ij} = x_i x_j,\ 1 \le i \le j \le N. \tag{59}$$

Thus the difficulty of BFP (57)-(59) is $N(N+1)/2$ nonconvex quadratic constraints (59). Therefore BFP (57)-(59) belongs to the class of nonconvex indefinite quadratic programming, which is one of most important classes in global optimization and has been a subject of intensive investigation in recent years due to its numerous applications [1, 14, 26, 32, 8, 2, 33]. In this section, we will show a new solution method for BFP (57)-(59) based on concave programming.
In the following Lemma, we reduce $N(N+1)/2$ indefinite quadratic equality constraints (59) to just one reverse convex constraint.

**Lemma 5** *Let $W$ be the $N \times N$ symmetric matrix consisting from $w_{ij}$ in (59), i.e. $W = [w_{ij}]_{1\le i,j\le N}, w_{ij} = w_{ji}\ \forall(i,j)$. Then $N(N+1)/2$ indefinite quadratic equality constraints (59) are equivalent to*

$$\begin{bmatrix} W & x \\ x' & 1 \end{bmatrix} \ge 0, \tag{60}$$

$$Trace(W - xx') = Trace(W) - \sum_{i=1}^{N} x_i^2 \le 0 \tag{61}$$

*where (60) is a LMI and (61) is reverse convex.*

*Proof* Since the implication (59) $\Rightarrow \{(60),(61)\}$ is obvious and we have to show only the inverse implication. But (60) is equivalent to $W - xx' \ge 0$ which particularly also implies $\text{Trace}(W - xx') \ge 0$. Therefore, in view of (61) we get $W - xx' \ge 0$ and $\text{Trace}(W - xx') = 0$ which are equivalent to $W - xx' = 0$, i.e. (59). $\square$

Now, from (60)-(61) we see that BFP (6) is feasible if and only if 0 is the optimal value of the following optimization problem

$$\min \text{Trace}(W) - \sum_{i=1}^{N} x_i^2 : \ (57),(58),(60). \tag{62}$$

Clearly, the objective function of (62) has the representation (10) with $y = W, f_1(W) = \text{Trace}(W), f_2(x) = -\sum_{i=1}^{N} x_i^2$ satisfying condition $(i),(ii)$ in Section 2. Therefore, we can propose CPA for solving this nonconvex program (62). In this case, applying formula (13), program (9) for computing a lower bound $\beta(M)$ of (62) for $x \in M = [p,q] \subset [a,b]$ is

$$\min\ \text{Trace}(W) - \sum_{i=1}^{N}[(p_i+q_i)x_i - p_iq_i] : \ (58),(60),\ x \in [p,q] \tag{63}$$

and the adaptive rule is concretized analogously.
Of course the local search algorithms like FW should be very helpful for computing a good upper bound to reduce the computing time of this algorithm.
Finally, if instead of BFP (57)-(59) we have to solve the the nonconvex optimization problem

$$\min \langle c, x\rangle : \ (57) - (59) \tag{64}$$

with $c \in R^N$, then it is better to using another equivalent representation of the nonconvex constraint (59) instead of (60)-(61). Namely, like (60)-(61) we can show that (59) is equivalent to (60) with

$$w_{ii} - x_i^2 \le 0, \ i = 1, 2, ..., N. \tag{65}$$

Note that (65) is just reverse convex constraints since the functions in its left hand side are concave. Thus, $N^2$ nonconvex *equality* constraints (59) have been substantially reduced to $N$ reverse nonconvex *inequality* constraints (65). An interesting feature of (65) is that it is decoupling in $x_i$ and thus convenient for a tight relaxation. Using (12) for computing the tight affine minorant of function $-x_i^2$, the resulted lower bound of (64) with $x \in M = [p, q]$ is provided by the optimal solution of the following LMI optimization programm

$$\begin{array}{rcl}\min \langle c, x\rangle & : & (57), (60), x \in [p, q], \\ & & w_{ii} - (p_i + q_i)x_i + p_i q_i \le 0, \ i = 1, \ldots, N.\end{array} \tag{66}$$

Based on this lower bound computing, like DCPA it is easy to produce a corresponding rectangular branch and bound algorithm.

## 4.2 Decomposition method for a special class of BMIs

In this subsection we will concentrate to the feasibility of the following particular case of BMI (6),

$$F_{00} + \sum_{i=1}^{N_1} x_i F_{i0} + \sum_{j=1}^{N_2} y_j F_{0j} + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} x_i y_j F_{ij} < 0, \tag{67}$$

$$x \in [a, b] \subset R^{N_1}, \ y \in [c, d] \subset R^{N_2}, a > 0, c > 0 \tag{68}$$

where $F_{ij}$ are symmetric matrices.
Without loss of generality we may assume $N_1 \le N_2$ (if $N_2 < N_1$ the roles of $x$ and $y$ are interchanged). BFP of (67), (68) has been considered first in [12, 13].
Obviously one can reduce (67)-(68) to the form (57)-(59) and then apply CPA developed in subsection 4.2 for solving (67)-(68). However, it is not best way to handle (67)-(68) since it requires branching performed on the space of *all* variables $(x, y)$. It is quite known in global optimization that the efficiency of a global optimization algorithm critically depends on its branching space and this fact motivates the recent decomposition methods in global optimization (see e.g. [15, 32]). This is very natural since in general the iteration number of a branch and bound algorithm depends *exponentially* on the dimension of its branching space. Now, (67)-(68) can be writen as

$$\min \ f(z, x, y) : (68), z_{ij} = x_i y_j, \ i = 1, 2, ..., N_1; \ j = 1, 2, ..., N_2, \tag{69}$$

with

$$f(z,x,y) = \lambda_{\max}[F_{00} + \sum_{i=1}^{N_1} x_i F_{i0} + \sum_{j=1}^{N_2} y_j F_{0j} + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} z_{ij} F_{ij}]$$

((67)-(68) is feasible if and only if the optimal value of (69) is negative).
Note the following fundamental *bilinear* features of (69): it becomes just *convex* LMI program when variable $x$ is held fixed. Thus from development in Section 2, considering $x$ as the "complicating variables", an effcient way to handle BFP (69) should be a branch and bound with branching performed on $x-$space of dimension $N_1 = \min\{N_1, N_2\}$ instead of $(x,y)$-space of dimension $N_1+N_2$. The resulted algorithms presented below with branching on that way may be practical even for large dimension of $y$ but not so large dimension of $x$.

### 4.2.1 LMI based method

**Bounding**

Given a rectangle $M = [p,q] \subset [a,b]$ the computation of a lower bounding $\beta(M)$ for values of $f(z,x,y)$ over the feasible points

$$z_{ij} = x_i y_j,\ i = 1,\dots,N_1,\ j = 1,\dots,N_2;\ (x,y) \in M \times [c,d] \tag{70}$$

is based on the following

**Lemma 6** *Assume* $p_i < q_i, c_j < d_j$. *We have*

$$p_i \le x_i \le q_i,\ \ c_j \le y_j \le d_j \tag{71}$$

*if and only if there exists* $z_{ij}$ *satisfying*

$$z_{ij} \ge \max\{c_j x_i + p_i y_j - p_i c_j, d_j x_i + q_i y_j - q_i d_j\} \tag{72}$$

$$z_{ij} \le \min\{c_j x_i + q_i y_j - q_i c_j, d_j x_i + p_i y_j - p_i d_j\}. \tag{73}$$

*Proof* The inequalities (71) imply

$$(x_i - p_i)(y_j - c_j) \ge 0, \quad (q_i - x_i)(d_j - y_j) \ge 0, \tag{74}$$

$$(x_i - p_i)(y_j - d_j) \le 0, \quad (x_i - q_i)(y_j - c_j) \le 0 \tag{75}$$

hence

$$x_i y_j \ge c_j x_i + p_i y_j - p_i c_j; \quad x_i y_j \ge d_j x_i + q_i y_j - q_i d_j, \tag{76}$$

$$x_i y_j \le d_j x_i + p_i y_j - p_i d_j; \quad x_i y_j \le c_j x_i + q_i y_j - q_i c_j, \tag{77}$$

Thus (72)-(73) holds with $z_{ij} = x_i y_j$.
Conversely, suppose $x_i, y_j$ satisfy (72)-(73) for some $z_{ij}$. From

$$z_{ij} \ge c_j x_i + p_i y_j - p_i c_j$$
$$z_{ij} \le c_j x_i + q_i y_j - q_i c_j$$

we get $0 \ge (c_j x_i + p_i y_j - p_i c_j) - (c_j x_i + q_i y_j - q_i c_j) = (p_i - q_i)(y_j - c_j)$, hence $y_j \ge c_j$ because $q_i - p_i > 0$. The other inequalities in (71) can be derived analogously.

□

Note that each of the inequalities (72), (73) splits into two linear inequalities, so the system (71)-(73) is actually a linear system. As a consequence of the above Lemma, a lower bound on $M = [p, q]$ is provided by the optimal value $\beta(M)$ of the subproblem (equivalent to a linear program under LMI constraints):

$$
\begin{array}{lll}
CQ(M) & \min\ f(z, x, y)\ : & \\
& z_{ij} \geq \max\{c_j x_i + p_i y_j - p_i c_j, d_j x_i + q_i y_j - q_i d_j\} & (78)\\
& z_{ij} \leq \min\{c_j x_i + q_i y_j - q_i c_j, d_j x_i + p_i y_j - p_i d_j\} & (79)\\
& i = 1, \ldots, N_1,\ j = 1, \ldots, N_2 & (80)
\end{array}
$$

Furthermore, by Lemma 6, $p_i \leq x_i(M) \leq q_i$, $c_j \leq y_j(M) \leq d_j$, $\forall i, j$ so $(\tilde{z}(M), x(M), y(M))$ with $\tilde{z}_{ij}(M) = x_i(M) y_i(M)$ is a feasible solution to BFP and $\alpha(M) := f(\tilde{z}(M), x(M), y(M))$ yields an upper bound for (69). Thus, the subproblem $CQ(M)$ provides not only a lower bound but also an upper bound.

**Remark 3.** The fact that the nonconvex constraint (70) can be relaxed to (79)-(80) and

$$(x, y) \in [p, q] \times [c, d] \tag{81}$$

is quite well known since the work of McCormic [18]. In view of our lemma 6, the constraint (81) is however superflous.
Moreover, it seems that most of control people don't know this well known fact in global optimization, which has been subsequently described in the text books on optimization [19, 14] or any paper on indefinite quadratic optimization (see e.g. [1, 2, 8, 26] for a few). For instance, in [12, 13], (70) is relaxed just to

$$x \in [p, q],\ y \in [c, d],\ p_i c_j \leq z_{ij} \leq q_i d_j, i = 1, \ldots, N_1,\ j = 1, \ldots, N_2 \tag{82}$$

which is a very coarse approximation of the nonconvex set (70).
On the other hand, the following relaxed constraint for (70)

$$x \in [p, q],\ y \in [c, d], z_{ij} \in \operatorname{conv}\{(p_i, c_j, p_i c_j), (p_i, d_j, p_i d_j), (q_i, c_j, q_i c_j), (q_i, d_j, q_i d_j)\} \tag{83}$$

used in [10] is nothing else than (79)-(80), (81), which has been also essentially established in [19]. However, the form (79)-(80), (81) which is an explicit description of (83), is more convenient for computational implementation.
Using the general BB scheme [16], both [12, 13] and [10] don't take the bilinear charaterization of (69) into account for BB processes. The branching in their BB algorithms is performed in the $(x, y)$-space (of dimension $N_1 + N_2$) rather than in the $x$-space (of dimension $N_1 \leq N_2$), and thus can be not practical for the case of large $N_1 + N_2$. Finally, the computation of the current best solution in [12, 13] involves finding a local minimum of $f(x, y, z)$ over $(x, y) \in [p, q] \times [r, s], z_{ij} = x_i y_j, i = 1, \ldots, N_1, j = 1, \ldots, N_2$, which may be computationally expensive since this local optimization problem is also NP-hard for itself.
Clearly, problem (69) belongs to the class of d.c. optimization since the indefinite quadartic constraints $z_{ij} = x_i y_j$ in (69) are a particular case of d.c. constraints [14, 32]. A d.c.

structure of (69) in quite different setting has observed in [17]. This is not surprising since there are infinite d.c. representations for a given problem and in principle, every optimization problem can be transformed to a d.c. optimization problem [28, 33]. However, the efficiency of a d.c. optimization algorithm does depend on the used d.c. representation. The d.c. representation of [17] leads to solving (nonconvex) concave programs at every iteration for computing a lower bound, which is obviously very computationally expensive. This is sharp constrast to our d.c. optimization algorithm, which requires solving only (convex) LMI or linear programms at every iteration.
Now, our branching performed only on the space of variables $x$ is looked as follows.

**Branching**

Let $M$ be the candidate for further subdivision at a given iteration and let $(z(M), x(M), y(M))$ be an optimal solution of $CQ(M)$. If $z_{ij}(M) = x_i(M)y_j(M)\ \forall(i,j)$, then it is a feasible solution to BMI and since $\beta(M)$ is a lower bound for (69) it follows that $(z(M), x(M), y(M))$ solves BFP. Thus, to ensure convergence the subdivision rule should tend to bring the difference $\max_{ij}\{|z_{ij}(M) - x_i(M)y_j(M)|\}$ to zero. To achieve this, we set

$$(i_M, j_M) \in \text{argmax}\{|z_{ij}(M) - x_i(M)y_j(M)|\ (i,j) \in N_1 \times N_2\}$$

and divide $M = [p,q]$ into two smaller rectangles by the plane

$$x_{i_M} = \frac{1}{2}(x_{i_M}(M) + \frac{z_{i_M j_M}(M)}{y_{i_M}(M)}).$$

**LMI based BB algorithm for BFP (LMIBBA)**

*Initialization.* Start with $M_0 = [a,b]$. Set $\mathcal{S}_1 = \mathcal{N}_1 = \{M_0\}$. Set $\kappa = 1$.

*Step 1.* For each $M \in \mathcal{N}_\kappa$ solve $CQ(M)$ to obtain the optimal value $\beta(M)$ and an optimal solution $(z(M), x(M), y(M))$. Define $(z^\kappa, x^\kappa, y^\kappa)$ to be the best among $(z^{\kappa-1}, x^{\kappa-1}, y^{\kappa-1})$ and $(\tilde{z}(M), x(M), y(M))$, $(z(M), \tilde{x}(M), y(M))$, $(z(M), x(M), \tilde{y}(M))$, $M \in \mathcal{N}_\kappa$, where $\tilde{z}_{ij}(M) = x_i(M)y_j(M)$, $\tilde{y}_i(M) = z_{ij}(M)/x_j(M)$, $\tilde{x}_i(M) = z_{ij}(M)/y_j(M)$. If $\alpha_\kappa = f(x^\kappa, y^\kappa, z^\kappa) < 0$, terminate: $(x^\kappa, y^\kappa)$ is a feasible solution of BMI (67)-(68.

*Step 2.* In $\mathcal{S}_\kappa$ delete all $M$ such that $\beta(M) \geq 0$. Let $\mathcal{R}_\kappa$ be the set of remaining rectangles. If $\mathcal{R}_\kappa = \emptyset$, terminate: BMI (67)-(68) is infeasible.

*Step 3.* Choose $M_\kappa \in \text{argmin}\{\beta(M)|\ M \in \mathcal{R}_\kappa\}$ and divide it into two smaller rectangles $M_{\kappa,1}, M_{\kappa,2}$ according to the above specified subdivision rule 3. Let $\mathcal{N}_{\kappa+1} = \{M_{\kappa,1}, M_{\kappa,2}\}$, $\mathcal{S}_{\kappa+1} = (\mathcal{R}_\kappa \setminus M_\kappa) \cup \mathcal{N}_{\kappa+1}$.

Set $\kappa \leftarrow \kappa + 1$ and go back to Step 1.

The following convergence proposition can be established similarly to Proposition 1.

**Proposition 4** *Either LMIBBA terminates after finitely many iterations, yielding a feasible solution to BFP (termination at step 1) or producing the evidence that BFP is infeasible (termination at step 2). Or it is infinite and then the system (67)-(68) is infeasible and the optimal value of problem (69) is zero.*

## 4.3 Linear programming based method

Just as in LPA for solving LFP, instead of solving convex programs (by LMI solvers) for lower bounding, we can also solve iterative linearizations of these programs, at least in the first stage of the procedure, when highly accurate lower bounds may not be worth the computational effort to obtain them. Thus, the lower bounding subproblem at each iteration is constructed by linearizing the convex constraints and replacing the convex objective function $f(z,x,y)$ by a convex piecewise affine minorant approaching $f(z,x,y)$ more and more closely as the algorithm proceeds.

**Bounding**

At iteration $\kappa$ we already have in hand a convex piecewise affine minorant $L_\kappa(z,x,y)$ of $f(z,x,y)$, of the form

$$L_\kappa(z,x,y) = \max\{l_j(z,x,y)|\ j \in J_\kappa\},$$

where $l_j(z,x,y), j \in J_\kappa$, are affine functions resulting from the previous iterations. A lower bound $\beta(M)$ for (69) is then provided by the optimal value of the linear program

$$LQ(M) \qquad \begin{array}{l} \min\ t\ : (78) - (80) \text{ and} \\ l_j(z,x,y) \le t,\ j \in J_\kappa \end{array}$$

**Updating $L_\kappa(z,x,y)$**

At iteration $\kappa$ let $M_\kappa$ be the rectangle with smallest $\beta(M)$, i.e. the candidate for further splitting. Let $(t(M_\kappa), z(M_\kappa), x(M_\kappa), y(M_\kappa))$ be an optimal solution of $LQ(M_\kappa)$. If $f(z(M_\kappa), x(M_\kappa), y(M_\kappa)) = t(M_\kappa)$, then set $L_{\kappa+1}(x,y) = L_\kappa(x,y)$, $J_{\kappa+1} = J_k$. Otherwise, $f(z(M_\kappa), x(M_\kappa), y(M_\kappa)) > t(M_\kappa)$, then $(t(M_\kappa), z(M_\kappa), x(M_\kappa), y(M_\kappa))$ does not belong to the convex set $\{(t,z,x,y)|\ f(z,x,y) \le t\}$ and we can separate it from the latter by the cutting plane

$$l_\kappa(z,x,y) \le t \tag{84}$$

where

$$\begin{aligned} l_\kappa(z,x,y) &= \langle \pi_\kappa, (z - z(M_\kappa), x - x(M_\kappa), y - y(M_\kappa))\rangle \\ &\quad + f(z(M_\kappa), x(M_\kappa), y(M_\kappa)) \end{aligned} \tag{85}$$

with $\pi_\kappa \in \partial f(z(M_\kappa), x(M_\kappa), y(M_\kappa))$.
So, when $f(z(M_\kappa), x(M_\kappa), y(M_\kappa)) > t(M_\kappa)$, we define

$$J_{\kappa+1} = J_\kappa \cup \{\kappa\},\ L_{\kappa+1}(z,x,y) = \sup_{j \in J_{\kappa+1}} \{l_j(z,x,y)\} \tag{86}$$

**LP based BB algorithm for BMI (LPBBA)**

*Initialization.* Start with $M_0 = [a,b]$ and a feasible point $(z^0, x^0, y^0)$ to (69). Compute $\pi_0 \in \partial f(z^0, x^0, y^0)$. Let

$$\begin{gathered} l_0(x,y) := \langle \pi_0, (z - z^0, x - x^0, y - y^0)\rangle + f(z^0, x^0, y^0), \\ J_1 = \{0\},\ L_1(x,y) = l_0(x,y). \end{gathered}$$

Define $\alpha_0 = f(z^0, x^0, y^0)$. Set $\mathcal{S}_1 = \mathcal{N}_1 = \{M_0\}$. Set $\kappa = 1$.

*Step 1.* For each $M \in \mathcal{N}_\kappa$ solve $LQ(M)$ to obtain $\beta(M)$ and $(z(M), x(M), y(M))$. Define $(z^\kappa, x^\kappa, y^\kappa)$ to be the best among all the feasible $(z(M), x(M), \tilde{y}(M))$, $(z(M), \tilde{x}(M), y(M))$, $(\tilde{z}(M), x(M), y(M))$ $M \in \mathcal{N}_\kappa$ where $\tilde{y}_i(M) = z_{ij}(M)/x_j(M)$, $\tilde{x}_i(M) = z_{ij}(M)/y_j(M)$, $\tilde{z}_{ij}(M) = x_i(M)y_j(M)$. If $\alpha_\kappa = f(x^\kappa, y^\kappa, z^\kappa) < 0$ terminate: $(x^\kappa, y^\kappa)$ is a feasible solution of (67)-(68).

*Step 2.* In $\mathcal{S}_\kappa$ delete all $M$ such that $\beta(M) \geq 0$. Let $\mathcal{R}_\kappa$ be the set of remaining rectangles. If $\mathcal{R}_\kappa = \emptyset$, terminate: BMI (67)-(68) is infeasible.

*Step 3.* Choose $M_\kappa \in \operatorname{argmin}\{\beta(M) |\ M \in \mathcal{R}_\kappa\}$ and divide it into two smaller rectangles $M_{\kappa,1}, M_{\kappa,2}$ according to the same subdivision rule as in LMIBBA. Let $\mathcal{N}_{\kappa+1} = \{M_{\kappa,1}, M_{\kappa,2}\}$, $\mathcal{S}_{\kappa+1} = (\mathcal{R}_\kappa \setminus M_\kappa) \cup \mathcal{N}_{\kappa+1}$.

*Step 4.* If $f(z(M_\kappa), x(M_\kappa), y(M_\kappa)) = t(M_\kappa)$, then let $J_{\kappa+1} = J_\kappa, L_{\kappa+1}(z, x, y) = L_k(z, x, y)$. Otherwise, $f(z(M_\kappa), x(M_\kappa), y(M_\kappa)) > t(M_\kappa)$, then take $\pi_\kappa \in \partial f(z(M_\kappa), x(M_\kappa), y(M_\kappa))$, and define $J_{\kappa+1} = J_\kappa \cup \{\kappa\}$ and $l_\kappa, L_{\kappa+1}$ by (85), (86).
Set $\kappa \leftarrow \kappa + 1$ and go back to Step 1.

The proof of the following result is analogous to that of Proposition 4.

**Proposition 5** *If LPBBA is infinite, then the system (67)-(68) is infeasible and the optimal value of problem (69) is zero.*

Just as in the case of LFP, a hybrid Algorithm for BFP can be used which in a first stage proceeds according to LPBBA and later swtiches to LMIBBA.

## 5 Computational experience

In this Section, we present some computational experience with the above developed algorithms for LFP and BFP, using MATLAB on PC with CPU Pentium 133 MHz.

### 5.1 LFP

CPA and DCPA for LFP were tested on problems with $k = 2, 3, 4$ (i.e. $m = 1, 2, 3$ in the corresponding LFP (4)-(5)) and with the dimension of the state space of the plant (1) ranging from 5 to 10 (hence the dimension of the space containing $Z$ is ranging from 30 to 110). For each value of $k$, up to 150 randomly generated instances were solved, with $a_i = 0.001$, and $b_i = 1000$ in (30). The value $\gamma$ in (29) was chosen to be equal to $\gamma_0/\ell$ for some $\ell \geq 2$, where $\gamma_0$ is the minimum of $\gamma$ such that LMI (29) with fixed $x_i = 1$, $y_i = 1$ is feasible (i.e. $\gamma_0$ is the optimal value without scaling). For such $\gamma$ LFP had a feasible solution in 60% of our examples and no feasible solution in the remaining 40%.
The results are presented in Table 1 which tell us that CPA performs better than DCPA. The figures in brackets in the AI column are the average numbers of iterations needed for computing the optimal value of problem (35)-(36), with tolerance $\varepsilon = 0.01$. These numbers represent the upper bounds of the average number of iterations for any $\gamma$ in LFP.
In more than 80% of our examples, the optimal value of problem (35)-(36), lies in the interval $[-0.2, 0.2]$ which is a small neighborhood of zero. In the remaining 20% it lies in the interval $[-0.9, 0.9]$. This means that the statistics given in Table 1 relate more or less to "worst cases" of LFP. As it often happens in deterministic global optimization, the optimal value is often found very early, but most of the time is spent on checking

their optimality. Since the termination criterion of Algorithm 1 is that either the current best value of $f(Z, x, y)$ is negative or the smallest lower bound $\beta(M_\kappa))$ is greater than $-\varepsilon$, this suggests that Algorithm 1 should require much less iterations if the optimal value lies outside the interval, say, $[-1.5, 1.5]$.

As expected, the number of iterations depends on $m$ but is little sensitive to the dimension of $Z$. To investigate the role of the subdivision rules, we have also run the algorithm by using different subdivision rules. It turned out that with regard to performance, the subdivision rules are ranked as follows:

1. the above proposed adaptive subdivision rule;
2. the adaptive bisection: $i_M$ is chosen as in (14), but the subdivision line is $x_{i_M} = \frac{1}{2}(p_{i_M}(M) + q_{i_M}(M))$;
3. the special rule: $i_M$ is chosen as in (14), but the subdivision line is $x_{i_M} = x_{i_M}(M)$;
4. the standard bisection: divide $M$ by a line orthogonal to the longest edge at the midppoint of this edege.

Thus the best subdivision rule is the one proposed above, while the standard bisection is by far the worst: the number of iterations can increase dramatically (from two to ten times) with the standard bisection when $k = 3, 4$.

Our computational experiments also show that the performance of DCPA does not change when the convex constraint (31) in (40) is replaced by its linear relaxation (46).

Note that a tighter upper bound for (35)-(36), in Step 1 could be obtained by solving two LMI problems

$$\min\ t:\ \mathcal{L}(Z, \tilde{x}, \tilde{x}^{-1}) < tI \tag{87}$$

with $\tilde{x} \in \{x(M), y(M)^{-1}\}$.

However, our experience shows that although slightly less iterations will be needed because of better initial upper bounds, more total CPU time will be required because of the time spent on solving these two additional LMI problems.

| Alg. | k | AI | MBN | ACPU5-10 |
|---|---|---|---|---|
| CPA | 2 | 9.2(17.8) | 5 | 1.45'-3.62' |
| DCPA | 2 | 13.4(25.3) | 7 | 2.05'-5.13' |
| CPA | 3 | 25.63(40.2) | 13 | 6.45'-15.79' |
| DCPA | 3 | 40.75(70.2) | 19 | 10.2'-24.92' |
| CPA | 4 | 35.23 (60.5) | 25 | 10.35'-33.72' |
| DCPA | 4 | 97.53 (120.5) | 69 | 24.28'-74.98' |

Tab. 1:AI: average # of iterations, MBN: max # of branches,
ACPU5-10: average CPU time/LFP when the state dimension varies from 5 to 10

## 5.2 BFP

LMIBBA and LPBBA were tested on problems with $N_1 = 1, 2, 3$ and $N_2 \leq N_1, N_2$ from 1 to 3. Specifically, we solved problems where (67) has the form

$$F_{00} + x_1 F_{01} + y_1 F_{10} + x_1 y_1 F_{11} \quad < \quad 0, \tag{88}$$

$$F_{00} + \sum_{i=1}^{3} x_i F_{0i} + y_1 F_{10} + \sum_{i=1}^{3} x_i y_1 F_{i1} \quad < \quad 0, \tag{89}$$

$$F_{00} + \sum_{i=1}^{2} (x_i F_{0i} + y_i F_{i0}) + \sum_{i=1}^{2} x_i y_i F_{ii} \quad < \quad 0, \tag{90}$$

$$F_{00} + \sum_{i=1}^{3} (x_i F_{0i} + y_i F_{i0}) + \sum_{i=1}^{3} x_i y_i F_{ii} \quad < \quad 0 \tag{91}$$

Since $N_2 \leq N_1$, the method branches upon $y$. So the space on which branching is perfomed is $R^1$ for (89), $R^2$ for (90), $R^3$ for (91). In each case, up to 150 randomly generated instances were solved, in which all variables are restricted to the interval $[0.001, 1000]$ and all matrices are of order from $3 \times 3$ to $10 \times 10$. The computational results when tolerance $\varepsilon = 0.1\%$ are shown in Table 2. To improve the performance of the algorithms, we also *restart* the algorithms (see e.g. [15, Chap.6]), when the number of partition sets has become too large: if $\alpha_\kappa$ is the incumbent value at the last iteration, then add the constraint $f(x, y, z) \leq \alpha_\kappa$ for the next cycle of iterations. This restart strategy is useful for keeping the number of partition sets within manageable limits and often speeds up the convergence. In all our tested examples, the optimal value is located in the segment $[-0.9, 0.9]$ which is a small neighborhood of 0. Thus, the results presented in Table 2 relate more or less to worst cases.

From the statistics of Table 2 we see that for the cases under consideration (small $N_1, N_2$ and small size of all matrices) LPBBA performs much better than LMIBBA. The number of iterations needed for both algorithms is almost the same, despite the fact that the constraints in the bounding subproblems in LMIBBA are more accurate.

| Alg. | BMI | AI for BFP | AI for OV | CPUtime for BFP | CPUtime for OV |
|---|---|---|---|---|---|
| LMIBBA | (88) | 5.12 | 20.13 | 13.51 sec. | 40.03 sec. |
| LPBBA | (88) | 7.01 | 22.70 | 1.40 sec. | 3.95 sec. |
| LMIBBA | (89) | 10.13 | 37.43 | 60.05 sec. | 214.48 sec. |
| LPBBA | (89) | 13.90 | 38.80 | 5.40 sec. | 20.35 sec. |
| LMIBBA | (90) | 16.25 | 38.9 | 65.99 sec. | 176.18 sec. |
| LPBBA | (90) | 18.27 | 32.00 | 5.53 sec. | 11.27 sec. |
| LMIBBA | (91) | 30.63 | 58.09 | 206.00 sec. | 320.17 sec. |
| LPBBA | (91) | 33.00 | 57.50 | 18.83 sec. | 46.99 sec. |

Tab. 2:AI: average # of iterations; OV: optimal value of (69)

Also note from the statistics in Table 2 that the degree of difficulty of a problem depends both on the number of "complicating variables" and the number of nonconvex constraints. For instance, both (88) and (89) have just one complicating variable (the space in which branching is performed is $R^1$), but (89) has 3 nonconvex constraints $z_{i1} = x_i y_1, i = 1, 2, 3$ compared with 1 nonconvex constraint $z_{11} = x_1 y_1$ in (88). So to reach the optimal value requires much more iterations in (89) than in (88).

For other computational examples using our LMIBBA and LPBBA to solve BMIs like (67) with $N_2 > 10N_1$ arising from robust constrained nonlinear control problems we refer the reader to [27]. Some comparisions of our methods with other can be found in [6].

Finally, it may be of interest to see how our algorithms solve the following example from [13]

$$\begin{aligned}\begin{bmatrix} -10 & -0.5 & -2 \\ -0.5 & 4.5 & 0 \\ -2 & 0 & 0 \end{bmatrix} + y \begin{bmatrix} -1.8 & -0.1 & -0.4 \\ -0.1 & 1.2 & -1 \\ -0.4 & -1 & 0 \end{bmatrix} + x \begin{bmatrix} 9 & 0.5 & 0 \\ 0.5 & 0 & -3 \\ 0 & -3 & -1 \end{bmatrix} \\ +xy \begin{bmatrix} 0 & 0 & 2 \\ 0 & -5.5 & 3 \\ 2 & 3 & 0 \end{bmatrix} < 0, \ (x,y) \in [-0.5, 2] \times [-3, 7], \end{aligned} \tag{92}$$

The results are given in Table 3. LMIBBA finds a feasible $(x, y) = (0.7492, 1.8051)$ after the first iteration with CPU time 3.13 sec., while LPBBA finds a feasible $(x, y) = (0.5499, 4.5643)$ after the second iteration with cpu-time 0.33 sec.

| Alg. | OV1 | IT. for OV1 | time1 | OV2 | IT. for OV2 | time2 |
|---|---|---|---|---|---|---|
| LMIBBA | -0.9551 | 7 | 17.19 | -0.9565 | 25 | 56.79 |
| LPBBA | -0.9565 | 16 | 4.06 | -0.9565 | 19 | 4.77 |

Table 3: Alg.: algorithm;OV1 (OV2, resp.): optimal value with tolerance 0.5% (0.001%, resp.) of value of last incumbent;IT.: # of iterations; time1: CPU time for OV1; time2:CPU time for OV2.

## 6 Conclusion

We have demonstrated in this paper that several important classes of robust control problems can be recast as d.c. optimization problems, which become convex when their complicating variables are held fixed. The exploited features of such optimization problems allows us to propose several branch and bound algorithms with branching performed only in the space of complicating variables that makes algorithms practicable.

**Acknowlegement.** We thank Eric Beran from Technical University of Demark and Olav Slupphaug from Norwegian University of Science and Technology for their helphul remarks and fruit discussions on BMI subject which have helped to improve the content of this paper.

APPENDIX

**Lemma 7** *[22]. Let $A_0, A_1, A_2, \dots, A_n$ be symmetric $n \times n$ matrices and $f : R^n \to R$ be a function defined by*

$$f(x) = \lambda_{\max}(A_0 + \sum_{i=1}^{n} x_i A_i) \ \forall x \in R^n.$$

*At a given $a \in R^n$, let $u$ be a unit eigenvector corresponding to the eigenvalue $f(a)$ of the matrix $A_0 + \sum_{i=1}^{n} a_i A_i$. Then $(u'A_1u, u'A_2u, \dots, u'A_nu) \in \partial f(a)$.*

*Proof* We have to show that

$$\sum_{i=1}^{n} u'A_i u(x_i - a_i) \le f(x) - f(a) \ \forall x \in R^n. \tag{93}$$

But the left hand side of (93) satisfies

$$u'(A_0 + \sum_{i=1}^{n} x_i A_i)u - u'(A_0 + \sum_{i=1}^{n} a_i A_i)u = u'(A_0 + \sum_{i=1}^{n} x_i A_i)u - f(a) \le f(x) - f(a)$$

since

$$u'(A_0 + \sum_{i=1}^{n} a_i A_i)u = f(a)$$

by the definition of $u$, and

$$u'(A_0 + \sum_{i=1}^{n} x_i A_i)u \le f(x) \ \forall x \in R^n$$

is obvious (for every unit vector $u$ in $R^n$ and for every symmetric matrix $A$ of dimension $n \times n$ one has $u'Au \le \lambda_{\max}(A)$).

## References

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

[2] F.A. Al-Khayyal, Jointly constrained bilinear programms and related problems: an overview, *Computers and Mathematics with Applications* 19(1990), 53-62.

[3] P. Apkarian, P. Gahinet, A Convex Characterization of Gain-Scheduled $\mathcal{H}_\infty$ controllers, *IEEE Trans. Automatic Control* 40(1995), 853-864. See also pp. 1681.

[4] P. Apkarian, H.D. Tuan, Robust control via concave optimization: local and global algorithms, *Proc of CDC 1998.*

[5] P. Apkarian, H.D. Tuan, Parameterized LMIs in control theory, To appear in *SIAM J. Control and Optimization.*

[6] E. Beran, Methods for optimization based fixed-order control design, *Ph.D. thesis*, Technical University of Denmark, 1997.

[7] J.C. Doyle, A. Packard, K. Zhou, Review of LFTs, LMIs and $\mu$, *Proc. of 30-th IEEE Conf. on Decision and Control*, 1991, 1227-1232.

[8] C. Floudas, V. Visweswaran, Quadratic optimization, in R. Horst and P. Pardalos *eds.), *Handbook on Global Optimization*, Kluwer, 1995, 217-269.

[9] M. Frank, P. Wolfe, An algorithm for quadratic programming, *Naval Res. Log. Quart.* 3(1956), 95-110.

[10] H. Fujioka, K. Hoshijima, Bounds for the BMI eigenvalues problems-A good lower bound and a cheap upper bound, *Trans. of SICE* 33(1997), 616-621.

[11] P. Gahinet, A. Nemirovski, A. Laub, M. Chilali, *LMI control toolbox,* The Math. Works Inc..

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

[13] K.C. Goh, M.G. Safonov, G.P. Papavassilopoulos, A global optimization approach for the BMI problem, *Journal of Global Optimization*, 7(1995), 365-380.

[14] R. Horst, H. Tuy, *Global optimization: deterministic approaches* (3rd edition), Springer, 1996.

[15] H. Konno, P.T. Thach, H. Tuy, *Optimization on Low Rank Nonconvex Structures,* Kluwer Academic Publishers, Boston, Dordrecht, London, to appear in 1996

[16] E.L. Lawler, D.E. Wood, Branch and bound methods: a survey, *Operation Research* 14(1966), 699-719.

[17] S.M. Lim, G.P. Papavassilopoulos, Numerical experience with parallel algorithms for solving the BMI problems, *Proc. of 13-th IFAC Congress*, San Fransisco, 1996, pp. 387-392.

[18] G.P. McCormic, Computability of global solutions to factorable nonconvex programs: part I-convex understimating problems, *Mathematical Programming* 10(1976), 147-175.

[19] G.P. McCormic, *Nonlinear programming: theory, algorithms and applications*, John Willey and Sons, 1982.

[20] A. Packard, K. Zhou, P. Pandey, J. Leonhardson, G. Balas, Optimal, constant I/O similarity scaling for full-information and state-feedback control problems, *Systems & Control Letters* 19(1992), 271-280.

[21] A. Packard, J. Doyle, The complex structured singular value, *Automatica* 29(1993), 71-109.

[22] E. Polak, Y. Wardi, A nondifferentiable optimization algorithm for the design of control systems subject to singular value inequalities over a frequency range, *Automatica* 18(1982), 267-283.

[23] R.T. Rockafellar, *Convex analysis*, Princeton University Press, 1970.

[24] M. Rotea, T. Iwasaki, An alternative to the D-K iteration?, *Proc. of the American Control Conference 1994*, pp. 53-57.

[25] M.G. Safonov, K.C. Goh, J.H. Ly, Control systems synthesis via bilinear matrix inequalities, *Proc. of the American Control Conference 1994*, pp. 45–49.

[26] H.D. Sherali, A. Alameddine, A new reformulation-linearization technique for bilinear programming problem, *J. of Global Optimization* 2(1992), 379-410.

[27] O. Slupphaug, On robust constrained nonlinear control and hybrid control: BMI and MPC-based state feedback schemes, it Ph.D. Thesis, Department of Engineering Cybernetics, Norwegian university of Science and Technology, Norway, October 1998.

[28] P.T. Thach, D.c sets, d.c. functions and nonlinear equations, *Mathematical Programming* 58(1993), 415-428.

[29] H.D. Tuan, P. Apkarian, Relaxation of parameterized LMIs with control applications, *International J. of Nonlinear Robust Controls* 9(1999), 59-84.

[30] H.D. Tuan, S. Hosoe, H. Tuy, D.C. optimization approach to robust controls: the optimal scaling value problem, *Proceedings of 1997 American Control Conference*, 350-355; Also to appear in *IEEE Trans. Automatic Control.*

[31] H.D. Tuan, Remarks on a global optimization algorithm for $H_\infty$ control, To appear in *IEEE Trans. Automatic Control.*

[32] H. Tuy, *Convex analysis and global optimization*, Kluwer Academic, 1998.

[33] H. Tuy, D.C. Optimization: Theory, Methods and Algorithms, in *Handbook of Global Optimization*, R. Horst and P. Pardalos eds, Kluwer Academic Publishers, 1995, pp. 149-216.

[34] H. Tuy, Canonical D.C. Programming: Outer Approximation Methods Revisited, *Operations Research Letters* 18(1995), 99-106.

[35] Y. Yamada, S. Hara, Global optimization for $H_\infty$ control with constant diagonal scaling, *IEEE Trans. on Automatic Controls* 43(1998), 191-203.