# A spectral quadratic-SDP method with applications to fixed-order $H_2$ and $H_\infty$ synthesis

*Pierre Apkarian* [*] *Dominikus Noll*[‡] *Jean-Baptiste Thevenet*[¶]
*Hoang Duong Tuan*[**]

**Abstract**

In this paper, we discuss a spectral quadratic-SDP method for the iterative resolution of fixed-order $H_2$ and $H_\infty$ design problems. These problems can be cast as regular SDP programs with additional nonlinear equality constraints. When the inequalities are absorbed into a Lagrangian function the problem reduces to solving a sequence of SDPs with quadratic objective function for which a spectral SDP method has been developed. Besides a description of the spectral SDP method used to solve the tangent subproblems, we report a number of computational results for validation purposes.

**Keywords:** Linear matrix inequality (LMI), semidefinite programming (SDP), bilinear matrix inequality (BMI), augmented Lagrangian method (AL), fixed-order synthesis, reduced-order synthesis, rank constraints, robust synthesis.

## 1 Introduction

Algebraically or rank-constrained LMI problems frequently arise in control engineering applications. A typical example is fixed- and reduced-order synthesis of output feedback controllers. The present paper is mainly concerned with this challenging problem, but our solution strategy applies to many other practical problems in control (see e.g. [25, 13, 12]).

We present an iterative technique which allows to compute solutions of the fixed-order $H_2$ and $H_\infty$ synthesis problem. Our method computes a stabilizing reduced-order controller whose $H_2$ or $H_\infty$ performance channel is locally optimal among all such controllers. We have previously presented two classes of nonlinear programming procedures which achieve similar goals: an augmented Lagrangian (AL) method [27, 2, 12] and a sequential semidefinite programming (S-SDP)

[*]ONERA-CERT, Centre d'études et de recherche de Toulouse, Control System Department, 2 av. Edouard Belin, 31055 Toulouse, France - Email : apkarian@cert.fr - Tel : +33 5.62.25.27.84 - Fax : +33 5.62.25.27.64.

[‡]Université Paul Sabatier, Mathmatiques pour l'Industrie et la Physique, 118, route de Narbonne, 31062 Toulouse, France - Email : noll@mip.ups-tlse.fr - Tel : +33 5.61.55.86.22 - Fax : +33 5.61.55.83.85.

[¶]ONERA-CERT and Université Paul Sabatier, Toulouse, France - Email : thevenet@certfr - Tel : +33 5.62.25.22.11

[**]School of Electrical and Telecommunication Engineering, University of New South Wales, UNSW Sydney, NSW 2052, Australia - Email: h.d.tuan@unsw.edu.au -Tel +61-2-9385-5375 - Fax +61-2-9385-5993

algorithm [13], which expands on the classical SQP method. SQP and AL have been known at least since the 1970s in the context of mathematical programming with classical equality and inequality constraints. Their local and global convergence properties have been extensively studied over the years, see for instance [8, 10, 9] for AL or [5, 6] for SQP. In [13] and [27] we have established similar convergence features for the more general programs under matrix inequality constraints. Proving global and fast local convergence of these methods is important since an increasing number of problems in control is identified as suited for optimization programs under matrix inequality constraints. The discussion in [27] focuses on convergence properties of the AL method. Here we dwell on a number of practical features aiming at a more efficient and reliable implementation of the AL algorithm in the specific setting of fixed-order $H_2$ and $H_\infty$ synthesis. Specifically, the AL method relies on the solution of tangent subproblems in the form of the minimization of a quadratic objective subject to SDP constraints. When suitably convexified, these problems can be solved using standard SDP codes. This option, however, is practically inefficient since the rearrangement of the tangent subproblem into a standard SDP requires an additional large and dense SDP constraint which in turn leads to prohibitive running times when standard primal-dual interior-point solvers are employed. This holds even for problem of modest sizes. In the interest of efficiency, we therefore propose a spectral quadratic SDP approach to solve the tangent subproblems directly. This is examined in sections 4 and 5 subsequently to a brief description of the AL algorithm in section 3. Finally, numerical examples are presented in section 6 to validate the proposed techniques.

# Notation

Our notation is standard. We let $\mathbb{S}^n$ denote the set of $n \times n$ symmetric matrices, $M^T$ the transpose of the matrix $M$ and $\operatorname{Tr} M$ its trace. For Hermitian or symmetric matrices , $M \succ N$ means that $M - N$ is positive definite and $M \succeq N$ means that $M - N$ is positive semi-definite. $\|A\|_F$ denotes the Frobenius norm of the matrix $A$. The symbol $\otimes$ stands for the usual Kronecker product of matrices and vec stands for the columnwise vectorization on matrices. We shall make use of the properties:

$$\operatorname{vec}(AXB) = (B^T \otimes A) \operatorname{vec} X, \ \operatorname{Tr}(AB) = \operatorname{vec}^T A^T \operatorname{vec} B$$

which hold for any matrices $A$, $X$ and $B$ of compatible dimensions. The Hadamard or Schur product is defined as

$$A \circ B = ((A_{ij} B_{ij})) .$$

The following holds for matrices of the same dimension:

$$\operatorname{vec} A \circ \operatorname{vec} B = \operatorname{diag}(\operatorname{vec} A) \operatorname{vec} B ,$$

where the operator diag forms a diagonal matrix with vec $A$ on the main diagonal.

# 2 Characterizations for fixed-order syntheses

In this section, we give suitable conditions for $H_2$ and $H_\infty$ syntheses. Since these results are fairly standard, we simply recap the central facts. We start with the simpler static output feedback and show how the more general fixed-order case can be handled through straightforward transformations of the problem data.

The general setting of the fixed-order synthesis problem is as follows. We consider a linear time-invariant plant $P(s)$ described in "standard form" by the state-space equations:

$$P(s): \qquad \begin{bmatrix} \dot{x} \\ z \\ y \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix}, \tag{1}$$

where

- $x \in \mathbb{R}^n$ is the state vector,
- $u \in \mathbb{R}^{m_2}$ is the vector of control inputs,
- $w \in \mathbb{R}^{m_1}$ is a vector of exogenous inputs,
- $y \in \mathbb{R}^{p_2}$ is the vector of measurements,
- $z \in \mathbb{R}^{p_1}$ is the controlled or performance vector.
- $D_{22} = 0$ is assumed without loss of generality.

Let $T_{w,z}(s)$ denote the closed-loop transfer functions from $w$ to $z$ for some static output-feedback control law

$$u = Ky\,. \tag{2}$$

Our aim is to compute $K$ subject to the following design constraints:

- *internal stability:* for $w = 0$ the state vector of the closed-loop system (1) and (2) tends to zero as time goes to infinity.
- *performance:* the $H_\infty$ norm $\|T_{w,z}(s)\|_\infty$ respectively the $H_2$ norm $\|T_{w,z}(s)\|_2$ is minimized where the closed-loop transfer $\|T_{w,z}(s)\|$ is described as

$$T_{w,z}(s): \left\{ \begin{array}{rcl} \dot{x} & = & (A + B_2KC_2)\,x + (B_1 + B_2KD_{21})\,w \\ z & = & (C_1 + D_{12}KC_2)\,x + (D_{11} + D_{12}KD_{21})\,w\,. \end{array} \right.$$

## 2.1 Static $H_\infty$ synthesis

With the above ingredients, the static $H_\infty$ synthesis problem is first transformed into a matrix inequality condition using the bounded real lemma [1]. Then the projection lemma from [16] is used to eliminate the unknown controller data $K$ from the cast. This gives:

**Proposition 2.1** *A stabilizing static output feedback controller $K$ with $H_\infty$ gain $\|T_{w,z}(s)\|_\infty \leq \gamma$ exists provided there exist $X, Y \in \mathbb{S}^n$ such that*

$$\mathcal{N}_Q^T \begin{bmatrix} A^TX + XA & XB_1 & C_1^T \\ B_1^TX & -\gamma I & D_{11}^T \\ C_1 & D_{11} & -\gamma I \end{bmatrix} \mathcal{N}_Q \;\prec\; 0 \tag{3}$$

$$\mathcal{N}_P^T \begin{bmatrix} YA^T + AY & B_1 & YC_1^T \\ B_1^T & -\gamma I & D_{11}^T \\ C_1Y & D_{11} & -\gamma I \end{bmatrix} \mathcal{N}_P \;\prec\; 0 \tag{4}$$

$$\begin{bmatrix} X & I \\ I & Y \end{bmatrix} \succ 0, \qquad XY - I \;=\; 0 \tag{5}$$

*where $\mathcal{N}_Q$ and $\mathcal{N}_P$ denote bases of the nullspaces of $Q := [\,C_2 \quad D_{21} \quad 0\,]$ and $P := [\,B_2^T \quad D_{12}^T \quad 0\,]$, respectively.* □

The reader is referred to [16, 3] for proofs and further details.

## 2.2 Static $H_2$ synthesis

The fixed-order $H_2$ synthesis follows similar lines. Again, the projection lemma [16] is the key tool to eliminate variable redundancies and turns a nonlinear SDP program into the more accessible form below. Recall that in the $H_2$ case, some feedtrough terms must be nonexistent in order for the $H_2$ performance to be well defined. We therefore assume

$$D_{11} = 0,\ D_{21} = 0$$

for the plant in (1).

With this extra condition we have the following

**Proposition 2.2** *A stabilizing static output feedback controller $K$ with $H_2$ performance $\|T_{w,z}(s)\|_2 \leq \sqrt{\gamma}$ exists provided there exist $X, Y \in \mathbb{S}^n$ such that*

$$\mathcal{N}_Q^T \begin{bmatrix} A^TX + XA & C_1^T \\ C_1 & -I \end{bmatrix} \mathcal{N}_Q \ \prec \ 0 \tag{6}$$

$$\mathcal{N}_P^T \begin{bmatrix} YA^T + AY & YC_1^T \\ C_1Y & -I \end{bmatrix} \mathcal{N}_P \ \prec \ 0 \tag{7}$$

$$\mathrm{Tr}\,(B_1^TXB_1) \leq \gamma \tag{8}$$

$$\begin{bmatrix} X & I \\ I & Y \end{bmatrix} \succ 0, \qquad XY - I \ = \ 0 \tag{9}$$

*where $\mathcal{N}_Q$ and $\mathcal{N}_P$ denote bases of the nullspaces of $Q := [\,C_2 \quad 0\,]$ and $P := [\,B_2^T \quad D_{12}^T\,]$, respectively.* □

## 2.3 Fixed-order synthesis

Fixed-order synthesis is concerned with the design of a dynamic controller $K(s) = C_K(sI - A_K)^{-1}B_K + D_K$ where $A_K \in \mathbb{R}^{k\times k}$ and $k < n$. It can be regarded as a static gain synthesis problem for an augmented system. Consequently, propositions 2.1 and 2.2 apply if we perform the following substitutions:

$$\begin{aligned} &K \to \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix}, \quad A \to \begin{bmatrix} A & 0 \\ 0 & 0_k \end{bmatrix} \qquad B_1 \to \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \qquad C_1 \to [\,C_1 \quad 0\,] \\ &B_2 \to \begin{bmatrix} 0 & B_2 \\ I_k & 0 \end{bmatrix}, \quad C_2 \to \begin{bmatrix} 0 & I_k \\ C_2 & 0 \end{bmatrix}, \quad D_{12} \to [\,0 \quad D_{12}\,], \quad D_{21} \to \begin{bmatrix} 0 \\ D_{21} \end{bmatrix}. \end{aligned} \tag{10}$$

The Lyapunov variables $X$ and $Y$ now lie in the augmented space $\mathbb{S}^{n+k}$. Recall that when a solution $(X, Y, \gamma)$ in proposition 2.1 or 2.2 has been computed, the corresponding optimal controller, static or dynamic, is reconstructed in an extra step by solving a single LMI problem [16].

# 3 Augmented Lagrangian with explicit LMIs

In this section, we describe how a local optimal solution of the fixed-order synthesis problem is computed. The problem is cast as an optimization problem with cost function the performance index $\gamma$ minimized subject to nonlinear equality and inequality constraints in propositions 2.1 and 2.2

$$\begin{array}{ll} \text{minimize} & f(x) := \gamma \\ \text{subject to} & h(x) := XY - I = 0\,, \\ & \mathcal{A}(x) \preceq 0\,. \end{array} \tag{11}$$

Here $x \in \mathbf{R}^N$, $N = (n+k)(n+k+1)+1$ regroups $x = (\gamma, X, Y)$ and should not be confused with the state vector in (1). The dimension of the equality constraint $h : \mathbf{R}^N \to \mathbf{R}^M$ is $M = (n+k)^2$. For a static controller we have $k = 0$. For notational ease the LMI constraints in (3) – (5), respectively (6) – (9), have been condensed into the single LMI $\mathcal{A}(x) \preceq 0$, where $\prec 0$ have systematically been replaced with $\preceq -\varepsilon I$ for a suitable small threshold $\varepsilon > 0$ to guarantee strict feasibility of the LMIs.

Following the idea of a partially augmented Lagrangian method, program (11) is now solved by a succession of simpler programs:

$$(P_{c,\Lambda}) \quad \begin{array}{ll} \text{minimize} & \Phi_c(x, \Lambda) \\ \text{subject to} & \mathcal{A}(x) \preceq 0 \end{array} \tag{12}$$

where $\Phi_c(x, \Lambda)$ is the augmented Lagrangian function

$$\Phi_c(x, \Lambda) = \gamma + \sum_{ij} \Lambda_{ij}(XY - I)_{ij} + \frac{c}{2} \sum_{ij} (XY - I)_{ij}^2\,,$$

equivalently expressed in matrix form as:

$$\Phi_c(x, \Lambda) = \gamma + \mathrm{Tr}\Big(\Lambda^T (XY - I)\Big) + \frac{c}{2}\mathrm{Tr}\Big((XY - I)^T (XY - I)\Big)\,. \tag{13}$$

Here $c$ is a positive penalty and $\Lambda$ is a matrix-valued Lagrange multiplier estimate. In order to drive the solutions $x_{c,\Lambda}$ of $(P_{c,\Lambda})$ to a solution of the original program (11), a suitable updating strategy $c \to c^+$, $\Lambda \to \Lambda^+$ has to be used. The algorithm below shows how this is done.

The rationale in $(P_{c,\Lambda})$ is that removing the difficult equality constraints $h = 0$ in (5) by putting them into the augmented objective leads to a program $(P_{c,\Lambda})$ which is easier to solve. Classical AL strategies would even recommend a similar augmentation for the matrix inequality constraints $\mathcal{A}(x) \preceq 0$ in order to end up with an unconstrained program. We prefer to keep the LMIs as explicit constraints due to their simpler affine structure.

Each of the new optimization problems (12) is itself solved by a succession of SDPs. This requires that at the current point $x$, a new iterate $x^+ = x + dx$ is obtained by minimizing the suitably convexified second-order Taylor series approximation of $\Phi_c(x+dx, \Lambda)$ about the current $x$ and subject to $\mathcal{A}(x+dx) \preceq 0$. Without convexification of the Hessian $\Phi_c''$, these tangent problems would remain difficult to solve. More details on how the SDP subproblems are solved will be presented in sections 4 and 5.

It is important to keep in mind that the motivation for using AL is that, for an appropriate choice of $(\Lambda^*, c^*)$, a local optimal solution $x^*$ of the original program (11) can be found by simply optimizing the function $\Phi_{c^*}(x, \Lambda^*)$ with respect to $x$. A thorough discussion on this mechanism

for classical constraints is given in [14]. Of course, the central task is to determine $(\Lambda^*, c^*)$, and the AL algorithm achieves this goal by forming a sequence $(\Lambda^k, c^k)$ converging to $(\Lambda^*, c^*)$.

Since the proposed algorithm is of second-order type, we need to compute the gradient and Hessian of (13). The first order information at the point $x = (X, Y, \gamma)$ is easily obtained. With $T$ the transformation matrix mapping the vectorized lower triangle of the symmetric matrix $X$ into its vec representation, the Jacobian of the matrix function $h(x) = XY - I$ is

$$J(x) = [\,(Y \otimes I)T \quad (I \otimes X)T \quad 0\,]\,,$$

and the gradient of the augmented Lagrangian $\nabla_x \Phi_c(x, \Lambda)$ is computed as

$$\nabla_x \Phi_c(x, \Lambda) = \begin{bmatrix} T^T \mathrm{vec}\,(\Lambda Y) \\ T^T \mathrm{vec}\,(X\Lambda) \\ 1 \end{bmatrix} + c\, J(x)^T \mathrm{vec}\,(XY - I)\,. \tag{14}$$

The Gauss-Newton approximation $\nabla^{GN}_{xx} \Phi_c(x, \Lambda)$ of the Hessian is

$$\nabla^{GN}_{xx} \Phi_c = \begin{bmatrix} 0 & T^T(I \otimes \Lambda)T & 0 \\ T^T(I \otimes \Lambda^T)T & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + c\, J(x)^T J(x)\,. \tag{15}$$

By definition it is obtained by omitting the term

$$c \sum_{i=1}^{M} h_i(x) \nabla^2_{xx} h_i(x) \tag{16}$$

from the full Hessian expansion $\nabla^2_{xx} \Phi_c(x, \Lambda)$. The rationale in (15) is that (16) is small when iterates $x$ get close to feasibility, where $h_i(x)$ get smaller. The Gauss-Newton approximation has the further advantage that it is easier to compute than $\nabla^2_{xx} \Phi_c$, is more positive definite than the true Hessian and asymptotically converges to the true Hessian, as $h(x)$ gets closer to zero by virtue of expression (16). With these preparations, a general description of the algorithm is now the following.

**AL-Algorithm for Fixed-Order Synthesis**

1. **Initial phase.** Choose a starting point $x^0$ such that $\mathcal{A}(x^0) \preceq 0$. This can be done by simply solving a feasibility SDP. Then initialize the penalty parameter $c^0 > 0$ and the Lagrange multiplier $\Lambda^0$. Fix $\rho < 1$, $0 < \mu < 1$ and $\varepsilon > 0$.

2. **Optimization phase.** Given $x^j$, $c^j > 0$ and the multiplier estimate $\Lambda^j$, solve the subproblem $(P_{c^j,\Lambda^j})$, i.e., minimize $\Phi_{c^j}(x, \Lambda^j)$ over $\{x : \mathcal{A}(x) \preceq 0\}$. Let $x^{j+1}$ be the solution so obtained. Possibly use $x^j$ as a starting point for the inner optimization.

3. **Update penalty and multiplier.**

$$\Lambda^{j+1} = \Lambda^j + c^j(X^{j+1}Y^{j+1} - I). \tag{17}$$

$$c^{j+1} = \begin{cases} \rho c^j & \text{if } \|X^{j+1}Y^{j+1} - I\|_F > \mu\|X^jY^j - I\|_F \\ c^j & \text{if } \|X^{j+1}Y^{j+1} - I\|_F \leq \mu\|X^jY^j - I\|_F \end{cases} \tag{18}$$

4. **Stopping test.** Dispense with the iteration if $\|X^{j+1}Y^{j+1} - I\|_F < \varepsilon$ and the necessary optimality conditions are satisfied or if the progress of the algorithm is negligible. Otherwise increase counter $j$ and go back to step 2.

5. **Terminating phase.** Given the solution $x = (\gamma, X, Y)$ with $\|XY - I\|_F < \varepsilon$, try to reconstruct a $k$-th-order controller. If the reconstruction fails, reduce $\varepsilon$, increase counter $j$ and go back to step 2.

# 4 Tangent subproblems

The optimization phase in our algorithm solves program $(P_{c,\Lambda})$ for fixed $c$ and $\Lambda$. This minimization is performed iteratively by generating search directions $dx$ about the current iterate $x$ through the tangent quadratic model

$$\begin{array}{ll} \min & \nabla\Phi_c(x,\Lambda)^T\, dx + \frac{1}{2}\, dx^T\, H\, dx \\ \text{s.t} & \mathcal{A}(x + dx) \preceq 0\,. \end{array} \tag{19}$$

In our present implementation, $H$ is a convexified version of the Hessian of the Lagrangian of $(P_{c,\Lambda})$, which by the linearity of the constraint $\mathcal{A}(x) \preceq 0$ is just a convexified version of $\Phi_c''$.

The resolution of the tangent subproblems $(P_{c,\Lambda})$ is the major computational load of the AL algorithm. It is therefore of the essence to avoid numerical fallacies. For instance, with $H \succ 0$ it is tempting to replace the quadratic term in the objective $x^THx$ by an additional LMI constraint. This is achieved through a Schur complement transformation:

$$\begin{array}{ll} \min & t \\ \text{s.t.} & \begin{bmatrix} t - \nabla\Phi_c(x,\Lambda)^T\, dx & dx^T \\ dx & 2\, H^{-1} \end{bmatrix} \geq 0 \\ & \mathcal{A}(x + dx) \preceq 0\,, \end{array}$$

In this form, the subproblem can be solved by currently available SDP solvers [17, 29]. However, this reformulation is inefficient as it creates an additional large and dense SDP constraint, which is usually even larger than the genuine LMIs $\mathcal{A}(x) \preceq 0$. It is therefore recommended to solve (19) directly, and the next section shows how this can be done.

# 5 A spectral quadratic SDP method

As we have just discussed, the resolution of the tangent subproblem (19) requires special attention as it is a critical component which determines both accuracy and efficiency of the proposed technique. We now discuss our Fortran 90 implementation of a spectral quadratic SDP method for programs of the form:

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T \, H \, x \\ \text{s.t} & \mathcal{A}(x) \preceq 0 \, . \end{array} \tag{20}$$

As before, $\mathcal{A} : \mathbb{R}^n \to \mathbb{S}^m$ is affine. Whenever convenient, we will expand the SDP constraint in the form $\mathcal{A}(x) = A_0 + \sum_{i=1}^{n} x_i A_i$. Our method is an extension of a penalty/barrier multiplier algorithm in [26] to quadratic objectives. A spectral penalty (SP) function for (20) is defined as

$$F(x,p) = c^T x + \frac{1}{2} x^T \, H \, x + \mathrm{Tr}\,(\phi_p(\mathcal{A}(x)) \, . \tag{21}$$

Here, $\phi_p$ is a scalar function which can be classically extended to a matrix function on the set of symmetric matrices by defining:

$$\phi_p(\mathcal{A}(x)) := S \, \mathrm{diag}(\phi_p(\lambda_1(\mathcal{A}(x)), \ldots, \lambda_m(\mathcal{A}(x)))S^T \, , \tag{22}$$

where $\lambda_i(\mathcal{A}(x))$ stands for the $i$th eigenvalue of $\mathcal{A}(x)$ and $S$ is the orthonormal matrix of associated eigenvectors. An alternative expression is then readily derived from (21) and (22)

$$F(x,p) = c^T x + \frac{1}{2} x^T \, H \, x + \sum_{i=1}^{m} \phi_p(\lambda_i(\mathcal{A}(x)) \, . \tag{23}$$

The rationale in (21) is that (20) and

$$\begin{array}{ll} \min & c^T x + \frac{1}{2} x^T \, H \, x \\ \text{s.t.} & \phi_p(\mathcal{A}(x)) \preceq 0 \, . \end{array} \tag{24}$$

are equivalent whenever $\phi_p$ is a strictly increasing scalar function such that $\phi_p(0) = 0$. In our implementation and following the recommendation in [26], we have used the log-quadratic penalty function $\phi_p(t) = p\phi_1(t/p)$ where

$$\phi_1(t) = \left\{ \begin{array}{lll} t + \frac{1}{2}t^2 & \text{if} & t \geq -\frac{1}{2} \\ -\frac{1}{4}\log(-2t) - \frac{3}{8} & \text{if} & t < -\frac{1}{2} \end{array} \right. .$$

The scalar $p$ is a penalty parameter which tends to zero and is used to enforce feasibility in the course of the iterations.

So far, we have only discussed pure SP functions. As with more classical penalty functions, SP functions lead to ill-conditioned minimization problems for small values of the penalty parameter.

An effective technique to alleviate this difficulty is to introduce Lagrange multipliers associated with the SDP constraints in (20). The resulting SP function is then an augmented Lagrangian function and one such candidate is described as

$$F(x,V,p) = c^T x + \frac{1}{2}\, x^T\, H\, x + \mathrm{Tr}\,(\phi_p(V^T \mathcal{A}(x) V)) = c^T x + \frac{1}{2}\, x^T\, H\, x + \sum_{i=1}^{m} \phi_p(\lambda_i(V^T \mathcal{A}(x) V))\,. \quad (25)$$

In this expression, the variable $V$ has the same dimension as $\mathcal{A}(x)$ and plays the role of a Lagrange multiplier factor as explained in the sequel. Note that in contrast with classical (quadratic) augmented Lagrangians, the Lagrange multiplier is not involved linearly in (25). Interestingly, this is not at all troublesome and a suitable first-order update formula $V \to V^+$ generalizing the classical case will be derived in section 5.2. Schematically, the spectral quadratic SDP technique is as follows:

---

**Spectral Quadratic SDP Technique**

1. **Initial phase.** Initialize the algorithm with $x^0$, $V^0$ and a penalty parameter $p^0$. Define $\rho$ with $0 < \rho \leq 1$.

2. **Optimization phase.** For fixed $V^j$ and $p^j$, minimize $F(x, V^j, p^j)$ and let $x^{j+1}$ be the solution. Use the previous iterate $x^j$ as a starting value for the inner optimization.

3. **Update penalty and multiplier.**. Update Lagrange multiplier factor $V^j \to V^{j+1}$ (see below) and penalty parameter $p^{j+1} = \rho\, p^j$. Increase $j$ and go back to step 2.

---

Minimization of $F(x, V^j, p^j)$ can be based on a Newton line search or a trust region approach. The latter is preferable when nonconvex problems are allowed. In either case, it is necessary to derive explicit formulas for the gradient and Hessian of $F(x, V, p)$.

## 5.1 Derivatives of SP functions

In the sequel, we give a rigorous derivation for the gradient and Hessian of $F(x, V, p)$.

As exposed in (25), a SP function is a function of eigenvalues $g(\lambda_1(x), \ldots, \lambda_m(x))$. According to the beautiful theory of spectral functions [23, 24], see also [28], they are continuously differentiable up to second-order whenever this is so for $g$ at $(\lambda_1(x), \ldots, \lambda_m(x))$. Following the derivation in Lewis [23, 24], given a symmetric function $f : \mathbb{R}^m \to \mathbb{R}$ and $\lambda : \mathbb{S}^m \to \mathbb{R}^m$ representing the eigenvalue map $\lambda(X) := (\lambda_1(X), \ldots, \lambda_m(X))$, the gradient of $f \circ \lambda$ at $X$ is obtained as

$$\nabla(f \circ \lambda)(X) = S\, \mathrm{diag}\, \nabla f\,(\lambda(X)) S^T\,,$$

where $S$ is an orthogonal matrix of eigenvectors, $X = S\, \mathrm{diag}\, \lambda(X) S^T$. Accordingly, its differential is obtained as

$$d(f \circ \lambda(X))[dX] = \mathrm{Tr}\,(S\, \mathrm{diag}\, \nabla f\,(\lambda(X)) S^T\, dX)\,.$$

Exploiting this result with $f(x) := \sum_{i=1}^m \phi_p(x_i)$ yields the differential of the SP functions in (25)

$$\begin{aligned} dF(x,V,p)[dx] &= (c+Hx)^T dx + \mathrm{Tr}\,(S\ \mathrm{diag}\,\phi_p'(\lambda_i(V^T\mathcal{A}(x)V))\,S^T\,d(V^T\mathcal{A}(x)V)) \\ &= (c+Hx)^T dx + \mathrm{Tr}\,(VS\ \mathrm{diag}\,\phi_p'(\lambda_i(V^T\mathcal{A}(x)V))(VS)^T\mathcal{A}_0(dx))\,, \end{aligned} \tag{26}$$

where $S$ is an orthogonal matrix of eigenvectors of $V^T\mathcal{A}(x)V$ and with the definition $\mathcal{A}_0(x) = \sum_{i=1}^n x_i A_i$, the linear part of $\mathcal{A}$. A vector form suitable for computations is then readily obtained as

$$dF(x,V,p)[dx] = dx^T(c + Hx + \mathbf{A}^T\ \mathrm{vec}\,(VS\ [\mathrm{diag}\,\phi_p'(\lambda_i(V^T\mathcal{A}(x)V))]\ (VS)^T)) \tag{27}$$

Here $\mathrm{vec}\,\mathcal{A}_0(dx) = \mathbf{A}\,dx$ when we define $\mathbf{A} := [\mathrm{vec}\,A_1 \ldots \mathrm{vec}\,A_n]$.

For the Hessian we use again a central result in [24], which states that for $f(x) = \sum_{i=1}^m \phi_p(x_i)$,

$$d^2(f\circ\lambda)(X)[dX,\,dX] = \mathrm{Tr}\,(S^T dX\,S\ H\circ\{S^T\,dX\,S\})\,,$$

where the symbol $\circ$ denotes the Hadamard or Schur product of 2 matrices and $H$ is defined as

$$H_{i,j} := \begin{cases} \frac{\phi_p'(\lambda_i)-\phi_p'(\lambda_j)}{\lambda_i-\lambda_j} & \text{if} \quad \lambda_i \neq \lambda_j \\ \phi_p''(\lambda_i) & \text{if} \quad \lambda_i = \lambda_j \end{cases}\,.$$

We obtain the quadratic form of the second order differential of the SP function as

$$d^2F(x,V,p)[dx,dx] = dx^T\,H\,dx + \mathrm{Tr}\,((VS)^T\mathcal{A}_0(dx)VS[H\circ\{(VS)^T\mathcal{A}_0(dx)VS\}])\,,$$

where $S$ is as in (26). As before this is more conveniently rewritten in vector form

$$d^2F(x,V,p)[dx,dx] = dx^T\,H\,dx + dx^T\mathbf{A}^T\,[VS\otimes VS]\ \mathrm{diag}\,\mathrm{vec}\,H\ [(VS)^T\otimes(VS)^T]\ \mathbf{A}\,dx \tag{28}$$

where properties of the Kronecker and Hadamard products given in the introductory section have been used.

Incidentally, expression (28) shows that the spectral component of $F(x,V,p)$ is convex since $H$ has only nonnegative entries.

## 5.2 Multiplier update rule

The first-order multiplier update rule is derived similarly to the classical case of polyhedral constraints. Assume $x^+$ is the solution of $\min F(x,V,p)$. A new multiplier factor estimate $V^+$ is then obtained by simply equating the gradients of the augmented Lagrangian (25) and the traditional Lagrangian of problem (20). The Lagrangian of (20) can be expressed as

$$L(x,V,p) := c^Tx + \frac{1}{2}\,x^T\,H\,x + \mathrm{Tr}\,(U\mathcal{A}(x)) = c^Tx + \frac{1}{2}\,x^T\,H\,x + \mathrm{Tr}\,(VV^T\mathcal{A}(x))$$

with multiplier $U := VV^T$. Its gradient at $x^+$ computed at the sought estimate $V^+$ thus takes the form

$$\nabla L(x^+,V^+,p) = c + Hx + \mathbf{A}^T\mathrm{vec}\,V^+V^{+T}\,.$$

Equating the latter expression with the gradient in (27) then gives

$$V^+V^{+T} = VS\ [\mathrm{diag}\,\phi_p'(\lambda_i(V^T\mathcal{A}(x)V))]\ (VS)^T\,. \tag{29}$$

Notice that by construction the right hand term is positive definite, so $V^+$ could for instance be chosen as a Cholesky factor.

### 5.3 Comments

We have introduced all ingredients for the implementation of the AL method of section 3. When a Gauss-Newton approximation of the Hessian is used, the quadratic objective is convex in the tangent subproblem and therefore a linesearch can be used within the spectral quadratic SDP method. This is what has been implemented in the alpha version of our Fortran code. The step length in the linesearch is required to satisfy the strong Wolfe conditions (see for instance [14]) in order to guarantee global convergence.

We must keep in mind however that the function in (13) is inherently nonconvex and hence a trust region approach [11] is more likely to produce good results with fewer iterations. An important advantage of the spectral quadratic SDP approach lies in the fact that nonconvex quadratic objectives can be handled when combined with a trust region technique. This promising feature will be studied in a future version of the code.

To conclude, notice that the presented method appears highly complex, as three iterative levels are needed. Naturally, we may consider the solutions of the SDP programs in (20) as black box outputs, where any available SDP solver could be used. However, in the present context it seems worthwhile to open this box and consider solvers tailored to the class of nonconvex quadratic SDP programs (20). Most currently available SDP solvers are not suited and quickly succumb as the problem size grows.

## 6 Numerical experiments

In this section, we evaluate the AL algorithm with the spectral quadratic SDP approach for the innermost tangent subproblems using different synthesis problems. The state-space data of theses applications are fully described in Appendix A. Both static and fixed-order syntheses are considered. Table 1 displays the number of quadratic SDP tangent subproblems that were necessary to construct a locally optimal controller (column 'quad SDP'). Column '$\|h\|_\infty$' provides the infinity norm of the nonlinear equality constraints in (5), (9) and (11) right before controller construction takes place. The last two columns show the achieved $H_\infty$ or $H_2$ performances or both when this applies. The reader is referred to [27] for a catalog of numerical experiments on problems of larger size (up to 40 states).

### 6.1 Transport airplane

We consider the design of a static $H_\infty$ controller for a transport airplane. State-space data as well as further details can be found in [18]. As reported in Table 1, a static $H_\infty$ controller $K$ has been computed after 24 quadratic SDP tangent subproblems of the AL algorithm:

$$K_\infty = [\,0.66997 \quad -0.63801 \quad -0.80285 \quad 0.10263 \quad 1.59297\,]$$

The associated $H_\infty$ performance is 2.22 and represents 30% improvement over the result in [22].

### 6.2 VTOL helicopter

The state-space data of the VTOL helicopter are borrowed from [21]. They are obtained by linearizing the helicopter rigid body dynamics at given flight conditions. This leads to the fourth-

order model described in Appendix A. Both $H_\infty$ and $H_2$ static controllers are computed with the AL method:

$$K_\infty = \begin{bmatrix} 0.80656 \\ 14.94467 \end{bmatrix}, \qquad K_2 = \begin{bmatrix} 0.11630 \\ 5.67734 \end{bmatrix}$$

We note again that $K_\infty$ provides about 40% improvement over the $H_\infty$ performance obtained in [22].

## 6.3 Chemical reactor

A model description for the chemical reactor can be found in [20]. Both $H_\infty$ and $H_2$ static controllers are computed with the AL method, which requires solving 18 and 12 tangent subproblems, respectively. The static controllers are given as

$$K_\infty = \begin{bmatrix} -5.92970 & -9.25089 \\ -4.46382 & -19.07289 \end{bmatrix}$$

for the $H_\infty$ performance criterion and

$$K_2 = \begin{bmatrix} 0.35714 & -2.62418 \\ 2.58159 & 0.77642 \end{bmatrix}$$

for the $H_2$ performance criterion.

## 6.4 Piezoelectric actuator

In this section, we consider the design of static controller for a piezoelectric actuator system. A comprehensive presentation of this system can be found in [7] and state-space data are reproduced in the appendix. Static $H_\infty$ and $H_2$ controllers are obtained as

$$K_\infty = [\,-1.54662 \quad -49.60093 \quad -547.65762\,],\ K_2 = [\,-34.93970 \quad -108.04828 \quad -4280.63837\,]\,.$$

The $H_\infty$ controller improves the performance given in [22] by several orders of magnitude.

## 6.5 Dynamic $H_\infty$ controllers

This example is interesting since there does not exist any static controller for the stabilizing problem alone, hence a dynamic controller is required [15]. Without modification, the AL algorithm runs forever when a controller with prescribed order does not exist. In order to force termination in this situation, we allow a maximum number, say 150, of tangent subproblems. When this limit is reached, this is interpreted as infeasibility of the problem or failure in the AL algorithm to compute a satisfactory controller.

The AL algorithm has been used to design a first-order and a second-order controller. The corresponding state-space data are the following:

$$A_K = -49.22999,\ B_K = 40.30850,\ C_K = -51.54691,\ D_K = 43.13098\,.$$

$$A_K = \begin{bmatrix} -26.61064 & 8.84372 \\ 7.73264 & -2.67573 \end{bmatrix},\ B_K = \begin{bmatrix} 18.80730 \\ -4.44757 \end{bmatrix},\ C_K = [\,-30.21888 \quad 9.03326\,],\ D_K = 21.44923\,.$$

As shown in Table 1, the AL algorithm was able to compute a first-order stabilizing controller, but the associated $H_\infty$ performance significantly deteriorates compared to a full-order controller. In contrast, if one allows second-order controllers, the achieved performance is globally optimal. This again suggests, as with similar experiments conducted in [27] and [2], that the AL algorithm often yields globally optimal solutions.

| problem | quad. SDP | $\|h\|_\infty$ | $H_\infty$ full/static | $H_2$ full/static |
|---|---|---|---|---|
| Transport airplane | 24 | $4.96e$–$5$ | 1.60/2.22 | - |
| VTOL helicopter $H_\infty$ | 17 | $6.32e$–$5$ | $9.57e$–$2$/$1.57e$–$1$ | - |
| VTOL helicopter $H_2$ | 13 | $3.33e$–$7$ | - | $8.71e$–$2$/$9.541e$–$2$ |
| Chemical reactor $H_\infty$ | 18 | $6.90e$–$5$ | 1.141/1.202 | - |
| Chemical reactor $H_2$ | 12 | $1.71e$–$7$ | - | 1.881/1.937 |
| Piezoelectric actuator $H_\infty$ | 21 | $9.77e$–$5$ | $9.633e$–$5$/$3.055e$–$3$ | - |
| Piezoelectric actuator $H_2$ | 29 | $6.64e$–$6$ | - | $1.923e$–$2$/$3.646e$–$2$ |
| Dynamic 1st-order $H_\infty$ design | 31 | $3.61e$–$6$ | 21.50/60.98 | - |
| Dynamic 2nd-order $H_\infty$ design | 42 | $1.71e$–$7$ | 21.60/21.60 | - |

TABLE 1: Results for static/fixed-order $H_\infty$ and $H_2$ syntheses

# 7 Conclusion

We have proposed an AL method for the fixed-order $H_\infty$ and $H_2$ synthesis problem. As supported by a number of numerical experiments, this is a very robust method provided that it is properly implemented. At the core of the computation, is the minimization of a quadratic cost subject to SDP constraints. This is one complication attached to the use of more elaborate subproblem models capturing second-order information from the original problem. A tailored spectral quadratic SDP technique has been developed under Fortran 90 to carry out these elemental steps efficiently. As emphasized in sections 4 and 5, efficiency is prone to dramatic deteriorations if one simply reformulates the problem to fit current SDP codes. The spectral quadratic SDP technique is more direct and offers scope for further development such as the use of a trust region strategy in place of classical linesearches. As is widely accepted, but has never been experienced for SDP constraints, this alternative strategy should provide both better numerical stability as well as better convergence rates on the inherently nonconvex problems studied in this paper. This promising feature is currently under investigation.

Overall, the AL method might appear quite involved as different computational stages are necessary. We have tried to demonstrate that this is worth the effort both from practical and theoretical viewpoints. See reference [27] for a detailed analysis of the latter. Simpler coordinate descent schemes often fail [19] whereas more elaborate first-order techniques experience numerical difficulties to reach local solutions [2].

Most importantly for control designers, our approach is not only applicable to stabilization (feasibility) problems, but also performance (linear objective minimization) problems, an option which is crucial in applications.

# References


[1] B. Anderson and S. Vongpanitlerd, *Network analysis and synthesis: a modern systems theory approach*, Prentice-Hall, 1973.

[2] P. Apkarian, D. Noll, and H. D. Tuan, *Fixed-order $\mathcal{H}_\infty$ control design via an augmented Lagrangian method* , Int. J. Robust and Nonlinear Control, 13 (2003), pp. 1137–1148.

[3] P. Apkarian and H. D. Tuan, *Concave Programming in Control Theory*, Journal of Global Optimization, 15 (1999), pp. 343–370.

[4] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, USA, Belmont, Mass., 1995.

[5] P. T. Boggs and J. W. Tolle, *Sequential Quadratic Programming*, Acta Numerica, (1995), pp. 1–52.

[6] J. Bonnans, *Local analysis of Newton-type methods for variational inequalitites and nonlinear programming*, Appl. Math. Optim., 29 (1994), pp. 161–186.

[7] B. M. Chen, *$H_\infty$ Control and Its Applications*, vol. 235 of Lectures Notes in Control and Information Sciences, Springer Verlag, New York, Heidelberg, Berlin, 1998.

[8] A. R. Conn, N. Gould, and P. L. Toint, *A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds*, SIAM J. Numer. Anal., 28 (1991), pp. 545 – 572.

[9] A. R. Conn, N. I. M. Gould, A. Sartenaer, and P. L. Toint, *Local convergence properties of two augmented Lagrangian algorithms for optimization with a combination of general equality and linear constraints*, Tech. Rep. TR/PA/93/27, CERFACS, Toulouse, France, 1993.

[10] ——, *Convergence properties of an augmented Lagrangian algorithm for optimization with a combination of general equality and linear constraints*, SIAM J. on Optimization, 6 (1996), pp. 674 – 703.

[11] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods*, MPS/SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[12] B. Fares, P. Apkarian, and D. Noll, *An Augmented Lagrangian Method for a Class of LMI-Constrained Problems in Robust Control Theory*, Int. J. Control, 74 (2001), pp. 348–360.

[13] B. Fares, D. Noll, and P. Apkarian, *Robust Control via Sequential Semidefinite Programming*, SIAM J. on Control and Optimization, 40 (2002), pp. 1791–1820.

[14] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, 1987.

[15] P. Gahinet, *Reliable computation of $H_\infty$ central controllers near the optimum*, Technical Report 13, INRIA, Domaine de Voluceau, Rocquencourt, 78150, Le Chesnay, France, 1992.

[16] P. Gahinet and P. Apkarian, *A Linear Matrix Inequality Approach to $H_\infty$ Control*, Int. J. Robust and Nonlinear Control, 4 (1994), pp. 421–448.

[17] P. Gahinet, A. Nemirovski, A. J. Laub, and M. Chilali, *LMI Control Toolbox* , The MathWorks Inc., 1995.

[18] D. Gangsaas, K. Bruce, J. Blight, and U.-L. Ly, *Application of modern synthesis to aircraft control: Three case studies*, IEEE Trans. Aut. Control, AC-31 (1986), pp. 995–1014.

[19] J. W. Helton and O. Merino, *Coordinate optimization for bi-convex matrix inequalitites*, in Proc. IEEE Conf. on Decision and Control, San Diego, CA, 1997, pp. 3609–3613.

[20] Y. S. Hung and A. G. J. MacFarlane, *Multivariable feedback: A classical approach*, Lectures Notes in Control and Information Sciences, Springer Verlag, New York, Heidelberg, Berlin, 1982.

[21] L. H. Keel, S. P. Bhattacharyya, and J. W. Howze, *Robust control with structured perturbations*, IEEE Trans. Aut. Control, 36 (1988), pp. 68–77.

[22] F. Leibfritz, *Computational design of stabilizing static output feedback controller*, Rapports 1 et 2 Mathematik/Informatik, Forschungsbericht 99-02, Universitt Trier, 1998.

[23] A. Lewis, *Derivatives of spectral functions*, Mathematics of Operations Research, 21 (1996), pp. 576–588.

[24] ——, *Twice differentiable spectral functions*, SIAM J. on Matrix Analysis and Applications, 23 (2001), pp. 368–386.

[25] M. Mesbahi and G. P. Papavassilopoulos, *Solving a Class of Rank Minimization Problems via Semi-Definite Programs, with Applications to the Fixed Order Output Feedback Synthesis*, in Proc. American Control Conf., 1997.

[26] L. Mosheyev and M. Zibulevsky, *Penalty/barrier multiplier algorithm for semidefinite programming*, Optimization Methods and Software, 13 (2000), pp. 235–261.

[27] D. Noll, M. Torki, and P. Apkarian, *Partially Augmented Lagrangian Method for Matrix Inequality Constraints*, To appear in SIAM Journal on Optimization, (2002). Rapport Interne , MIP, UMR 5640, Maths. Dept. - Paul Sabatier University.

[28] A. Shapiro, *On differentiability of symmetric matrix valued functions*, School of Industrial and Systems Engineering, Georgia Institute of Technology, (2002). Preprint.

[29] J. F. Sturm, *Using SeDuMi 1.02, A Matlab Toolbox for Optimization over Symmetric Cones*, tech. rep., Tilburg University, Tilburg, Netherlands, Oct. 2001.

# A Appendix

## A.1 Transport airplane

$$A = \begin{bmatrix} -0.01365 & 0.17800 & 0.00017 & -0.56100 & -0.03726 & 0 & 0.01365 & -0.01311 & 0 \\ -0.01516 & -0.75200 & 1.00100 & 0.00127 & -0.06311 & 0 & 0.01516 & 0.05536 & 0 \\ 0.00107 & 0.07896 & -0.87250 & 0 & -3.39900 & 0 & -0.00107 & -0.00581 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -20 & 10.72000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -50 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.44470 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.44470 & 0.00440 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.00440 & -0.44470 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 \\ 0.94310 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.15500 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -48.82000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 50 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 0.00457 & 0.22649 & -0.02374 & 0 & -0.07297 & 0 & -0.00457 & -0.01667 & 0 \\ 0.70711 & 0 & 0 & 0 & 0 & 0 & -0.70711 & 0 & 0 \end{bmatrix}, \quad D_{12} = \begin{bmatrix} 0.70711 \\ 0.70711 \end{bmatrix}$$

$$C_2 = \begin{bmatrix} 0.00646 & 0.32030 & -0.03358 & 0 & -0.10320 & 0 & -0.00646 & -0.02358 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ -0.01365 & 0.17800 & 0.00017 & -0.56100 & -0.03726 & 0 & 0.01365 & -0.01311 & 0 \\ 0 & -13.58000 & 0 & 13.58000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$D_{21} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad D_{11} = 0, \; D_{22} = 0\,.$$

## A.2 VTOL helicopter

$$A = \begin{bmatrix} -0.0366 & 0.0271 & 0.0188 & -0.4555 \\ 0.0482 & -1.0100 & 0.0024 & -4.0208 \\ 0.1002 & 0.3681 & -0.7070 & 1.4200 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0.0468 & 0 \\ 0.0457 & 0.0099 \\ 0.0437 & 0.0011 \\ -0.0218 & 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0.4422 & 0.1761 \\ 3.5446 & -7.5922 \\ -5.5200 & 4.4900 \\ 0 & 0 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 1.41421 & 0 & 0 & 0 \\ 0 & 0.70711 & 0 & 0 \end{bmatrix}, \; D_{11} = 0, \; D_{12} = \begin{bmatrix} 0.70711 & 0 \\ 0 & 0.70711 \end{bmatrix}$$

$$C_2 = [\,0 \quad 1 \quad 0 \quad 0\,], \; D_{21} = 0, \; D_{22} = 0\,.$$

## A.3 Chemical reactor

$$A = \begin{bmatrix} 1.3800 & -0.2077 & 6.7150 & -5.6760 \\ -0.5814 & -4.2900 & 0 & 0.6750 \\ 1.0670 & 4.2730 & -6.6540 & 5.8930 \\ 0.0480 & 4.2730 & 1.3430 & -2.1040 \end{bmatrix},\ B_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ B_2 = \begin{bmatrix} 0 & 0 \\ 5.6790 & 0 \\ 1.1360 & -3.1460 \\ 1.1360 & 0 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},\ D_{11} = 0,\ D_{12} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$C_2 = \begin{bmatrix} 1 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 \end{bmatrix},\ D_{21} = 0,\ D_{22} = 0.$$

## A.4 Piezoelectric actuator

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -274921.63009 & -73.29154 & -274921.63009 & 0 & 0 \\ 0 & 0 & -0.95970 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} 0 & 0 \\ -274921.63009 & 0 \\ 0 & 0 \\ 0 & -1 \\ 0 & 0 \end{bmatrix},\ B_2 = \begin{bmatrix} 0 \\ 2256831.56417 \\ -3.3956e{-}7 \\ 0 \\ 0 \end{bmatrix}$$

$$C_1 = [\,0 \quad 0 \quad 0 \quad 0 \quad 1\,],\ D_{11} = 0,\ D_{12} = 0$$

$$C_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},\ D_{21} = 0,\ D_{22} = 0\,.$$

## A.5 Fixed $H_\infty$ 2nd-order controller

$$A = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 1 & -1 \\ 0 & 1 & -2 \end{bmatrix},\ B_1 = \begin{bmatrix} 1 & 2 & 0 \\ 0 & -1 & 0 \\ 1 & 1 & 0 \end{bmatrix},\ B_2 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix},\ D_{11} = 0,\ D_{12} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$C_2 = [\,0 \quad -1 \quad 1\,],\ D_{21} = [\,0 \quad 0 \quad 1\,],\ D_{22} = 0,\,.$$