Fixed-Order H_∞ Control Design via a Partially Augmented Lagrangian Method

Pierre Apkarian [*]   *Dominikus Noll* [‡]   *Hoang Duong Tuan* [¶]

Abstract

In this paper we develop an augmented Lagrangian method to determine local optimal solutions of the reduced- and fixed-order H_∞ synthesis problems. We cast these synthesis problems as optimization programs with a linear cost subject to linear matrix inequality (LMI) constraints along with nonlinear equality constraints representing a matrix inversion condition. The special feature of our algorithm is that only equality constraints are included in the augmented Lagrangian, while LMI constraints are kept explicitly in order to exploit currently available semi definite programming (SDP) codes. The step computation in the tangent problem is based on a Gauss-Newton model, and a specific line search and a first-order Lagrange multiplier update rule are used to enhance efficiency. A number of computational results are reported and underline the strong practical performance of the algorithm.

Keywords: Linear matrix inequalities, fixed-order synthesis, reduced-order synthesis, rank constraints, robust synthesis, semi definite programming, augmented Lagrangian method.

1 Introduction

Algebraically or rank-constrained LMI problems frequently arise in control engineering applications. Two prominent examples of this type are fixed and reduced-order synthesis of output feedback controllers. While the present paper is mainly concerned with these problems, we mention that our solution strategies apply to many other practical problems in control (see e.g. [19, 14, 13]).

The goal of the paper is to obtain an iterative technique which allows to compute solutions of the fixed-order H_∞ synthesis problem. Here, *solution* means a locally optimal solution, that is, a reduced-order controller which stabilizes the plant and induces a locally minimal H_∞ norm for the performance channel. Reasonably good procedures to achieve this goal are the Augmented Lagrangian (AL) algorithm and the Sequential Quadratic Programming (SQP) algorithm. AL and SQP are well-known in the context of mathematical programming with classical equality and inequality constraints. There these techniques have well-established convergence properties. Global

[*]ONERA-CERT, Centre d'études et de recherche de Toulouse, Control System Department, 2 av. Edouard Belin, 31055 Toulouse, FRANCE - Also with Maths. Dept. of Université Paul Sabatier - Email : `apkarian@cert.fr` - Tel : +33 5.62.25.27.84 - Fax : +33 5.62.25.27.64.

[‡]Université Paul Sabatier, Mathématiques pour l'Industrie et la Physique, 118, route de Narbonne, UMR 5640, 31062 Toulouse, FRANCE - Email : `noll@mip.ups-tlse.fr` - Tel : +33 5.61.55.86.22 - Fax : +33 5.61.55.83.85.

[¶]Department of Control and Information, Toyota Technological Institute, Hisakata 2-12-1, Tenpaku, Nagoya 468-8511, JAPAN - Email: `tuan@toyota-ti.ac.jp` -Tel +81 52.809.1815 - Fax +81 52.809.1721

convergence for AL is for instance proved in [7], for SQP in [4]. According to the optimization terminology, *global* refers to convergence toward a locally optimal solution from an arbitrary, even remote, starting point. The convergence rates of AL and SQP in a neighborhood of a local optimal solution are also known; see [9, 8] for classical constraints.

The situation is less well understood when matrix inequality constraints, that is, LMIs, BMIs, etc arise. Even for the simplest case of LMIs, so far no convergence proofs have been established for the AL algorithm. The approaches cited above depend heavily on the polyhedral structure of the constraints and do not carry over to matrix inequalities.

Recently, we have established local convergence of the SQP algorithm for matrix inequality constraints [14]. In a yet unpublished manuscript [20], to be submitted shortly, we have proved global convergence of the *Augmented Lagrangian Method for Matrix Inequality Constraints* as used in the present paper. Proving convergence of these methods is important since an increasing number of problems in control lead to optimization problems with matrix inequality constraints. The discussion in [20] is mostly of general and theoretical nature whereas we address here a number of practical features leading to a more effective and reliable implementation of the AL algorithm in the specific setting of fixed-order H_∞ synthesis. Specifically, we pay special attention to the Newton step via Gauss-Newton approximation, exact line search, multiplier update rule in section 4. Numerical examples are presented in section 5.

2 A motivating example

In this introductory section we shall look at the static output feedback stabilization problem as a particularly motivating example. The fixed-order H_∞ synthesis problem will be examined in section 3.

Without additional performance embroideries, the static output feedback stabilization problem requires finding a pair of symmetric matrices (X, Y) such that

$$XY = I, \qquad x := (X, Y) \in \mathcal{X} \tag{1}$$

where x is the decision vector gathering the decision variables in X and Y, and where $\mathcal{X}$ denotes a convex set of LMI constraints on the matrices X, Y, which will be given in detail in section 3.

The feasibility problem (1) is inherently nonconvex, and one possible numerical approach is to minimize a suitable norm expression $\|XY - I\|$ subject to the LMI constraint $x \in \mathcal{X}$. A popular strategy based on this idea (considered successful by various authors) is the Frank & Wolfe or *conditional gradient* algorithm [17], which is tested and analyzed in [11]. We refer the reader to [3] for a standard textbook on the subject. Despite the promising statistics presented in [11], the conditional gradient algorithm has been reported to fail on many practical problems.

At a closer look, it is fairly easy to generate examples where the conditional gradient method fails, simply by restricting the achievable stability degree of the system. In the event of a failure, as a rule the LMI feasibility regions are "small" and ill-conditioned, and the first-order algorithm is not at ease. A thorough analysis of the breakdown of the conditional gradient algorithm was given by Dunn in [12] for polyhedral sets, but the findings remain correct for LMI constrained regions.

A rather typical situation encountered when solving (1) is displayed in Figure 1, where we have plotted the error norm $\|XY - I\|_F$ as a function of the iteration index. The dotted line shows that the Frank and Wolfe algorithm reaches a small norm level after a few iterations, but gets stalled on this plateau forever. The designer is then facing the question: *Does this plateau level provide a true and reliable solution of the problem?* From a theoretical point of view, the conditional gradient algorithm does not provide any satisfactory convergence theory, which would support this decision. Nonetheless, in some cases, the attained plateau level turns out sufficiently good to reconstruct valid controller gains from X and Y.

Here we propose a much more reliable algorithmic approach to problem (1), which uses either the partial AL method developed in [13, 20], or the successive semi definite programming (SSDP) technique discussed in [14]. Both techniques are of second-order type in the sense that gradient steps are replaced with Newton type search steps, which exploit duality with respect to either nonlinear equality constraints or both equality and LMI constraints in order to improve and secure convergence. From the point of view of robustness we prefer the AL method, as it is easier to implement and to combine with currently available SDP solvers than the SSDP method, at the cost of a somewhat slower speed.

Important advantages of both techniques are that they guarantee convergence to a local solution satisfying the algebraic constraint $XY - I = 0$ in the limit [20, 15] under fairly standard hypothesis. And secondly that they converge linearly in the worst case. The more sophisticated SSDP has been proven to converge locally superlinearly in [14].

An important aspect of our approach is that it extends to performance minimization problems under rank and LMI constraints. This class of problems can be cast as

$$\begin{array}{ll} \text{minimize} & c'x \\ \text{subject to} & x \in \mathcal{X} \\ & h(x) = 0\,, \end{array}$$

where as before $\mathcal{X}$ is an LMI constraint set, and $h(x) = 0$ encodes a rank or an algebraic constraint. The fixed-order H_∞ control is a typical instance of this program, to be looked at later in this paper.

A typical behavior of the AL algorithm is displayed in the same Figure 1 (continuous line). We observe that this time the algorithm shows no sign of failure and gradually decreases the error norm to zero. Notice that the feasibility problem in (1) has been reformulated as

$$\begin{array}{ll} \text{minimize} & \|XY - I\|_F^2 \\ \text{subject to} & x \in \mathcal{X} \\ & XY - I = 0\,, \end{array}$$

and is handled through the partially AL function

$$\Phi_c(x, \Lambda) := \|XY - I\|_F^2 + \operatorname{Tr}\left(\Lambda'(XY - I)\right) + c/2\|XY - I\|_F^2\,.$$

Here, Λ is the Lagrange dual variable attached to the equality constraints $XY - I = 0$, and c is a penalty parameter bending the iterates toward the surface $XY - I = 0$. The AL algorithm now requires computing a sequence x^k of minimizers of $\Phi_c(\cdot, \Lambda)$ for an increasing sequence of penalty parameters $c = c^k$ and suitably updated multiplier estimates $\Lambda = \Lambda^k$.

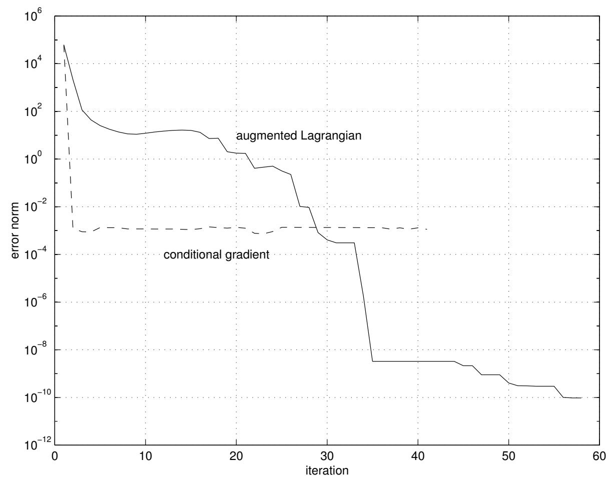

FIGURE 1: Error norms in conditional gradient and augmented Lagrangian

3 Fixed-order H_∞ synthesis

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

$$P(s): \qquad \begin{bmatrix} \dot{x} \\ z \\ y \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix}, \tag{2}$$

where $x \in \mathbf{R}^n$ is the state vector, $u \in \mathbf{R}^{m_2}$ is the vector of control inputs, $w \in \mathbf{R}^{m_1}$ is a vector of exogenous inputs, $y \in \mathbf{R}^{p_2}$ is the vector of measurements and $z \in \mathbf{R}^{p_1}$ is the controlled or performance vector.

Let $T(s)$ denote the closed-loop transfer functions from w to z for some dynamic output-feedback control law $u = K(s)y$. Our aim is to compute a k-th order output-feedback controller

$$K(s) = C_K(sI - A_K)^{-1}B_K + D_K, \qquad A_K \in \mathbf{R}^{k\times k}, (k \le n) \tag{3}$$

which meets the following design requirements:

- *internal stability:* for $w = 0$ the state vector of the closed-loop system (2) and (3) tends to zero as time goes to infinity.
- *performance:* the H_∞ norm $\|T(s)\|_\infty$ is minimized.

As is well-known, the closed-loop system can be described as

$$\left[\begin{array}{c|c} A_{c\ell} & B_{c\ell} \\ \hline C_{c\ell} & D_{c\ell} \end{array}\right] = \left[\begin{array}{c|c} A_a + B_a K_a C_a & B_{1,a} + B_a K_a D_{21,a} \\ \hline C_{1,a} + D_{12,a} K_a C_a & D_{11} + D_{12,a} K_a D_{21,a} \end{array}\right] \tag{4}$$

where

$$\begin{array}{llll} K_a := \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix}, & A_a := \begin{bmatrix} A & 0 \\ 0 & 0_k \end{bmatrix} & B_{1,a} := \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, & C_{1,a} := \begin{bmatrix} C_1 & 0 \end{bmatrix} \\ B_a := \begin{bmatrix} 0 & B_2 \\ I_k & 0 \end{bmatrix}, & C_a := \begin{bmatrix} 0 & I_k \\ C_2 & 0 \end{bmatrix}, & D_{12,a} := \begin{bmatrix} 0 & D_{12} \end{bmatrix}, & D_{21,a} := \begin{bmatrix} 0 \\ D_{21} \end{bmatrix}. \end{array} \tag{5}$$

With these ingredients, the fixed-order H_∞ 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 A_K, B_K, C_K, D_K from the cast. As a result, the Bounded Real Lemma matrix inequality reduces to the following set of conditions: *Find symmetric matrices X and Y in $\mathcal{S}^{n+k}$ such that*

$$\begin{aligned}
\mathcal{N}_Q' \begin{bmatrix} A_a'X + XA_a & XB_{1,a} & C_{1,a}' \\ B_{1,a}'X & -\gamma I & D_{11}' \\ C_{1,a} & D_{11} & -\gamma I \end{bmatrix} \mathcal{N}_Q &< 0 \qquad (6) \\
\mathcal{N}_P' \begin{bmatrix} YA_a' + A_aY & B_{1,a} & YC_{1,a}' \\ B_{1,a}' & -\gamma I & D_{11}' \\ C_{1,a}Y & D_{11} & -\gamma I \end{bmatrix} \mathcal{N}_P &< 0 \qquad (7) \\
X > 0, \qquad Y > 0, \qquad XY - I &= 0 \qquad (8)
\end{aligned}$$

where $\mathcal{N}_Q$ and $\mathcal{N}_P$ denote any bases of the nullspaces of Q and P with

$$Q := [\,C_a \quad D_{21,a} \quad 0\,], \qquad P := [\,B_a' \quad D_{12,a}' \quad 0\,]\,.$$

□

The reader is referred to [16, 2] for proofs and further details. Notice that according to [17], it is possible to replace the positive definiteness constraints in (8) with

$$\begin{bmatrix} X & I \\ I & Y \end{bmatrix} \geq 0\,, \qquad (9)$$

without loss of generality. This option helps stabilizing algorithms and eliminates spurious solutions.

4 Augmented Lagrangian with explicit LMI constraints

In this section, we present our approach to finding local solutions of the fixed-order H_∞ synthesis problem. The problem is recast as an optimization problem using a cost function which combines the L_2-gain index γ and a penalty term accounting for the nonlinear constraint in (8), attributing a high cost to infeasible points. The LMI constraints, being different in nature, are not included in the objective but kept explicitly.

For future use and simplicity of manipulations, let $\mathcal{X}$ denote the set determined by the LMI constraints in (6), (7) and (9), where < 0 has been replaced with ≤ 0, or if we wish, with $\leq -\varepsilon I$ for a small threshold $\varepsilon > 0$ in order to guarantee strict feasible solutions to the LMIs. The vector x regroups the ensemble of decision variables (X, Y, γ) and should not be confused with the state vector in (2). With these notations, the fixed-order H_∞ synthesis problem is equivalent to :

$$\begin{array}{ll} \min & f(x) := \gamma \\ \text{s.t.} & h(x) := XY - I = 0\,, \\ & x \in \mathcal{X}\,. \end{array} \qquad (10)$$

with $x \in \mathbf{R}^N$, $N = (n+k)(n+k+1)+1$, $h : \mathbf{R}^N \to \mathbf{R}^M$, $M = (n+k)^2$, and where the LMI size L is generically given as $L = 2(2n + k + m_1 + p_1) - m_2 - p_2$.

Following the general idea of the AL method, the key in solving (10) is now to eliminate the non-convex constraints $h(x) = 0$ in (8) by including them into a partially AL function. This allows us to approximate the difficult non-convex synthesis problem by a two-stage series of easier LMI subproblems, as we proceed to indicate: The non-convex problem (10) is approximated by a series of new optimization problems, each of which involves minimizing the AL function, $\Phi_c(x, \Lambda)$, defined as:

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

subject to the LMI constraints $x \in \mathcal{X}$. In matrix form, the new objective is:

$$\Phi_c(x, \Lambda) = \gamma + \mathrm{Tr}\Big(\Lambda'(XY - I)\Big) + \frac{c}{2}\mathrm{Tr}\Big((XY - I)'(XY - I)\Big)\,, \tag{11}$$

where c is a positive penalty and Λ is a Lagrange multiplier matrix. Each of the new optimization problems

$$\begin{array}{ll} \text{minimize} & \Phi_c(x, \Lambda) \\ \text{subject to} & x \in \mathcal{X} \end{array} \tag{12}$$

is by itself solved by a sequence of SDPs. 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 $x + dx \in \mathcal{X}$.

It is important to keep in mind that the motivation for using the AL is that, for an appropriate, fixed choice of (Λ^*, c^*), a local optimal solution x^* of the original program (10) can be found by simply optimizing the function $\Phi_{c^*}(x, \Lambda^*)$ with respect to x. A thorough discussion on this mechanism is given in [15]. Of course, the central task is to determine (Λ^*, c^*), and the AL algorithm achieves this goal by forming a sequence (Λ^k, c^k) converging to (Λ^*, c^*).

Since the proposed algorithm is of second-order type, we need to compute the gradient and Hessian of (11). 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 Lagrangian $\nabla_x \Phi_c(x, \Lambda)$ is computed as

$$\nabla_x \Phi_c(x, \Lambda) = \begin{bmatrix} T'\mathrm{vec}\,(\Lambda Y) \\ T'\mathrm{vec}\,(X\Lambda) \\ 1 \end{bmatrix} + c\, J(x)'\mathrm{vec}\,(XY - I)\,. \tag{13}$$

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

$$\nabla^{GN}_{xx}\Phi_c = \begin{bmatrix} 0 & T'(I \otimes \Lambda)T & 0 \\ T'(I \otimes \Lambda')T & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + c\, J(x)'J(x)\,. \tag{14}$$

By definition it is obtained by omitting the term

$$c\sum_{i=1}^{M} h_i(x)\nabla^2_{xx} h_i(x) \tag{15}$$

from the full Hessian expansion $\nabla^2_{xx}\Phi_c(x,\Lambda)$. The rationale in this approximation, often referred to as Gauss-Newton approximation, is that (15) is small when the iterates x get close to feasibility, and the terms $h_i(x)$ 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 (15).

Notice that due to the inherent non-convexity of the problem, neither $\nabla^2_{xx}\Phi_c$ nor $\nabla^{GN}_{xx}\Phi_c$ can be expected to be positive (semi) definite. According to the experience with optimizers in traditional nonlinear programming gained during recent years, it is considered best to include the negative curvature information in $\nabla^2_{xx}\Phi_c$ or $\nabla^{GN}_{xx}\Phi_c$ into the search strategies by using trust region steps [22, 10]. Unfortunately, in our case, this would lead to an LMI constrained tangent problem with a nonconvex quadratic objective, a problem with a complexity similar to the original program. As opposed to the classical case, where the tangent problem is an indefinite quadratic program, for which tailored solution strategies exist, this new type of tangent problem presently does not lend itself to efficient algorithmic solutions, and we are therefore forced to privilege an older strategy, viz. to convexify the true or Gauss-Newton Hessian $\nabla^{GN}_{xx}\Phi_c$ in order to obtain an SDP tangent problem. Our proposal of a potential convexifying technique is discussed in section 4.1.

With these preparations, a general description of the algorithm is now the following.

Algorithm for Fixed-Order H_∞ Synthesis

1. Initial phase . Initialize the algorithm with $x^0 \in \mathcal{X}$. This can be done by simply solving a feasibility SDP. Then initialize the penalty parameter $c^0 > 0$ and the Lagrange multiplier Λ^0. Fix $\rho < 1$, $0 < \mu < 1$ and $\varepsilon > 0$.

2. Optimization phase. For $j = 0, 1, \cdots$, minimize $\Phi_{c^j}(x, \Lambda^j)$ over $x \in \mathcal{X}$, and let x^{j+1} be the solution so obtained. Possibly use the previous iterate x^j as a starting value for the inner optimization.

3. Update penalty and multiplier.

$$\Lambda^{j+1} = \Lambda^j + c^j(X^{j+1}Y^{j+1} - I). \tag{16}$$

$$c^{j+1} = \begin{cases} \rho c^j & \text{if } \|X^{j+1}Y^{j+1} - I\|_F > \mu\|X^jY^j - I\|_F \\ c^j & \text{if } \|X^{j+1}Y^{j+1} - I\|_F \leq \mu\|X^jY^j - I\|_F \end{cases} \tag{17}$$

4. Terminating phase . If $\|XY - I\|_F < \varepsilon$, try to reconstruct a k-th-order H_∞ controller. If the reconstruction fails, reduce ε, set $j = j + 1$ and return to **Step 2**.

4.1 Inner steps

The inner steps in our algorithm involve minimizing the AL $\Phi_c(\cdot, \Lambda)$ for fixed c and Λ. This minimization is performed iteratively by generating search directions dx about the current iterate

x through the tangent model

$$\begin{array}{ll} \min & \nabla\Phi_c(x,\Lambda)'\,dx + \frac{1}{2}\,dx'\,H\,dx \\ \text{s.t.} & x + dx \in \mathcal{X} \end{array}$$

As indicated before, here we have replaced the true Hessian $\nabla^2_{xx}\Phi_c$ by its Gauss-Newton approximation $H = \nabla^{GN}_{xx}\Phi_c$, or more precisely by a strictly convex approximation of $\nabla^{GN}_{xx}\Phi_c$ obtained e.g. by further adding a positive diagonal correction term. Due to the convexity of H, the resulting tangent problem may now be handled as a general conic programming problem [23] or as an SDP via the equivalent formulation

$$\begin{array}{ll} \min & t \\ \text{s.t.} & \begin{bmatrix} t - \nabla\Phi_c(x,\Lambda)'\,dx & dx'\,L \\ L'\,dx & 2\,I \end{bmatrix} \succeq 0 \\ & x + dx \in \mathcal{X}\,, \end{array}$$

where L Cholesky factorizes H, that is, $H = L\,L'$.

From a practical point of view, instead of modifying the Hessian matrix directly and computing the Cholesky factor of the resulting matrix, we find it more reliable and efficient to compute its symmetric indefinite factorization

$$\nabla^{GN}_{xx}\Phi_c = P'LDL'P$$

where P is a permutation matrix, L a unit lower triangular matrix and D is block diagonal with diagonal blocks of dimension 1 or 2. We then trivially modify the diagonal blocks of D such that $D + E$ is positive definite, finally letting $H = P'L(D+E)L'P$. The advantage of this approach is that it can be accomplished at roughly the cost of a customary Cholesky factorization [6].

4.2 Line search

The search direction computation just discussed must be followed by an appropriate line search to ensure convergence of the method. In our case, because of the polynomial nature of the AL in (11), the step length computation can be performed *exactly* at almost no cost, using a simple third-order polynomial root computation routine. Indeed, within the search interval $[x,\ x+dx]$, the AL is a fourth-order polynomial

$$\Phi_c(x + \alpha\,dx, \Lambda) = a_4\alpha^4 + a_3\alpha^3 + a_2\alpha^2 + a_1\alpha + a_0\,.$$

Its minimum on the interval $[0,\ 1]$ is either attained at 0, 1 or at values of α for which the derivative of $\phi(\alpha) := \Phi_c(x + \alpha\,dx, \Lambda)$ vanishes, that is,

$$4a_4\alpha^3 + 3a_3\alpha^2 + 2a_2\alpha + a_1 = 0\,.$$

4.3 First-order multiplier updates

The multiplier update $\Lambda^+ = \Lambda + c\,h(x)$ in (16) is known as the first-order multiplier update rule. It is an essential ingredient to establish convergence of the AL algorithm. An in-depth discussion of the multiplier update rule and its importance regarding the convergence of the AL is provided in [20].

5 Numerical illustrations

In this section, we provide a catalog of results that have been obtained using the AL algorithm. The reader is referred to [20] for additional illustrations of the AL algorithm.

From now on, we will use the notation

$$\left[\begin{array}{c|c|c} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ \hline C_2 & D_{21} & D_{22} \end{array}\right]$$

to refer to the state-space data of the standard form. The experiments reported below have been performed using the algorithm parameters:

$$\varepsilon = 10^{-5},\ \rho = 4.0,\ \mu = 0.2\,.$$

5.1 Randomly generated problems

A preliminary assessment of the method can be carried out using full-order controllers. In this case, there exist well established techniques to compute the globally optimal H_∞ performance [18, 21, 16], and this allows us to check the quality of the gain γ obtained by our algorithm. It is instructive to verify that our method is still efficient, even though one disregards the hidden convexity in the constraint set specified by (6), (7), (9) in tandem with the inversion constraint $XY - I = 0$. In these experiments, we have randomly generated 1000 stabilizable and detectable plants as described in [11]. Dimensions of our tests are given as

$$\begin{array}{lll} n = \{3,\, 6,\, 10\}\,, & m_1 = \{3,\, 7\}\,, & m_2 = \{1,\, 2,\, 4\}\,, \\ p_1 = \{3,\, 7\}\,, & p_2 = \{1,\, 2,\, 4\}\,. & \end{array}$$

Remarkably enough, our method not only never failed, but also invariably returned the optimal γ within machine accuracy for all tests.

As one may argue that the hidden convexity in this type of test problems alleviates the difficulty, we performed similar tests for the reduced-order case with the problem generation technique outlined in [11]. Again no failure in computing a stabilizing controller was observed, and the method performed very efficiently in computing a locally optimal cost. As an important side aspect, notice that our method is apparently insensitive to the reachable stability degree in the system, a major source of breakdown in first-order or gradient-based methods, as outlined in the introductory section 2.

5.2 Comparison with a randomized algorithm

Reference [5] proposes a randomized algorithm to compute a first-order H_∞ controller for the augmented plant described by the following state-space data:

$$\left[\begin{array}{cccccccc|ccc|cc}
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1.5 & -1.5 & 0 & 0.0057 & 1.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.16 & 0.8 \\
-12 & 12 & -0.6 & -0.0344 & -12 & 0 & 0 & 0 & 0 & 0 & 0 & -19 & -3 \\
-0.852 & 0.29 & 0 & -0.014 & -0.29 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0115 & -0.0087 \\
0 & 0 & 0 & 0 & -0.73 & 2.8289 & 0 & 0 & 0.1146 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1.25 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1000 & 0 & 0 & 1024 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1000 & 0 & 0 & 1024 & 0 & 0 \\
\hline
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.01 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.01 \\
\hline
1 & 0 & 0 & 0 & 0 & 0 & -139.0206 & 0 & 0 & 142.8571 & 0 & 0 & 02 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & -139.0206 & 0 & 0 & 142.8571 & 0 & 0
\end{array}\right]$$

Using their technique, a first-order contoller with H_∞ performance $\gamma = 4.8937$ was obtained. In contrast, our AL method achieved a performance of $\gamma = 1.821$, a result which required solving 46 SDPs, and corresponded to a performance improvement of 62%. The optimal first-order controller we obtained is

$$\left[\begin{array}{c|cc} -27.617 & 0.23275 & -0.60698 \\ \hline 65.735 & 0.46106 & -0.26079 \\ 406.8 & 0.69252 & -0.56237 \end{array}\right]$$

Here the problem dimensions were $N = 91$, $M = 81$ and $L = 44$.

Evolution of the error norm $\|XY - I\|_F$, the gain γ and the penalty parameter c during the iterations are displayed in Figure 2. While the penalty parameter was forced to increase or be stable, the other quantities were allowed to vary freely and to grow locally until a balance between the cost value and the error norm was attained. As described in section 4, the penalty parameter is kept constant during inner steps.

A remarkable feature of our algorithm is that a good guess of the optimal gain γ is already obtained at an early stage of the iteration process, while later iterations serve to satisfy the equality constraints at almost constant cost.

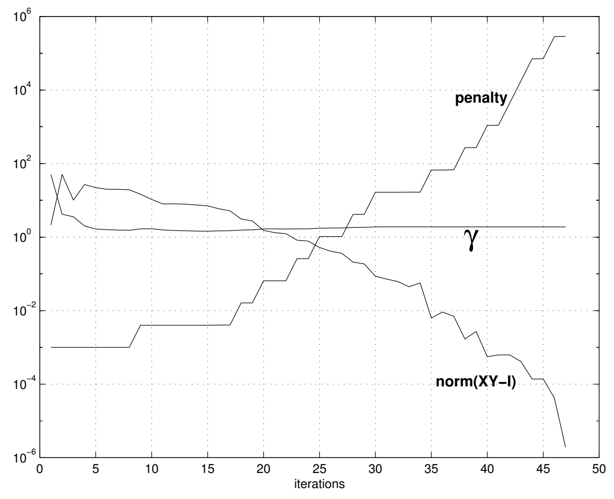

FIGURE 2: AL method: error norm, γ, penalty

6 Conclusion

In this paper, an AL method has been developed to solve the fixed-order H_∞ synthesis problem. Implementation of this method involves several practical algorithmic issues which have been discussed. Four important features of the proposed AL are: (i) LMIs are not incorporated into the AL but kept explicitly throughout the iterations; (ii) we consider a Gauss-Newton approximation of the full Hessian; (iii) an exact line search is used; (iv) a first-order rule is used for the equality multipliers. An important consequence of (i)-(ii) is that subproblem iterations reduce to simple SDP problems.

As theoretically expected [20], our method shows strong and reliable performance on a number of numerical tests and compares favorably to existing techniques. In addition to its strong convergence properties, 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 and H. D. Tuan, *Concave Programming in Control Theory*, Journal of Global Optimization, 15 (1999), pp. 343–370.

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

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

[5] G. Calafiore, F. Dabbene, and R. Tempo, *Randomized Algorithms for Reduced Order H_∞ Design*, in Proc. American Control Conf., Chicago, Illinois, 2000, pp. 3837–3839.

[6] S. H. Cheng and N. J. Highaam, *A modified cholesky algorithm based on a symmetric indefinite factorization*, SIAM J. on Matrix Analysis and Applications, 19 (1998), pp. 256–268.

[7] 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.

[8] 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.

[9] ——, *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.

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

[11] M. C. de Oliveira and J. C. Geromel, *Numerical Comparison of Output Feedback Design Methods*, in Proc. American Control Conf., Albuquerque, NM, June 1997, pp. 72–76.

[12] J. C. Dunn, *Convergence Rates for Conditional Gradients Sequences Generated by Implicit Step Length Rules*, SIAM J. on Control and Optimization, 18 (1979), pp. 473–487.

[13] 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.

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

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

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

[17] L. E. Ghaoui, F. Oustry, and M. AitRami, *An Algorithm for Static Output-Feedback and Related Problems*, IEEE Trans. Aut. Control, 42 (1997), pp. 1171–1176.

[18] K. Glover and J. C. Doyle, *State-space formulae for all stabilizing controllers that satisfy the H_∞-norm bound and relations to risk sensitivity* , Syst. Control Letters, 11 (1988), pp. 167–172.

[19] 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.

[20] D. Noll, M. Torki, and P. Apkarian, *Partially Augmented Lagrangian Method for Matrix Inequality Constraints*, (2002). submitted to SIAM Journal on Optimization.

[21] A. Packard, K. Zhou, P. Pandey, and G. Becker, *A collection of robust control problems leading to LMI's*, in Proc. IEEE Conf. on Decision and Control, vol. 2, Brighton, Dec. 1991, pp. 1245–1250.

[22] M. Powell, *On the global convergence of trust-region algorithms for unconstrained optimization*, MATH. Programming, 29 (1984), pp. 297–303.

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