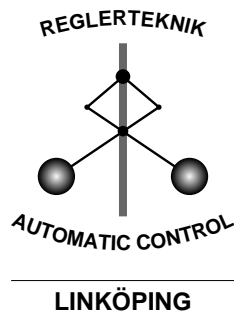


# A cutting plane method for solving KYP-SDPs

Ragnar Wallin, Chung-Yao Kao, Anders Hansson

Division of Automatic Control  
Department of Electrical Engineering  
Linköpings universitet, SE-581 83 Linköping, Sweden  
WWW: <http://www.control.isy.liu.se>  
E-mail: [ragnarw@isy.liu.se](mailto:ragnarw@isy.liu.se), [cykao@ee.unimelb.edu.au](mailto:cykao@ee.unimelb.edu.au),  
[hansson@isy.liu.se](mailto:hansson@isy.liu.se)

31st October 2006




Report no.: LiTH-ISY-R-2731

Technical reports from the Control & Communication group in Linköping are available at <http://www.control.isy.liu.se/publications>.

## **Abstract**

Semidefinite programs originating from the Kalman-Yakubovich-Popov lemma are convex optimization problems and there exist polynomial time algorithms that solve them. However, the number of variables is often very large making the computational time extremely long. Algorithms more efficient than general purpose solvers are thus needed. To this end structure exploiting algorithms has been proposed, based on the dual formulation. In this paper a cutting plane algorithm is proposed. It is shown that it in certain cases outperforms both general purpose solvers and structure exploiting solvers.

**Keywords:** Semidefinite programming, Kalman-Yakubovich-Popov lemma, Cutting plane method, Stability analysis.

	<b>Avdelning, Institution</b> Division, Department  Division of Automatic Control Department of Electrical Engineering	<b>Datum</b> Date  2006-10-31
	<b>Språk</b> Language  <input type="checkbox"/> Svenska/Swedish <input checked="" type="checkbox"/> Engelska/English  <input type="checkbox"/> _____	<b>Rapporttyp</b> Report category  <input type="checkbox"/> Licentiatavhandling <input type="checkbox"/> Examensarbete <input type="checkbox"/> C-uppsats <input type="checkbox"/> D-uppsats <input checked="" type="checkbox"/> Övrig rapport <input type="checkbox"/> _____
<b>URL för elektronisk version</b>  <a href="http://www.control.isy.liu.se">http://www.control.isy.liu.se</a>		LiTH-ISY-R-2731
<b>Titel</b> A cutting plane method for solving KYP-SDPs Title		
<b>Författare</b> Ragnar Wallin, Chung-Yao Kao, Anders Hansson Author		
<b>Sammanfattning</b> Abstract  <p>Semidefinite programs originating from the Kalman-Yakubovich-Popov lemma are convex optimization problems and there exist polynomial time algorithms that solve them. However, the number of variables is often very large making the computational time extremely long. Algorithms more efficient than general purpose solvers are thus needed. To this end structure exploiting algorithms has been proposed, based on the dual formulation. In this paper a cutting plane algorithm is proposed. It is shown that it in certain cases outperforms both general purpose solvers and structure exploiting solvers.</p>		
<b>Nyckelord</b> Keywords      Semidefinite programming, Kalman-Yakubovich-Popov lemma, Cutting plane method, Stability analysis.		

# A cutting plane method for solving KYP-SDPs

Ragnar Wallin<sup>a</sup>, Chung-Yao Kao<sup>b</sup>, Anders Hansson<sup>a</sup>

<sup>a</sup>*Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden*

<sup>b</sup>*Department of Electrical and Electronic Engineering, The University of Melbourne, Victoria 3010, Australia*

---

## Abstract

Semidefinite programs originating from the Kalman-Yakubovich-Popov lemma are convex optimization problems and there exist polynomial time algorithms that solve them. However, the number of variables is often very large making the computational time extremely long. Algorithms more efficient than general purpose solvers are thus needed. To this end structure exploiting algorithms has been proposed, based on the dual formulation. In this paper a cutting plane algorithm is proposed. It is shown that it in certain cases outperforms both general purpose solvers and structure exploiting solvers.

*Key words:* Semidefinite programming, Kalman-Yakubovich-Popov lemma, Cutting plane method, Stability analysis.

---

## 1 Introduction

Semidefinite programs derived from the Kalman-Yakubovich-Popov lemma (KYP-SDPs) are convex optimization problems and have the following form

$$\begin{aligned} v_{\text{opt}} = \inf_{x,P} & c^T x + \text{Tr}(CP) \\ \text{s.t.} & \begin{bmatrix} A^T P + P A & P B \\ B^T P & 0 \end{bmatrix} + M_0 + \sum_{i=1}^p x_i M_i > 0 \end{aligned} \quad (1)$$

where  $A \in \mathbb{R}^{n \times n}$ ,  $B \in \mathbb{R}^{n \times m}$ ,  $P \in \mathbb{S}^n$  and  $M_k \in \mathbb{S}^{n+m}$ ,  $i = 0, 1, \dots, p$ . Notation  $\text{Tr}(\cdot)$  denotes the trace of the given matrix and  $\mathbb{S}^n$  is the set of symmetric  $n \times n$  matrices. We assume that the pair  $(A, B)$  is stabilizable. Without lack of generality we also assume that  $A$  has no eigenvalues on the imaginary axis. The latter assumption is only made to simplify the presentation of the algorithm. Due to the stabilizability assumption eigenvalues on the imaginary axis can be relocated by feedback; details are given in Appendix B. The matrix  $C$  is assumed to be negative semidefinite, which is the usual case in applications. Moreover, we assume that the optimal value  $v_{\text{opt}}$  is finite. In case there are several matrix

variables  $P_j$  it is possible to generalize the results to

$$\begin{aligned} \inf_{x,P} & c^T x + \sum_{j=1}^N \text{Tr}(C_j P_j) \\ \text{s.t.} & \begin{bmatrix} A_j^T P_j + P_j A_j & P_j B_j \\ B_j^T P_j & 0 \end{bmatrix} + M_{0,j} + \sum_{i=1}^p x_i M_{i,j} > 0, \\ & j = 1, \dots, N \end{aligned}$$

but for simplicity we only treat the case with one variable. A generalization is straightforward. The only restriction is that the matrix  $P_j$  must only appear in one constraint. Note that a standard Linear Matrix Inequality (LMI)

$$F = F_0 + \sum_{i=1}^p x_i F_i > 0$$

is a special case of the constraint (1) with  $n = 0$ . Thus, in general, we can handle a mixture of KYP constraints and standard LMIs.

It may seem that KYP-SDPs are problems of a special form, but they appear in numerous applications in control and signal processing. A partial list of applications includes linear system design and analysis [5,25], robust control analysis using integral quadratic constraints [32,26,3], quadratic Lyapunov function search [6], and filter design [1].

The most common approach for solving SDPs is to use interior-point methods, e.g. [36,37]. Unfortunately the

---

*Email addresses:* ragnarw@isy.liu.se (Ragnar Wallin), cykao@ee.unimelb.edu.au (Chung-Yao Kao), hansson@isy.liu.se (Anders Hansson).

number of variables in the SDP often gets very large, making it hard or even impossible to solve with general purpose software. Under the circumstances that  $n$  is much larger than  $m$  and  $p$ , the computational time for each iteration in a general purpose interior-point method is proportional to  $n^6$  for the case of KYP-SDPs. Therefore interior-point algorithms utilizing structure have been developed, e.g. [43,1,44,21,11,40,24]. It has been shown that it is possible to reduce the computational time for each iteration to  $\mathcal{O}(n^3)$ . Also iterative methods have been employed for computing the search directions, e.g. [22,23,44,12]. For iterative methods it is not possible to provide any theoretical results on the improvement in computational complexity. The results depend on the numerical accuracy asked for, and what type of pre-conditioner is used in the iterative solvers.

In many applications the variable  $P$  is of little intrinsic value, since it is often only introduced to convert a semi-infinite frequency domain inequality to a finite dimensional LMI as in (1). This equivalence is usually referred to as the KYP lemma. Because of this, several researchers have proposed alternatives to standard interior-point methods for solving KYP-SDPs. These methods include cutting-plane methods and outer approximation methods, e.g. [34,41,21,28], and interior-point methods based on alternative barrier-functions for the frequency domain constraint, [27]. It was shown that the per-iteration computation complexity of these algorithm is  $\mathcal{O}(n^3)$ , which provides the main saving of the computational time.

In this paper we extend the work on cutting-plane methods (CPM) to also cover the case when the variable  $P$  is of intrinsic value. We will later on see an application where this is the case. An analytical center cutting plane (ACCP) algorithm is implemented to study the efficiency of applying CPM to solve KYP-SDPs. We will show that the per-iteration computational complexity of such an algorithm is  $\mathcal{O}(n^3)$ . Furthermore, it is known that the number of iterations for ACCP algorithm to converge to an  $\epsilon$ -accurate suboptimal solution is  $\mathcal{O}(p^2/\epsilon^2)$ , where  $p$  is the number of decision variables to be optimized [15]. The overall complexity in theory is therefore  $\mathcal{O}(n^3 p^2/\epsilon^2)$ . The worst-case complexity makes ACCP algorithm seemingly a rather slow algorithm, especially when a suboptimal solution with good accuracy is required. However, the results of numerical experiments have shown that it seems like the computational complexity in practice is  $\mathcal{O}(n^3 p |\log \epsilon|)$ , [28]. We remark that our results are based on a reformulation of the optimization problem such that CPM is applied to a problem where only the  $x$ -variables are present. In case CPM would have been applied directly to the KYP-SDP, the complexity would have been  $\mathcal{O}(n^6 p^2/\epsilon^2)$ .

The remaining part of the paper is organized as follows. In Section 2 we review the applications of KYP-SDPs to the linear quadratic regulator and an extension. In Sec-

tion 3 we rewrite the KYP-SDP to fit the cutting plane framework. We also present some results on how solutions to algebraic Riccati equations can be used to solve certain subproblems for fixed value of  $x$ . In Section 4 we introduce the cutting plane method and discuss its computational complexity with respect to the number of iterations to convergence. In Section 5 we show how to generate feasibility cuts and value cuts for KYP-SDPs. In Section 6 we discuss the computational complexity of the cutting plane method per iteration. We also make a comparison of its overall complexity with interior-point methods. In Section 7 implementation issues are addressed, in Section 8 numerical experimental results are presented, and finally in Section 9 some conclusions are made.

## 2 Linear Quadratic Regulators

As mentioned in the introduction there are numerous applications of KYP-SDPs. In this section we will focus on explaining the application to linear quadratic regulators.

Consider the continuous-time dynamical system model

$$\dot{\xi} = A\xi + Bu \quad (2)$$

with initial value  $\xi(0) = \xi_0$ ,  $A \in \mathbb{R}^{n \times n}$ ,  $B \in \mathbb{R}^{n \times m}$ ,  $\xi(t) \in \mathbb{R}^n$ , and  $u(t) \in \mathbb{R}^m$ . Assume that  $(A, B)$  is controllable.

### 2.1 Quadratic Objective

Define the cost index

$$J = \int_0^\infty \begin{bmatrix} \xi \\ u \end{bmatrix}^T M \begin{bmatrix} \xi \\ u \end{bmatrix} dt \quad (3)$$

where

$$M = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \in \mathbb{S}^n$$

with  $R > 0$ . It is well-known (see, e.g., [45]), that the infimal value of  $J$  with respect to  $u(\cdot)$  subject to (2) and such that  $\lim_{t \rightarrow \infty} \xi(t) = 0$  is, whenever it exists, given by  $\xi_0^T P \xi_0$ , where  $P \in \mathbb{S}^n$  solves the KYP-SDP

$$\begin{aligned} & \sup \xi_0^T P \xi_0 \\ \text{s. t. } & \begin{bmatrix} A^T P + P A & P B \\ B^T P & 0 \end{bmatrix} + M > 0. \end{aligned} \quad (4)$$

The optimal  $u(\cdot)$  is given as a state feedback  $u(t) = -R^{-1}(PB + S)^T \xi(t)$ . Here we see an application where the variable  $P$  is of intrinsic interest and appears in the

objective. For this special case, as will be discussed in detail later on, the optimal  $P$  can be found as the maximal solution of the algebraic Riccati equation

$$A^T P + PA + Q - (PB + S)^T R^{-1} (PB + S) = 0,$$

Very efficient methods are available for computing this solution, [30]. The computational complexity of these methods is in the order of  $n^3$ . However, slight generalizations of the above problem formulation require the solution of general KYP-SDPs. An example is given next.

## 2.2 Quadratic constraints

Define the cost indices

$$J_i = \int_0^\infty \begin{bmatrix} \xi \\ u \end{bmatrix}^T M_i \begin{bmatrix} \xi \\ u \end{bmatrix} dt, \quad i = 0, \dots, p \quad (5)$$

where  $M_i \in \mathbb{S}^n$ . Now consider the constrained optimization problem

$$\begin{aligned} & \inf J_0 \\ & \text{s. t. } J_i \leq c_i, \quad i = 1, \dots, p \\ & \quad (2) \text{ and } \lim_{t \rightarrow \infty} \xi(t) = 0 \end{aligned} \quad (6)$$

with respect to  $u(\cdot)$ . The optimal value to this problem, whenever it exists, is given by  $\xi_0^T P \xi_0$ , where  $P$  solves the KYP-SDP

$$\begin{aligned} & \sup \xi_0^T P \xi_0 - c^T x \\ & \text{s. t. } \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} + M_0 + \sum_{i=1}^p x_i M_i > 0 \\ & \quad x_i > 0, \quad i = 1, \dots, p \end{aligned} \quad (7)$$

(see [6]). Here we see an application where both the variables  $P$  and  $x$  are of intrinsic interest. Moreover, we have multiple constraints, some of which only involve  $x$ .

## 2.3 Riccati equations

We will later on see that for general KYP-SDPs the cutting plane method presented in this paper compute the solution by solving a sequence of algebraic Riccati equations. To this end we need the following result proven in Appendix A. Let us introduce the quadratic matrix

$$\mathcal{Q}(P) = A^T P + PA + Q - (PB + S)R^{-1}(PB + S)^T$$

Assume that the pair  $(A, B)$  is stabilizable, that there is a  $P \in \mathbb{S}^n$  such that  $\mathcal{Q}(P) > 0$  and that  $R > 0$ . Then there exists a unique solution  $P_r$  to the Riccati equation

$$\mathcal{Q}(P) = 0, \quad (8)$$

such that  $A_r = A - BR^{-1}(P_r B + S)^T$  is Hurwitz. Furthermore,  $P_r$  is the maximal solution, i.e.

$$P_r > P, \quad \forall P \text{ such that } \mathcal{Q}(P) > 0$$

It is not a good idea to use the iterative method in Appendix A to compute the solution. Instead, methods based on the Hamiltonian matrix

$$H = \begin{bmatrix} A - BR^{-1}S^T & BR^{-1}B^T \\ Q - SR^{-1}S^T & -(A - BR^{-1}S^T)^T \end{bmatrix} \quad (9)$$

are preferred, [30]. These methods compute the solution based on the real ordered Schur form of the Hamiltonian. It will be shown that the existence of  $P \in \mathbb{S}^n$  such that  $\mathcal{Q}(P) > 0$  holds is equivalent to the Hamiltonian having no eigenvalues on the imaginary axis. This condition is trivially checked once the Schur form has been computed.

## 3 Preliminaries

In this section we will reformulate the optimization problem (1) to fit in the framework of the cutting plane method. First we will make a separation of the variables  $x$  and  $P$  in such a way that we define an optimization problem in terms of only  $x$ , where the optimization over  $P$  is made implicit. We show that the optimization problem in  $x$  is convex and that its objective function is continuously differentiable with respect to  $x$ . We also show that the implicit optimization over  $P$  for fixed  $x$  can be carried out by solving an algebraic Riccati equation. Also, the dual of the implicit problem is discussed. These results are the key to efficiently compute value and feasibility cuts in the cutting plane method.

### 3.1 Separation of variables

Let us first define the operator  $\mathcal{F} : \mathbb{S}^n \rightarrow \mathbb{S}^{n+m}$  as

$$\mathcal{F}(P) = \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix}$$

the operator  $\mathcal{G} : \mathbb{R}^p \rightarrow \mathbb{S}^{n+m}$  and its partitioning as

$$\begin{aligned} \mathcal{G}(x) &= M_0 + \sum_{i=1}^p x_i M_i = \begin{bmatrix} Q(x) & S(x) \\ S^T(x) & R(x) \end{bmatrix} \\ &= \begin{bmatrix} Q_0 & S_0 \\ S_0^T & R_0 \end{bmatrix} + \sum_{i=1}^p x_i \begin{bmatrix} Q_i & S_i \\ S_i^T & R_i \end{bmatrix} \end{aligned}$$

the adjoint operator  $\mathcal{F}^* : \mathbb{S}^{n+m} \rightarrow \mathbb{S}^n$  as

$$\mathcal{F}^*(Z) = AZ_{11} + Z_{11}A^T + BZ_{12}^T + Z_{12}B^T$$

where  $Z$  is partitioned as

$$Z = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{12}^T & Z_{22} \end{bmatrix}$$

the set  $\mathcal{X}$  as

$$\mathcal{X} = \{x : \exists P \in \mathbb{S}^n : \mathcal{F}(P) + \mathcal{G}(x) > 0\}$$

and the set  $\mathcal{Z}$  as

$$\mathcal{Z} = \{Z : \mathcal{F}^*(Z) = C, Z \geq 0\}$$

The set  $\mathcal{X}$  is the set of all  $x$  for which there exists a feasible solution to (1). If we then separate the optimization in  $x$  and  $P$  and for all  $x \in \mathcal{X}$  define the function

$$h(x) = c^T x + \inf_P \text{Tr}(CP) \quad (10)$$

s.t.  $\mathcal{F}(P) + \mathcal{G}(x) > 0$

we can write (1) as

$$v_p = \inf_{x \in \mathcal{X}} h(x) \quad (11)$$

The optimal solution to the problems (1) and (11) are identical.

We will frequently need to check if a given  $x$  is an element of  $\mathcal{X}$  or not. This is equivalent to investigating if there exists a  $P \in \mathbb{S}^n$  such that  $\mathcal{F}(P) + \mathcal{G}(x) > 0$ . By the Schur complement formula this is equivalent to

$$R(x) > 0 \quad (12)$$

$$\mathcal{Q}(P, x) > 0 \quad (13)$$

where

$$\mathcal{Q}(P, x) = A^T P + PA + Q(x) - (PB + S(x))R^{-1}(x)(PB + S(x))^T$$

If (12) holds and  $A$  has no eigenvalues on the imaginary axis, then (13) holds if and only if the associated Hamiltonian matrix similarly defined as in (9) has no eigenvalues on the imaginary axis, [2]. In case the Hamiltonian has an eigenvalue  $\lambda$  on the imaginary axis it follows from Lemma 1 and Proposition 12 in [2] that

$$G(x, \lambda) = \begin{bmatrix} (\lambda I - A)^{-1} B \\ I \end{bmatrix}^* \mathcal{G}(x) \begin{bmatrix} (\lambda I - A)^{-1} B \\ I \end{bmatrix} \quad (14)$$

is not positive definite.

### 3.2 Riccati reformulation

We will now for  $x \in \mathcal{X}$  show that

$$h(x) = c^T x + \text{Tr}(CP_r) \quad (15)$$

where  $P_r$  is the maximal solution to the Riccati equation  $\mathcal{Q}(P, x) = 0$ . To see this, first notice that  $R(x) > 0$  since  $x \in \mathcal{X}$ . Hence, there exists, as previously shown, a unique maximal solution  $P_r$  to the Riccati equation. For this solution it holds that

$$P_r > P \quad \forall P \text{ such that } \mathcal{Q}(P, x) > 0$$

Thus,

$$\text{Tr}(C(P_r - P)) = -\text{Tr}(C(P - P_r)) \leq 0$$

as  $C \leq 0$  and  $P - P_r < 0$  and the product of two negative semidefinite matrices has nonnegative eigenvalues. This implies that  $\text{Tr}(CP_r) \leq \text{Tr}(CP)$  for any  $P$  such that  $\mathcal{Q}(P) > 0$ .

### 3.3 Dual reformulation

For a given  $x \in \mathcal{X}$  we can in (10) replace

$$\inf_P \text{Tr}(CP) \quad (16)$$

s.t.  $\mathcal{F}(P) + \mathcal{G}(x) > 0$

with its dual

$$\sup_{Z \in \mathcal{Z}} \text{Tr}(-Z\mathcal{G}(x)) \quad (17)$$

which will yield the same value as strong duality holds because  $v_{\text{opt}}$  is finite by assumption and (1) implies strict feasibility [4]. Hence,  $h(x)$  can also be written as

$$h(x) = c^T x - \inf_{Z \in \mathcal{Z}} \text{Tr}(Z\mathcal{G}(x)) \quad (18)$$

We now show that an optimal solution to (17) is

$$\bar{Z} = \begin{bmatrix} \bar{Z}_{11} & \bar{Z}_{12} \\ \bar{Z}_{12}^T & \bar{Z}_{22} \end{bmatrix} = \begin{bmatrix} I \\ F_r \end{bmatrix} \bar{Z}_{11} \begin{bmatrix} I & F_r^T \end{bmatrix}$$

where  $\bar{Z}_{11}$  solves the Lyapunov equation

$$A_r \bar{Z}_{11} + \bar{Z}_{11} A_r^T = C \leq 0 \quad (19)$$

Here the matrices  $F_r$  and  $A_r$  are defined as

$$F_r = -R^{-1}(x^j)(P_r(x^j)B + S(x^j))^T$$

$$A_r = A + BF_r$$

where  $P_r$  is the maximal solution to the Riccati equation. To show that  $\bar{Z}$  is optimal we first notice that the

maximal solution is stabilizing, i.e.  $A_r$  is Hurwitz. Since  $C$  is negative semidefinite this implies that  $\bar{Z}_{11}$  is positive semidefinite. That  $\bar{Z}_{11}$  is positive semidefinite in turn means that  $\bar{Z} \geq 0$  by construction. We also have that

$$\begin{aligned} A\bar{Z}_{11} + \bar{Z}_{11}A^T + B\bar{Z}_{12}^T + \bar{Z}_{12}B^T \\ = A_r\bar{Z}_{11} + \bar{Z}_{11}A_r^T = C \end{aligned}$$

Hence,  $\bar{Z}$  is in  $\mathcal{Z}$ . As we have assumed that the primal is bounded from below and strictly feasible a dual feasible  $\bar{Z}$  is optimal if and only if complementary slackness holds [4]. This is true as

$$\begin{aligned} \bar{Z} \begin{bmatrix} A^T P_r + P_r A + Q(x^j) & P_r B + S(x^j) \\ B^T P_r + S(x^j)^T & R(x^j) \end{bmatrix} \\ = \begin{bmatrix} I \\ F_r \end{bmatrix} \bar{Z}_{11} \mathcal{M} = 0 \end{aligned}$$

To see the last equality, note that

$$\begin{aligned} \mathcal{M} &= \begin{bmatrix} I & F_r^T \end{bmatrix} \begin{bmatrix} A^T P_r + P_r A + Q(x^j) & P_r B + S(x^j) \\ B^T P_r + S(x^j)^T & R(x^j) \end{bmatrix} \\ &= \begin{bmatrix} A^T P_r + P_r A + Q(x^j) + F_r^T (B^T P_r + S(x^j)^T) \\ P_r B + S(x^j) + F_r^T R(x^j) \end{bmatrix} \\ &= \begin{bmatrix} \mathcal{Q}(P_r, x^j) \\ P_r B + S(x^j) - P_r B - S(x^j) \end{bmatrix} = 0 \end{aligned}$$

The last equality follows from the fact that  $P_r$  satisfies the Riccati equation  $\mathcal{Q}(P_r, x^j) = 0$ .

### 3.4 Convexity and differentiability of $h(x)$

We now show that the function  $h(x)$  is convex and continuously differentiable on  $\mathcal{X}$ . To see that the function  $h(x)$  is convex we refer to equation (18) in Section 3.3. Expressed in this form,  $h(x)$  can be seen as a pointwise supremum over a set of convex functions and hence it is convex [7, page 81]. To see that  $h(x)$  is also continuously differentiable, notice that (10) can be rewritten as in (15). Then the following properties hold:

- (1) For any  $x^j \in \mathcal{X}$  it is possible to find a  $P_r \in \mathbb{S}^n$  such that  $\mathcal{Q}(P_r, x^j) = 0$ .
- (2)  $\mathcal{Q}(P, x)$  is continuously differentiable with respect to  $x \in \mathcal{X}$  and  $P \in \mathbb{S}^n$ .

If we in addition have that

$$\begin{aligned} \frac{\partial \text{vec}(\mathcal{Q}(P_r, x^j))}{\partial \text{vec}(P)^T} \text{ is invertible } \forall x^j \in \mathcal{X} \text{ and } P_r \\ \text{such that } \mathcal{Q}(P_r, x^j) = 0 \end{aligned}$$

then, according to the implicit function theorem,  $\mathcal{Q}(P_r, x^j) = 0$  defines a function  $P(x)$  in a neighbourhood of  $x^j \in \mathcal{X}$  which is continuously differentiable. Hence, let us investigate if  $\frac{\partial \text{vec} \mathcal{Q}(P_r, x^j)}{\partial \text{vec} P^T}$  is invertible. If we differentiate  $\mathcal{Q}(P, x)$  with respect to  $P$ , we obtain

$$\begin{aligned} \frac{\partial \text{vec}(\mathcal{Q}(P_r, x^j))}{\partial \text{vec}(P)^T} &= I_n \otimes A^T + A^T \otimes I_n \\ &\quad - I_n \otimes P_r B R^{-1}(x^j) B^T \\ &\quad - P_r B R^{-1}(x^j) B^T \otimes I_n \\ &\quad + I_n \otimes S(x^j) R^{-1}(x^j) B^T \\ &\quad - S(x^j) R^{-1}(x^j) B^T \otimes I_n \\ &= I_n \otimes \left( A - B R^{-1}(x^j) (P_r B + S(x^j))^T \right)^T \\ &\quad + \left( A - B R^{-1}(x^j) (P_r B + S(x^j))^T \right)^T \otimes I_n \\ &= I_n \otimes A_r^T + A_r^T \otimes I_n \end{aligned} \tag{20}$$

The matrix in (20) is the matrix we get if we vectorize a Lyapunov operator  $\mathcal{L}(A_r, P_r) = A_r^T P_r + P_r A_r$ , since

$$\text{vec}(\mathcal{L}(A_r, P)) = (I_n \otimes A_r^T + A_r^T \otimes I_n) \text{vec}(P)$$

The Lyapunov operator is invertible if  $A_r$  is Hurwitz. The matrix  $A_r$  is Hurwitz as  $P_r$  is a stabilizing solution to the Riccati equation. Thus,  $\frac{\partial P}{\partial x}$  exists and for all  $x \in \mathcal{X}$

$$\frac{\partial h(x^j)}{\partial x_i} = c_i + \text{Tr} \left( C \frac{\partial P(x^j)}{\partial x^j} \right), \quad i = 1, 2, \dots, p$$

and furthermore these derivatives are continuous.

## 4 Cutting plane method

Cutting plane methods can be used for solving convex optimization problems and were independently introduced in [8] and [29]. The methods rely on polyhedron approximations of convex sets. An important requirement on the optimization problem is that it is computationally cheap to decide if a so-called trial point is feasible or not. Furthermore, there must be an efficient way to produce cuts that generate an increasingly refined polyhedron approximation of the solution set. The KYP-SDP has those qualities and cutting plane methods are thus well suited to solve the problem.

### 4.1 Derivation of algorithm

Let us consider the the problem

$$\inf_{x \in \mathcal{X}} h(x) \tag{21}$$



where  $h(x)$  is a real-valued, continuously differentiable, convex function and  $\mathcal{X}$  is a convex set. Assume that we have a set of feasible points and have computed the function values and the gradients at those points

$$h(x^1), \dots, h(x^k), \quad \nabla h(x^1), \dots, \nabla h(x^k)$$

Affine lower bounds on  $h(x)$  are

$$h(x) \geq h(x^j) + \nabla^T h(x^j)(x - x^j) \quad \forall x \in \mathcal{X}, \quad 1 \leq j \leq k$$

and hence

$$h(x) \geq h_k^{\text{lb}}(x) := \max_{1 \leq j \leq k} (h(x^j) + \nabla^T h(x^j)(x - x^j))$$

The function  $h_k^{\text{lb}}(x)$  is piecewise linear, convex and for all  $x \in \mathcal{X}$  less than or equal to  $h(x)$ . It follows that

$$h(x_{\text{opt}}) \geq L_k := \inf_{x \in \mathcal{X}} h_k^{\text{lb}}(x)$$

where  $h(x_{\text{opt}})$  is the optimal value of (21). The minimization problem on the right-hand side can be rewritten using an epigraph formulation

$$\begin{aligned} L_k = \inf_{x \in \mathcal{X}, L} L \quad \text{subj. to} \\ L \geq h(x^j) + \nabla^T h(x^j)(x - x^j), \quad 1 \leq j \leq k \end{aligned} \quad (22)$$

We call the constraints (22) value cuts as all  $x$  with a lower value of the objective function  $h(x)$  than  $h(x^j)$  lie in the halfspace  $\{x | \nabla^T h(x^j)(x - x^j) < 0\}$ . The output of the program is a lower bound on  $h(x_{\text{opt}})$  and a feasible point. Having this feasible point we can compute a new function value, a new gradient and add a new value cut. However, we want to solve an even simpler problem. Instead of optimizing over  $x \in \mathcal{X}$  we optimize over  $x$  lying in a convex outer approximation of  $\mathcal{X}$ . This will also yield a lower bound on  $h(x)$ . This bound will be possibly lower as we optimize over a larger set. We will use a polyhedral outer approximation of  $\mathcal{X}$ . The  $k$ th relaxed optimization problem is then

$$\begin{aligned} L_k = \inf_{x, L} L \\ \text{s.t. } L \geq h(x^j) + \nabla^T h(x^j)(x - x^j), \quad j \in V_k \\ 0 > a_j^T(x - x^j), \quad j \in F_k \end{aligned} \quad (23)$$

where  $V_k$  and  $F_k$  are mutually exclusive and  $V_k \cup F_k = \{1, 2, \dots, N_k\}$ . The last constraints in (23) are called feasibility cuts and describe the polyhedral outer approximation of  $\mathcal{X}$ . The output of this linear program is a new trial point  $x^j$ , feasible or infeasible, and a lower bound on  $h(x_{\text{opt}})$ . If the new trial point is feasible we add a value cut to the program and if we get an infeasible point in the outer approximation we can improve the outer approximation by adding a feasibility cut. A feasibility cut

at  $x^j \notin \mathcal{X}$  is a linear constraint such that every feasible point lies in the halfspace

$$\{x | a^T(x - x^j) + \gamma \leq 0\}$$

If  $\gamma > 0$  the cut is deep and if  $\gamma = 0$  the cut passes through  $x^j$ . Observe that all the constraints in (23) are linear and describe a polyhedron  $\mathcal{P}_k$ . Thus, if we define  $y = \begin{bmatrix} L & x^T \end{bmatrix}^T$ , the optimization problem can be written as

$$L_k = \inf_y y_1, \quad \text{subj. to } y \in \mathcal{P}_k \quad (24)$$

where  $\mathcal{P}_k = \{y | d_j - c_j^T y \geq 0, \quad j = 1, \dots, N_k\}$ . A generic cutting plane algorithm is described by

• **Initialize**

- (1) Get an initial upper bound  $v_u$  and a lower bound  $v_l$  on the optimal value  $h(x_{\text{opt}})$ .
- (2) Get an initial relaxed optimization problem (24).

• **Repeat**

- (1) Get a trial point  $x^j$  and a corresponding lower bound  $v_l^+$ . The lower bound is the optimal value of the relaxed problem.
- (2) If  $x^j$  infeasible, then add a feasibility cut to the polyhedron approximation.
- (3) If  $x^j$  feasible, then add a value cut to the polyhedron approximation and compute an upper bound  $v_u^+$ . The upper bound is  $h(x^j)$ .
- (4) Update the bounds
  - (a) If  $x^j$  feasible, then  $v_u = \min\{v_u, v_u^+\}$ .
  - (b)  $v_l = \max\{v_l, v_l^+\}$

• **Until**  $v_u - v_l < \epsilon$

#### 4.2 Analytical Center Cutting Plane Algorithm

In practice, the cutting plane method described in the previous section may converge fast to the optimal solution if the relaxed optimization problem is a good approximation of the original one. If this is not the case convergence properties can be very poor. It is well-known that the worst-case iteration count is  $\mathcal{O}(1/\epsilon^p)$  [9]; i.e., the algorithm converges to an  $\epsilon$ -accuracy solution in iterations, where  $p$  is the number of decision variables to be optimized. A remedy to this poor convergence property is to use a centering method where a certain kind of center of the polyhedral approximation,  $\mathcal{P}_k$ , is computed in each iteration to serve as a candidate for the optimal solution. Centering methods were first introduced in [31]. Many different centering algorithms exist like the largest inscribed sphere method [10], the volumetric method [38] and the analytic center cutting plane method [17], [13], [14]. To achieve good speed of convergence, one should choose a center which not only allows the algorithm to refine the polyhedral approximation significantly but also can be computed efficiently. Among all possible centers, the analytic center has proven to be

a good choice for that purpose. The concept of analytical center was introduced by [35], where its use in the cutting plane methods was also alluded. The analytical center cutting plane (ACCP) algorithm and its implementation were then proposed in [13] and [46]. Subsequently, the theory underlying the ACCP algorithm has been studied in depth. The estimates of complexity for the basic method and several of its enhancements are provided [15,47,16,18,20].

Let  $\mathcal{P}_k$  be the polyhedron approximation at the  $k^{\text{th}}$  iteration. The analytical center of  $\mathcal{P}_k$  is the unique minimizer of

$$\mathcal{L}_k(y) = -\sum_{j=1}^{N_k} \log(d_j - c_j^T y)$$

over the interior of  $\mathcal{P}_k$ .

Thus, instead of solving the problem (24) we solve the analytic center problem corresponding to (24)

$$\inf_x \mathcal{L}_k(x), \quad \text{subj. to} \quad (25)$$

$$x \in \text{Interior}(\mathcal{P}_k)$$

The analytic center problem can be solved efficiently with, for example, methods described in [48].

Regarding the complexity of ACCP algorithm, it was shown in [15] that, the basic setup of the algorithm – has a complexity estimation of  $\mathcal{O}^*(p^2/\epsilon^2)$  for the total number of iterations, where  $p$  is the number of decision variables, and the  $\mathcal{O}^*$  notation means that the lower order terms are ignored. In practice, it often occurs that the oracle in the cutting plane algorithm can generate more than one cutting hyperplane upon a single inquiry. In [19], the case where two central cuts are placed in each iteration was analyzed. The worst-case iteration count is also  $\mathcal{O}^*(p^2/\epsilon^2)$ . The case of multiple central cuts was analyzed in [47]. It is shown that the algorithm converges after  $\mathcal{O}^*(\kappa^2 p^2/\epsilon^2)$  cutting planes have been generated, where  $\kappa$  is the maximum number of cuts generated at any given iteration.

In practice, the ACCP algorithm usually terminates long before the number of iterations reach the worst-case estimation. Some experiments suggest that the  $\mathcal{O}^*(p^2/\epsilon^2)$  bound is a very conservative estimation, and the algorithm may converge as fast as  $\mathcal{O}(p|\log \epsilon|)$  [28].

## 5 Computation of feasibility and value cuts

### 5.1 Feasibility cuts

A feasibility cut should be computed if a trial point  $x^j$  is not an element of  $\mathcal{X}$ . As has been shown above this is the

case if  $x^j$  yields an  $R(x^j)$  which is not positive definite or in case  $R(x^j)$  is positive definite but the Hamiltonian matrix (9) has at least one eigenvalue,  $\lambda$ , on the imaginary axis, in which case  $G(x^j, \lambda)$  in (14) is not positive definite.

So whenever a feasibility cut is to be constructed it holds that an LMI  $F(x^j) = F_0 + \sum_{i=1}^p x_i F_i > 0$  is not satisfied. Then we can generate a cut as in [6, page 13]. There exists a nonzero  $v^j$  such that

$$(v^j)^T F(x^j) v^j = (v^j)^T (F_0 + \sum_{i=1}^p x_i F_i) v^j \leq 0$$

Define  $a$  by

$$a_i = -(v^j)^T F_i v^j, \quad i = 1, 2, \dots, p$$

Then for any  $x$  satisfying  $a^T(x - x^j) \geq 0$  we have  $(v^j)^T F(x) v^j \leq 0$  and hence every  $x$  in  $\mathcal{X}$  lies in the half-space  $\{x | a^T(x - x^j) < 0\}$  which defines the cut

$$a^T(x - x^j) < 0$$

### 5.2 Value cuts

To compute the value cut we need to compute  $\nabla h(x)$ . The components of  $\nabla h(x^j)$  are from (15)

$$\frac{\partial h}{\partial x_i} = c_i + \text{Tr} \left( C \frac{\partial P_r(x^j)}{\partial x_i} \right), \quad i = 1, 2, \dots, p$$

Tedious calculations show that the matrix  $\frac{\partial P_r(x^j)}{\partial x^j}$  satisfies the Lyapunov equation

$$A_r^T \frac{\partial P_r(x^j)}{\partial x_i} + \frac{\partial P_r(x^j)}{\partial x_i} A_r + Q_r = 0, \quad i = 1, 2, \dots, p$$

where

$$\begin{aligned} A_r &= A - BR^{-1}(x^j)(P_r(x^j)B + S(x^j))^T \\ Q_r &= Q_i - S_i R^{-1}(x^j)(P_r(x^j)B + S(x^j))^T \\ &\quad - (P_r(x^j)B + S(x^j))R^{-1}(x^j)S_i^T + (P_r(x^j)B \\ &\quad + S(x^j))R^{-1}(x^j)R_i R^{-1}(x^j)(P_r(x^j)B + S(x^j))^T \end{aligned}$$

and where  $P_r$  is the maximal solution to the algebraic Riccati equation. Hence,  $\nabla h(x^j)$  could be computed by solving one Riccati equation and  $p$  Lyapunov equations. However, there is a more efficient way to find  $\nabla h(x^j)$ . For all  $x \in \mathcal{X}$  and  $Z \in \mathcal{Z}$  it follows from (18) that

$$h(x) = c^T x - \inf_{Z \in \mathcal{Z}} \text{Tr}(Z\mathcal{G}(x)) \geq c^T x - \text{Tr}(Z\mathcal{G}(x)) \quad (26)$$

with equality only for  $Z = \bar{Z}$  where  $\bar{Z} \in \mathcal{Z}$  maximizes the righthand side of the inequality. For every other  $x \in \mathcal{X}$  and  $Z \in \mathcal{Z}$  inequality holds. Thus,  $\{y|y = c^T x - \text{Tr}(\mathcal{G}(x)\bar{Z})\}$  defines a supporting hyperplane to  $h(x)$ . As  $h(x)$  is continuously differentiable there is only one such hyperplane. Hence,

$$\frac{\partial h}{\partial x_i} = c_i - \text{Tr}(M_i \bar{Z})$$

Remember that  $\bar{Z}$  can be computed from the solution to the algebraic Riccati equation.

## 6 Remarks on Computational Complexity

In this section, we comment on the computational complexities of solving KYP-SDP (1) using cutting plane methods and the general-purpose SDP solvers. Arguments are given to explain why the cutting plane methods may perform better than a general-purpose SDP solver which solves (1) without exploiting the special structure of the problem. Recall that the dimension of matrices  $A$  and  $B$  are  $n \times n$  and  $n \times m$ , respectively, and the number of the decision variables in  $x$  is  $p$ . We assume that  $n$  is order-of-magnitude larger than  $m$  and  $p$ .

In a cutting plane algorithm, the two main tasks to be performed in each iteration are checking feasibility of a trial point and generating cutting hyperplanes. In the case of solving KYP-SDPs, the most expensive computational procedures involved are those to find eigenvalues and eigenvectors of the the Hamiltonian matrix (9), to find the stabilizing solution to the algebraic Riccati equation (8), and to find the solution to the Lyapunov equation (19). All of these requires  $\mathcal{O}(n^3)$  arithmetic operations. Hence, the per-iteration computational complexity is estimated to be  $\mathcal{O}(n^3)$ . As such, if a standard analytical center cutting plane algorithm is implemented to solve KYP-SDPs, the overall worst-case computational complexity is, according to the discussion in Section 4.2 and above,  $\mathcal{O}^*(p^2/\epsilon^2) \cdot \mathcal{O}(n^3)$ .

On the other hand, solving KYP-SDP (1) using the primal-dual interior point algorithms can be proven to converge in  $\mathcal{O}(\sqrt{n} \log \epsilon)$  Newton steps [33]. In each Newton step, without utilizing the special structure of the problem, the computational complexities of finding a Newton descent direction is proportional to  $n^6$ . Therefore, the overall computational complexity of solving KYP-SDPs using a general-purpose SDP solver is  $\mathcal{O}(\sqrt{n} \log \epsilon) \cdot \mathcal{O}(n^6)$ . Should the special structure of the KYP-SDP be utilized, the per-iteration complexity could be knocked down to  $\mathcal{O}(n^4)$  or  $\mathcal{O}(n^3)$ . See [43,1,44,21,11,40,24].

From the complexity analysis, we observe a fundamental difference between the cutting plane methods and the

interior point methods. In a cutting plane algorithm, the complexity of each iteration is low but the algorithm requires many iterations to converge. On the contrary, the interior point algorithm requires only a few iterations to converge but the computational complexity of each iteration is much higher if the structure of the KYP-SDP is not taken into account. If the saving in each iteration is significant enough, a cutting plane algorithm (such as the ACCP algorithm) would outperform a general-purpose SDP solver. We will see that this observation agrees with the numerical experiments presented in the next section.

Finally, we note that there are experimental evidences that in practice, the ACCP algorithm might converge must faster than  $\mathcal{O}^*(p^2/\epsilon^2) \cdot \mathcal{O}(n^3)$ . The complexity might be as good as  $\mathcal{O}(n^3 p |\log \epsilon|)$  [28].

## 7 Implementation Issues

In this section, we discuss a few implementation issues.

### 7.1 Bounds on $h(x_{opt})$ and a stopping criterion

Every feasible solution found give an upper bound on the optimal solution of (10). Hence the smallest upper bound is obtained by finding the minimum over the set of  $h$  values evaluated at the available feasible solutions.

Lower bounds of (10) can be found by solving (23). However, a more efficient way is to solve the dual of (23) which is also a linear program. The optimal objective of the dual is equal to the optimal objective of (23), and more importantly, any suboptimal feasible solution to the dual problem will provide a lower bound. Thus, we do not have to solve the dual problem exactly.

The algorithm should terminate when the difference between the upper bound and the lower bound is sufficiently small.

### 7.2 Checking feasibility and solving the Riccati equation

As was mentioned earlier, the first condition for a trial point  $x$  to be feasible to the KYP-SDP is that  $R(x) > 0$  and the second condition is that the Hamiltonian matrix (9) has no eigenvalues on the imaginary axis. Checking the second condition gets very ill-conditioned when  $R(x)$  is close to being singular. If this is the case, it is better to compute the generalized eigenvalues of a certain matrix pencil [39]. It turns out that the finite eigenvalues of this pencil are the same as the eigenvalues of the Hamiltonian matrix.

The matlab function CARE checks if  $R(x) > 0$ , for a given  $x$ , and also uses the matrix pencil if it is necessary from a numerical point of view. We have modified the

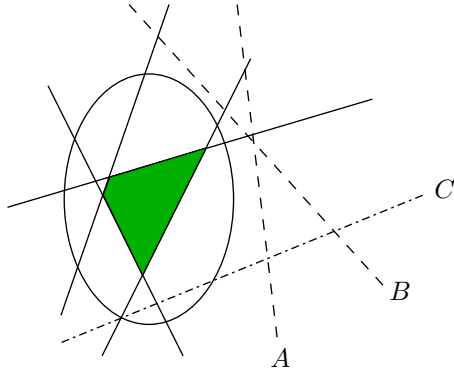


Fig. 1. The shaded area represents the polyhedron which contain the feasible set, which is contained in an ellipsoid. The redundant hyperplanes  $A$  and  $B$  do not intersect the ellipsoid, and hence will be dropped. The redundant hyperplanes  $C$ , on the other hand, is kept because it intersects the ellipsoid.

function CARE to deliver the eigenvalues of  $R(x)$  if it is not positive definite and the eigenvalues of the Hamiltonian if the ordered Schur form cannot be computed due to eigenvalues too close to the imaginary axis. Hence we can compute the feasibility cuts. The eigenvalues of the Hamiltonian with smallest magnitude of the real part are considered as being on the imaginary axis. Of course this results in conservatism as some feasible points are ruled out by the tighter condition. However, this conservatism has to be accepted as it is impossible to verify exactly if the eigenvalues are on the imaginary axis.

### 7.3 Constraint Dropping

One of the major issues with applying/implementing cutting plane methods is that the number of cutting planes (used to refine the approximation of the objective function and the feasible set) increases as the algorithm proceeds. As such, the complexity of finding trial points, which in the case of ACCP algorithm are the analytical centers of the polyhedral approximation of the feasible set, becomes higher and higher. The way to avoid such difficulty is to drop all redundant cutting planes. This, however, turns out to be a computationally costly operation; implementation of such operation is not practical.

In the case of ACCP algorithm, a heuristic mechanism can be adopted to drop redundant constraints, which does not incur much computation burden. The idea goes as follows. Given a polyhedron and its analytical center, or an sufficiently accurate approximation, one can identify an ellipsoid, centered at the analytical center (or the approximation), which contain the polyhedron. This ellipsoid comes with the procedure of finding the analytical center and hence there is no extra computational cost for identifying it. The set of hyperplanes which do not intersect the ellipsoid is obviously redundant and can be dropped. Note that to check whether a hyperplane intersects an ellipsoid can be done using an algebraic for-

mula, and hence incurs very little computational cost. This heuristic idea of course does not allow one to drop *all* redundant constraints, but indeed helps to speed up the ACCP algorithm. The idea is illustrated in Figure 1.

## 8 Numerical example

In this section we compare the analytic center cutting plane algorithm with other solvers. We solve the KYP-SDPs using the ACCP algorithm, SDPT3 [37], which is a general purpose solver, and KYPD [42,24] which is based on solving the dual problem to (1).

All computations were done in Matlab version 7.1.0.183 (R14) Service Pack 3 running under Linux. The computer used was a Dell Optiplex GX620 with an Intel Pentium 4 3.20 GHz processor and 2 GB RAM.

The generated optimization problems are on the form (1). The  $A$  and  $B$  matrices are randomly selected using the command `rss` in the control system toolbox. We choose to let the matrix  $B$  have five columns in all of the optimization problems. If the matrix  $A$  has eigenvalues with a real part larger than  $-0.1$  we add  $-0.1I$  to  $A$ . The matrices  $M_i$ ,  $i = 1, 2, \dots, p$  are generated as

$$M_i = \begin{bmatrix} 0_n & X \\ X^T & I_m \end{bmatrix}$$

where the elements of  $X \in \mathbb{R}^{n \times m}$  are drawn from a uniform distribution  $[0, 1]$ . The matrix  $M_0$  is chosen as

$$M_0 = N + N^T$$

where the elements of  $N \in \mathbb{R}^{n+m \times n+m}$  are drawn from a uniform distribution  $[0, 1]$ . The elements of the vector  $c$  are drawn from a uniform distribution  $[0, 1]$ . Finally to generate the matrix  $C$  we first compute

$$D = E + E^T$$

where the elements of  $E \in \mathbb{R}^{n+m \times n+m}$  are drawn from a uniform distribution  $[0, 1]$ . To make sure that  $C$  is negative definite we define it as

$$C = D - (\lambda_{\min}(D) - 0.5)I$$

where  $\lambda_{\min}(D)$  is the smallest eigenvalue of  $D$ .

SDPT3 can utilize sparsity. To introduce some sparsity we blockdiagonalize the matrix  $A$ . To this end, we first diagonalize the matrix  $A$  by the transformation

$$\bar{A} = T^{-1}AT$$

Then we transform the first part of the constraint in (1) as

$$\begin{bmatrix} \bar{A}^H \bar{P} + \bar{P} \bar{A} & \bar{P} \bar{B} \\ \bar{B}^H \bar{P} & 0 \end{bmatrix} = \begin{bmatrix} T & 0 \\ 0 & I \end{bmatrix}^H \begin{bmatrix} A^T P + P A & P B \\ B^T P & 0 \end{bmatrix} \begin{bmatrix} T & 0 \\ 0 & I \end{bmatrix}$$

with  $\bar{P} = T^H P T$  and  $\bar{B} = T^{-1} B$ . The matrices  $M_i$  are transformed analogously.

One negative aspect with the diagonalization is that the constraint will be complex valued if  $A$  has complex valued eigenvalues. There are two remedies to this dilemma. Either we can solve a real SDP involving LMIs with twice as many rows and columns as the original one [7] or we can transform the complex diagonal  $A$ -matrix into a real block diagonal one of the same size. We prefer the second alternative which is performed in the following way. Let us first assume that the eigenvalues are ordered on the diagonal. First we have all real eigenvalues and then the complex ones follow in complex conjugated pairs. To transform the  $A$ -matrix from being complex diagonal to being real block diagonal we only have to do a congruence transformation  $\bar{A} = V^H A V$ . The matrix  $V$  has ones on the diagonal for all rows with real eigenvalues and blocks

$$S = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -i \\ 1 & i \end{bmatrix}$$

on the diagonal for rows with complex conjugated eigenvalues. If we have a complex conjugated block in the  $A$ -matrix it will be transformed as

$$\begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix} \begin{bmatrix} \frac{a+ib}{2} & 0 \\ 0 & \frac{a-ib}{2} \end{bmatrix} \begin{bmatrix} 1 & -i \\ 1 & i \end{bmatrix} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \quad (27)$$

The congruence transformation will also result in a real  $B$ -matrix. Express the matrix  $P$  as  $\sum_{k=1}^{\frac{n(n+1)}{2}} \bar{x}_k E_k$ , where  $E_k$  is the standard basis for symmetric matrices. Then the constraint in (1) can be expressed as

$$\sum_{k=1}^{\frac{n(n+1)}{2}} \bar{x}_k F_k + \tilde{M}_0 + \sum_{i=1}^p x_i \tilde{M}_i > 0$$

where the matrices  $F_k$  are sparse. We solved those sparse optimization problems using the ACCP method (ACCPM) and SDPT3 providing the solver with the information that the matrices were sparse.

We also solved the dense optimization problems with ACCPM, SDPT3 and KYPD. KYPD used SDPT3 as an underlying solver, and exploits structure and lowrank properties following from the blockdiagonalization of the matrix  $A$ . For ACCPM it did not matter if the problems

were sparse or dense, and the computational times for SDPT3 when the matrices were sparse were only slightly shorter than the computational times when the matrices were dense. Thus we only show computational times for the dense problems. The termination criterium was that the relative error of the objective function should be less than  $10^{-6}$ .

SDPT3 started to swap memory when the size of the matrix  $A$  was greater than  $100 \times 100$  and KYPD started to swap memory when the matrix  $A$  was greater than  $150 \times 150$ . The ACCP algorithm can solve considerably larger problems without starting to swap memory. We solved a problem with  $A \in \mathbb{R}^{500 \times 500}$ ,  $B \in \mathbb{R}^{500 \times 5}$  and 5  $x$ -variables in approximately 80 minutes.

In Figure 2 we show computational times for different values of  $n$ , where  $A \in \mathbb{R}^{n \times n}$  when the number of  $x$ -variables is 5. The times shown are averages over forty randomly generated optimization problems of each size.

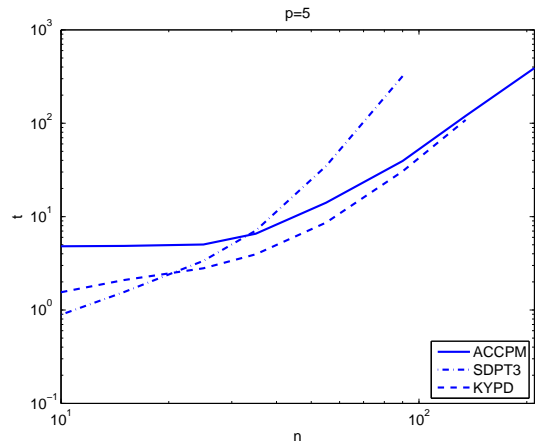


Fig. 2. Computational times when the number of  $x$ -variables is five

In figures 3 and 4 we show the results when the number of  $x$ -variables are 15 and 25, respectively.

It can be seen that both KYPD and ACCPM, for large values of  $n$ , have a slope of approximately 3 in all figures but the level of the curve for ACCPM increases when the number of  $x$ -variables increases. For all curves there is a point where ACCPM outperforms SDPT3 if the value of  $n$  is large enough. Also, ACCPM can handle larger problems than KYPD without swapping memory. KYPD performs at its best when the number of columns in  $B$  is much smaller than the number of columns in  $A$ , which is the case in the generated problems. If we had the same number of columns in  $A$  and  $B$  the performance of KYPD would be closer to the performance of SDPT3 and it would be preferable to use ACCPM.

The number of iterations it took to solve the optimization problems using ACCPM were approximately 40

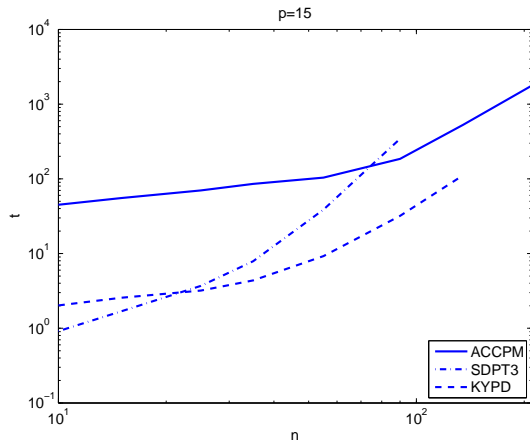


Fig. 3. Computational times when the number of  $x$ -variables is fifteen

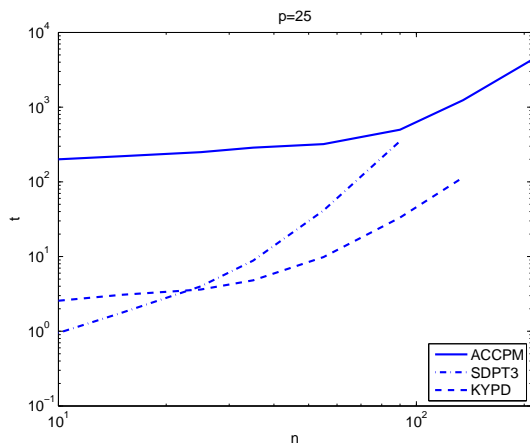


Fig. 4. Computational times when the number of  $x$ -variables is twentyfive

times the number of  $x$ -variables. About 23 % of the added cuts were value cuts and on average 0.7 cuts were dropped in each iteration. The above numbers are valid for all problems solved.

## 9 Conclusions

In this paper an analytic center cutting plane method for KYP-SDPs has been presented. It outperforms general purpose SDP solvers for large values of  $n$  and can handle larger problems than the structure exploiting algorithm KYPD. Problems up to the size of  $n = 500$  has been solved. Even larger problems may be possible to solve if time permits. For larger values of  $p$  and moderate values of  $n$  structure exploiting algorithms are to be preferred.

### Acknowledgements

The authors gratefully acknowledge financial support from The Swedish Research Council under contract No. 40469101.

## References

- [1] B. Alkire and L. Vandenberghe. Convex optimization problems involving finite autocorrelation sequences. *Mathematical Programming Series A*, 93:331–359, 2002.
- [2] V. Balakrishnan and L. Vandenberghe. Semidefinite programming duality and linear time-invariant systems. Technical Report TR-ECE 02-02, School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana, July 2002.
- [3] V. Balakrishnan and F. Wang. Efficient computation of a guaranteed lower bound on the robust stability margin for a class of uncertain systems. *IEEE Transactions on Automatic Control*, 44(11):2185–2190, November 1999.
- [4] A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization : analysis, algorithms, and engineering applications*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, USA, 2001.
- [5] S. Boyd and C. Barratt. *Linear controller design: Limits of performance*. Prentice Hall, 1991.
- [6] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, Philadelphia, Pennsylvania, USA, 1994.
- [7] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [8] E. W. Cheney and A. A. Goldstein. Newtons method for convex programming and Tchebycheff approximation. *Numerische Mathematik*, 1:253–268, 1959.
- [9] V. Demyanov and L. Vasilev. *Nondifferentiable Optimization*. Optimization Software. Springer Verlag, 1985.
- [10] J. Elzinga and T. G. Moore. A central cutting plane method for the convex programming problem. *Mathematical Programming*, 8:134–145, 1975.
- [11] Y. Genin, Y. Hachez, Yu. Nesterov, and P. Van Dooren. Optimization problems over positive pseudo-polynomial matrices. *SIAM Journal on Matrix Analysis and Applications*, 25(3):57–79, 2003.
- [12] J. Gillberg and A. Hansson. Polynomial complexity for a Nesterov-Todd potential-reduction method with inexact search directions. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, USA, 2003.
- [13] J.-L. Goffin, A. Haurie, and J.-P. Vial. Decomposition and nondifferential optimization with the projective algorithm. *Management Science*, 38(2):284–302, 1992.
- [14] J.-L. Goffin, A. Haurie, J.-P. Vial, and D. L. Zhu. Using central prices in the decomposition of linear programs. *European Journal of Operational Research*, 64:393–409, 1993.
- [15] J.-L. Goffin, Z.-Q. Luo, and Y. Ye. Complexity analysis of an interior cutting plane method for convex feasibility problems. *SIAM J. of Optimization*, 6(3):638–652, August 1996.
- [16] J.-L. Goffin and F. Sharifi Mokhtarian. Using the primal dual infeasible Newton method in the analytical method for problems defined by deep cutting planes. *Journal of Optimization Theory and Applications*, 101:35–58, 1999.
- [17] J.-L. Goffin and J.-P. Vial. Cutting planes and column generation techniques with the projective algorithm. *Journal of Optimization Theory and Applications*, 65:409–429, 1990.
- [18] J.-L. Goffin and J.-P. Vial. Shallow, deep and very deep cuts in the analytical center cutting plane method. *Mathematical Programming*, 84:89–103, 1999.

- [19] J.-L. Goffin and J.-P. Vial. A two-cuts approach in the analytical center cutting plane method. *Mathematical Methods of Operations Research*, 49(1):149–169, 1999.
- [20] J.-L. Goffin and J.-P. Vial. Multiple cuts in the analytical center cutting plane method. *SIAM Journal of Optimization*, 11(1):266–288, 2000.
- [21] Y. Hachez. *Convex optimization over nonnegative polynomials: structured algorithms and applications*. PhD thesis, Université catholique de Lovain, Louvain, Belgium, 2003.
- [22] A. Hansson and L. Vandenberghe. Efficient solution of linear matrix inequalities for integral quadratic constraints. In *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, 2000.
- [23] A. Hansson and L. Vandenberghe. A primal-dual potential reduction method for integral quadratic constraints. In *Proceedings of the American Control Conference*, pages 3013–3017, Arlington, Virginia, USA, 2001.
- [24] J. Harju, R. Wallin, and A. Hansson. Utilizing low rank properties when solving KYP-SDPs. In *Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego, California, USA, 2006. To be presented.
- [25] H. Hindi, B. Hassibi, and S. Boyd. Multiobjective  $H_2/H_\infty$ -optimal control via finite-dimensional Q-parabeterization and linear matrix inequalities. In *Proceedings of the American Control Conference*, volume 5, pages 3244–3249, Philadelphia, Pennsylvania, USA, 1998.
- [26] U. Jönsson. *Robustness analysis of uncertain and nonlinear systems*. PhD thesis, Lund Institute of Technology, Lund, Sweden, 1996.
- [27] C.-Y. Kao and A. Megretski. A new barrier function for IQC optimization problems. In *Proceedings of the American Control Conference*, volume 5, pages 4281–4286, Denver, Co, USA, June 2003.
- [28] C.-Y. Kao, A. Megretski, and U. T. Jönsson. Specialized fast algorithms for IQC feasibility and optimization problems. *Automatica*, 40(2):239–252, 2004.
- [29] J. E. Kelley. The cutting plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8:703–712, 1960.
- [30] A. J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Transactions on Automatic Control*, 24(6):913–921, 1979.
- [31] A. Y. Levin. On an algorithm for the minimization of convex functions over convex sets. *Soviet Mathematical Doklady*, 6:286–290, 1965.
- [32] A. Megretski and A. Rantzer. System analysis via integral quadratic constraints. *IEEE Transactions on Automatic Control*, 42(6):819–830, 1997.
- [33] Y. Nesterov and A. Nemirovski. *Interior Point Polynomial Methods in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, 1994.
- [34] P. Parrilo. Outer approximation algorithms for KYP-based LMIs. In *Proceedings of the American Control Conference*, volume 4, pages 3025–3028, Arlington, Virginia, USA, June 2001.
- [35] G. Sonnevand. New algorithms in convex programming based on a notion of ‘centre’ (for systems of analytical inequalities) and on rational extrapolation. In *Trends in Mathematical Optimization: Proceedings of the 4th French-German Conference on Optimization*, pages 311–327, Irsee West-Germany, 1986. K. H. Hoffmann, J. B. Hiriart-Urruty, C. Lemaréchal, and J. Zowe, eds.
- [36] J. F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11-12:625–653, 1999.
- [37] K. C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3—a Matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.
- [38] P. Vaidya. A new algorithm for minimizing convex functions over convex sets. *Mathematical programming*, 73:291–341, 1996.
- [39] P. Van Dooren. A generalized eigenvalue approach for solving Riccati equations. *SIAM Journal on Scientific and Statistical Computing*, 2(2):121–135, 1981.
- [40] L. Vandenberghe, V. R. Balakrishnan, R. Wallin, A. Hansson, and T. Roh. *Positive Polynomials in Control*, chapter Interior-point algorithms for semidefinite programming problems derived from the KYP lemma. Lecture Notes on Control and Information Sciences. Springer Verlag, 2005.
- [41] A. Varga and P. Parrilo. Fast algorithms for solving  $H_\infty$ -norm minimization problem. In *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, Florida, USA, 2003.
- [42] R. Wallin and A. Hansson. KYPD: A solver for semidefinite programs derived from the Kalman-Yakubovich-Popov lemma. In *IEEE Conference on Computer Aided Control Systems Design*, Taipei, Taiwan, 2004.
- [43] R. Wallin, H. Hansson, and L. Vandenberghe. Efficient implementations of interior-point methods for integral quadratic constraints. In *Fourth SIAM conference on linear algebra in signals, systems and control*, Boston, Massachusetts, USA, 2001. *Abstract only*.
- [44] R. Wallin, H. Hansson, and L. Vandenberghe. Comparison of two structure exploiting algorithms for integral quadratic constraints. In *4th IFAC Symposium on Robust Control Design*, Milan, Italy, 2003.
- [45] J. C. Willems. Least squares stationary optimal control and the algebraic Riccati equation. *IEEE Transactions on Automatic Control*, AC-16(6):621–634, 1971.
- [46] Y. Ye. A potential reduction algorithm allowing column generation. *SIAM Journal of Optimization*, 2:7–20, 1992.
- [47] Y. Ye. Complexity analysis of the analytical center cutting plane method that uses multiple cuts. *Mathematical Programming*, 78:85–104, 1997.
- [48] Y. Ye. *Interior point algorithms: Theory and analysis*. John Wiley & Sons, New York, New York, USA, 1997.
- [49] K. Zhou, J. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice Hall, Upper Saddle River, New Jersey, USA, 1996.

## A Proof of the existence of a unique maximal solution to the algebraic Riccati equation

The proof follows a similar proof in [49]. Let  $P \in S^n$  be such that  $\mathcal{Q}(P) > 0$ . As  $(A, B)$  is stabilizable there exists an  $F_0$  such that

$$A_0 = A + BF_0$$

is Hurwitz. Now let  $P_0$  be the unique solution to the Lyapunov equation

$$A_0^T P_0 + P_0 A_0 + F_0^T R F_0 + F_0^T S^T + S F_0 + Q = 0$$

Then  $P_0$  is symmetric. Define

$$\hat{F}_0 = F_0 + R^{-1}(PB + S)^T$$

and we get

$$A_0^T(P_0 - P) + (P_0 - P)A_0 = -\hat{F}_0^T R \hat{F}_0 - Q(P) < 0$$

The stability of  $A_0$  implies that

$$P_0 > P$$

Starting with  $P_0$ , we shall define a decreasing sequence of symmetric matrices  $\{P_i\}$ . Associated with  $\{P_i\}$ , we shall also define a sequence of Hurwitz matrices  $\{A_i\}$  and a sequence of matrices  $\{F_i\}$ . Assume inductively that we have already defined matrices  $\{P_i\}$ ,  $\{A_i\}$  and  $\{F_i\}$  for  $i$  up to  $n-1$  such that  $\{P_i\}$  is symmetric and

$$P_0 > P_1 > \dots > P_{n-1} > P$$

$$A_i = A + BF_i \text{ is Hurwitz,} \quad (\text{A.1})$$

$$i = 1, 2, \dots, n-1$$

$$F_i = -R^{-1}(P_{i-1}B + S)^T, \quad (\text{A.2})$$

$$i = 1, 2, \dots, n-1$$

$$A_i^T P_i + P_i A_i = -F_i^T R F_i - F_i^T S^T - S F_i - Q, \quad (\text{A.3})$$

$$i = 1, 2, \dots, n-1 \quad (\text{A.4})$$

Next, introduce

$$F_n = -R^{-1}(P_{n-1}B + S)^T$$

$$A_n = A + BF_n$$

We shall first show that  $A_n$  is Hurwitz. Then, using (A.4) we define a symmetric matrix  $P_n$  with  $P_{n-1} > P_n > 0$ . Now, using (A.4), with  $i = n-1$ , we get

$$A_{n-1}^T P_{n-1} + P_{n-1} A_{n-1} + F_{n-1}^T R F_{n-1}$$

$$+ F_{n-1}^T S^T + S F_{n-1} + Q =$$

$$A_n^T P_{n-1} + P_{n-1} A_n + Q + F_n^T R F_n + F_n^T S^T + S F_n$$

$$+ (F_n - F_{n-1})^T R (F_n - F_{n-1}) = 0$$

Let

$$\hat{F}_n = F_n + R^{-1}(PB + S)^T$$

then

$$A_n^T(P_{n-1} - P) + (P_{n-1} - P)A_n = -\hat{F}_n^T R \hat{F}_n$$

$$- Q(P) - (F_n - F_{n-1})^T R (F_n - F_{n-1}) \quad (\text{A.5})$$

Now assume that  $A_n$  is not Hurwitz, i.e. there exists a  $\lambda$  with  $\text{Re}\lambda \geq 0$  and  $x \neq 0$  such that  $A_n x = \lambda x$ . Pre-multiply (A.5) with  $x^T$  and post-multiply by  $x$

$$2\text{Re}\lambda x^T (P_{n-1} - P)x =$$

$$-x^T (\hat{F}_n^T R \hat{F}_n + Q(P) + (F_n - F_{n-1})^T R (F_n - F_{n-1}))x$$

Since it is assumed that  $P_{n-1} > P$  each term on the right-hand side of the above equation has to be zero. Thus, we have

$$x^T (F_n - F_{n-1})^T R (F_n - F_{n-1})x = 0$$

This implies

$$(F_n - F_{n-1})x = 0$$

But now

$$A_{n-1}x = (A + BF_{n-1})x = (A + BF_n)x = A_n x = \lambda x$$

which is a contradiction with  $A_{n-1}$  being Hurwitz. Hence  $A_n$  is Hurwitz. Introduce  $P_n$  as the unique solution to the Lyapunov equation

$$A_n^T P_n + P_n A_n = -F_n^T R F_n - F_n^T S^T - S F_n - Q \quad (\text{A.6})$$

Then  $P_n$  is symmetric. Next, we have

$$A_n^T(P_n - P) + (P_n - P)A_n = -\hat{F}_n^T R \hat{F}_n - Q(P) < 0$$

and by using (A.5)

$$A_n^T(P_{n-1} - P_n) + (P_{n-1} - P_n)A_n =$$

$$- (F_n - F_{n-1})^T R (F_n - F_{n-1}) < 0$$

Since  $A_n$  is Hurwitz we have

$$P_{n-1} > P_n > P$$

We have a decreasing sequence  $\{P_i\}$  and the sequence is bounded below by  $P_i > P$ . Hence the limit

$$P_f = \lim_{n \rightarrow \infty} P_n$$

exists and is symmetric and we have  $P_f > P$ . Passing the limit in (A.6) we get  $Q(P_f) = 0$ . Hence,  $P_f$  is a solution to the algebraic Riccati equation. Since  $P$  is an arbitrary element satisfying  $Q(P) > 0$  and  $P_f$  is independent of the choice  $P$ , we have

$$P_f > P, \quad \forall P \text{ such that } Q(P) > 0$$

In particular,  $P_f$  is the maximal solution of the algebraic Riccati equation. To establish the Hurwitz property of the maximal solution, note that  $A_n$  is Hurwitz for any  $n$ . Hence, in the limit, the eigenvalues of

$$A - BR^{-1}(P_f B + S)^T$$

will have nonpositive real parts. We also have that

$$A_r^T(P_r - P) + (P_r - P)A_r =$$

$$-Q(P) - (P_r - P)BR^{-1}B^T(P_r - P) < 0 \quad (\text{A.7})$$



Now suppose  $(P_r - P)$  is singular and there is an  $x \neq 0$  such that  $(P_r - P)x = 0$ . By premultiplying (A.7) with  $x^H$  and postmultiplying (A.7) by  $x$  we get  $x^H Q(P)x = 0$ , a contradiction. That  $A_r = A - BR^{-1}(P_r B + S)^T$  is Hurwitz then follows from the Lyapunov theorem.

## B Relaxing Assumptions

The assumption that  $A$  is Hurwitz is only used to assure that  $A$  has no eigenvalues on the imaginary axis. Suppose  $A$  is not Hurwitz. As the pair  $(A, B)$  is stabilizable we know that there exists an  $F$  such that  $\bar{A} = A + BF$  is Hurwitz. Define the full rank matrix

$$T = \begin{bmatrix} I & 0 \\ F & I \end{bmatrix}$$

Premultiplying the constraint (1) with  $T^T$  and postmultiply with  $T$  yields

$$\begin{aligned} & \inf_{x, P} c^T x + \text{Tr}(CP) \\ & \text{s.t.} \quad \begin{bmatrix} \bar{A}^T P + P \bar{A} & P B \\ B^T P & 0 \end{bmatrix} + \bar{M}_0 + \sum_{i=1}^p x_i \bar{M}_i > 0 \end{aligned}$$

where  $\bar{A} = A + BF$  is Hurwitz and

$$\bar{M}_i = T^T M_i T, \quad k = 0, 1, \dots, p$$

The solution  $(x_{\text{opt}}, P_{\text{opt}})$  is the same as for the original problem, because semidefiniteness is invariant under congruence.