H_∞ SYNTHESIS VIA A NONSMOOTH, NONCONVEX OPTIMIZATION APPROACH

MUSA MAMMADOV AND ROBERT ORSI

Abstract: A numerical method for solving the H_∞ synthesis problem is presented. The problem is posed as an unconstrained, nonsmooth, nonconvex minimization problem. The optimization variables consist solely of the entries of the output feedback matrix. No additional variables, such as Lyapunov variables, need to be introduced. The main part of the optimization procedure uses a line search mechanism where the descent direction is defined by a recently introduced dynamical systems approach. Numerical results for various benchmark problems are included in the paper. In addition, the effectiveness of a preliminary part of the algorithm for successfully and quickly finding stabilizing controllers is also demonstrated.

1 Introduction

The H_∞ synthesis problem involves finding an output feedback control matrix K that minimizes the H_∞ norm of a certain transfer function, subject to the constraint that K is stabilizing. This is a challenging problem and even finding a stabilizing K can be difficult. Indeed, if the entries of K are restricted to lie in prescribed intervals, then finding a stabilizing K is an NP-hard problem [6].

Existing numerical methods for the H_∞ synthesis problem are often based on first reformulating the problem into one involving linear matrix inequalities (LMIs) and an additional nonconvex rank constraint or nonconvex equality constraint. Numerical methods for such reformulations of the problem include those based on linearization [9], [17], [20]; alternating projections [13], [14], [26]; augmented Lagrangian methods [3], [4], [10], [25]; and sequential semidefinite programming [11].

The H_∞ synthesis problem can also be reformulating into a problem involving bilinear matrix inequalities (BMIs). Numerical methods for such reformulations of the problem include [11], [18], [22] and [28]. See also the references therein.

A disadvantage of these approaches is that they require the introduction of Lyapunov variables. As the number of Lyapunov variables grows quadratically with the number of state variables, the total number of variables can be quite large and even problems of moderate size can lead to numerical difficulties [2].

In this paper the H_∞ synthesis problem is posed as an unconstrained, nonsmooth, nonconvex minimization problem. The optimization variables for this reformulation consist

solely of the entries of the output feedback matrix K and no additional variables, such as Lyapunov variables, need to be introduced. The approach taken to solve this problem is based on using the recently developed global optimization algorithm presented in [23] and [24]. This optimization algorithm uses a line search mechanism where the descent direction is defined via a dynamical systems approach. It can be applied to a wide range of functions, requiring only function evaluations to work. In particular it does not require gradient (or gradient like) information and hence it is well suited to optimizing our reformulation of the H_∞ synthesis problem.

Similar approaches, that is, ones based on directly minimizing an appropriate nonsmooth function of K, are taken in [7] in addressing various problems of robust stabilization, and in [1] and [2] for the H_∞ synthesis problem. The cost function we use is different to the ones used in these other works, as is our underlying method of optimization.

In [7] when optimizing robust stability and in [1] and [2] when dealing with the H_∞ synthesis problem, a stabilization solution is first sought by trying to solve some auxiliary problem and then optimization is performed locally about this solution. We employ a similar approach in that the main part of the algorithm is preceded by a section designed specifically to find a stabilization solution. Like the main part of the algorithm, this first part is based on decreasing a nonsmooth, nonconvex cost function, though the optimization procedure used is different to the one used in the main part of the algorithm.

The paper is structured as follows. In Section 2 we recall the H_∞ synthesis problem as well as a specialization of this problem, the robust stabilization problem. In Section 3 we reformulate these problems as unconstrained optimization problems in the output feedback matrix K. We also mention some of the issues involved in trying to solve such problems. Section 4 outlines the optimization approach used. Numerical experiments for various H_∞ synthesis and robust stabilization problems are presented in Section 5. In addition, this section contains experiments demonstrating the effectiveness of the preliminary part of the algorithm for successfully and quickly finding stabilizing controllers. The paper ends with some concluding remarks.

2 Problem Formulations

2.1 The H_∞ Synthesis Problem

Recall the *static output feedback H_∞ synthesis problem.*

Problem 1 Given a linear time invariant (LTI) system

$$\begin{bmatrix} \dot{x} \\ z \\ y \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0 \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix}, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^{m_2}$ is the control, $y \in \mathbb{R}^{p_2}$ is the measured output, $w \in \mathbb{R}^{m_1}$ is the external input and $z \in \mathbb{R}^{p_1}$ is the controlled output, find a static output feedback

$$u = Ky$$

such that the H_∞ norm of $T_{w,z}(s, K)$, the closed loop transfer function from w to z, is minimal over the set of K for which $A + B_2 K C_2$ is stable. □

We note that, given a system (1) and a output feedback matrix K, the closed loop dynamics from w to z are given by

$$\begin{bmatrix} \dot{x} \\ z \end{bmatrix} = \begin{bmatrix} A + B_2 K C_2 & B_1 + B_2 K D_{21} \\ C_1 + D_{12} K C_2 & D_{11} + D_{12} K D_{21} \end{bmatrix} \begin{bmatrix} x \\ w \end{bmatrix}.$$

As is well known, the *dynamic output feedback H_∞ synthesis problem* can be posed as a static output feedback problem for an augmented system. Indeed, for a given system (1), suppose we would like to find an order $k \le n$ dynamic controller of the form

$$\begin{bmatrix} \dot{x}_K \\ u \end{bmatrix} = \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} \begin{bmatrix} x_K \\ y \end{bmatrix}.$$

Here $x_K \in \mathbb{R}^k$. Then the dynamic output feedback H_∞ synthesis problem is equivalent to Problem 1 with the following substitutions:

$$K \to \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix}, \; A \to \begin{bmatrix} A & 0 \\ 0 & 0_k \end{bmatrix}, \; B_1 \to \begin{bmatrix} B_1 \\ 0 \end{bmatrix},$$

$$B_2 \to \begin{bmatrix} 0 & B_2 \\ I_k & 0 \end{bmatrix}, \; C_1 \to \begin{bmatrix} C_1 & 0 \end{bmatrix}, \; C_2 \to \begin{bmatrix} 0 & I_k \\ C_2 & 0 \end{bmatrix},$$

$$D_{12} \to \begin{bmatrix} 0 & D_{12} \end{bmatrix}, \; D_{21} \to \begin{bmatrix} 0 \\ D_{21} \end{bmatrix}.$$

0_k and I_k denote the $k \times k$ zero and identity matrices respectively. Note that K which was $m_2 \times p_2$ has been replaced by a matrix of dimension $(k + m_2) \times (k + p_2)$.

2.2 The Robust Stabilization Problem

Before introducing the robust stabilization problem, we present some preliminaries.

If X is a square matrix, let $\alpha(X)$ denote the maximum of the real parts of the eigenvalues of X,

$$\alpha(X) := \max_i \text{Re}(\lambda_i(X)).$$

Of course, X is stable if and only if $\alpha(X) < 0$.

For $X \in \mathbb{C}^{n \times n}$, let $\beta(X)$ denote its *complex stability radius* [15],

$$\beta(X) := \min\{\|E\| \mid E \in \mathbb{C}^{n \times n}, \alpha(X + E) \ge 0\}.$$

Here $\|\cdot\|$ denotes the maximum singular value norm, $\|E\| = \sigma_{\max}(E)$. $\beta(X)$ is zero if and only if X is unstable. The complex stability radius of a stable matrix X determines how robust the stability of X is with respect to additive (complex) perturbations of X. For any X, $\beta(X)$ gives the distance to the unstable matrices.

The *robust stabilization problem* is the following.

Problem 2 Given a linear time invariant (LTI) system

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control, and $y \in \mathbb{R}^p$ the output, find a static output feedback control law

$$u = Ky$$

that maximizes the complex stability radius of the closed loop system matrix $A + BKC$:

$$\max_{K \in \mathbb{R}^{m \times p}} \beta(A + BKC).$$

$\square$

Problem 2 is a special case of Problem 1, as we now show. Suppose X is a stable matrix with associated transfer function $H(s) := (sI - X)^{-1}$. Then $\beta(X)$ and the H_∞ norm of H are related by

$$\beta(X) = \|H(s)\|_\infty^{-1}.$$

As a result, Problem 2 is equivalent to minimizing the H_∞ norm of the transfer function $(sI - (A + BKC))^{-1}$ subject to the constraint that $A + BKC$ is stable. Given A, B and C as in Problem 2, taking the same A, $B_1 = I$, $B_2 = B$, $C_1 = I$, $C_2 = C$, $D_{11} = 0$, $D_{12} = 0$ and $D_{21} = 0$, it can be readily shown that Problem 1 reduces to Problem 2.

3 A Nonsmooth, Nonconvex Optimization Problem

Using the terminology of Problem 1, define

$$f(K) := \begin{cases} -\|T_{w,z}(s,K)\|_\infty^{-1}, & \text{if } \alpha(A + B_2 K C_2) < 0, \\ \alpha(A + B_2 K C_2), & \text{if } \alpha(A + B_2 K C_2) \geq 0. \end{cases} \tag{2}$$

The main idea behind our approach is to try to solve Problem 1 by trying to solve the following unconstrained minimization problem:

$$\min_{K \in \mathbb{R}^{m_2 \times p_2}} f(K).$$

Our motivation for choosing this particular objective function is as follows. The set of stabilizing K's is $\{K \mid \alpha(A + B_2 K C_2) < 0\}$ and our aim is to minimize $\|T_{w,z}(s,K)\|_\infty$ over this set. Finding a K that minimizes $\|T_{w,z}(s,K)\|_\infty$ is the same as finding a K that minimizes $-\|T_{w,z}(s,K)\|_\infty^{-1}$. However, using $-\|T_{w,z}(s,K)\|_\infty^{-1}$ has the following advantage. Within the stabilizing set, $-\|T_{w,z}(s,K)\|_\infty^{-1}$ is negative and converges to zero if $\alpha(A + B_2 K C_2)$ converges zero. It follows that f is a continuous function of K, that is a globally defined extension of $-\|T_{w,z}(s,K)\|_\infty^{-1}$. (Note that $\|T_{w,z}(s,K)\|_\infty$ does not have a useful continuous extension as it becomes unbounded as K goes to the boundary of the set of stabilizing K's.) Furthermore, f penalizes non-stabilizing K's.

The fact that $\|T_{w,z}(s,K)\|_\infty$ can only be evaluated at stabilizing K's makes minimizing this quantity more difficult. Non-stabilizing K's provide a rather limited amount of information in regard to this objective function. The only information they do provide is the extent to which they are in fact non-stabilizing. This information is given by the quantity $\alpha(A + B_2 K C_2)$, and has been incorporated into f.

The main part of the algorithm, which will be used to minimize f, see Section 4, only needs to be able to evaluate f in order to work. There exist efficient numerical methods for calculating H_∞ norms (and hence for calculating f). We use the Matlab function `hinfnorm`.

We now make some observations regarding the robust stabilization problem. These observations will of course also necessarily tell us something about the more general H_∞ synthesis problem.

If the problem we are considering is actually a robust stabilization problem, i.e., a case of Problem 2, then in the definition of f, the term $-\|T_{w,z}(s, K)\|_\infty^{-1}$ is just $-\beta(A + BKC)$. Both α and $-\beta$ are nonsmooth and nonconvex, and β, but not α, is locally Lipschitz [7]. As noted in [7], lack of convexity means finding a global minimizer of $-\beta$ can be expected to be difficult and lack of smoothness means it is not possible to use standard local optimization methods such as steepest descent and Newton type methods. (Apparently applying such local optimization methods leads to problems at points where the gradient of β is discontinuous.)

Therefore we have a nonsmooth, nonconvex global optimization problem; quite a difficult problem. All other known formulations of the H_∞ synthesis problem, such as those involving rank constrained LMIs, are also nonconvex and global in nature. While to our disadvantage our formulation is nonsmooth, to our advantage we have not had to introduce Lyapunov variables and hence we have a problem formulation in many less variables than we would have otherwise. Here are some additional, particular aspects of the problem that are worth keeping in mind.

As already mentioned, just finding a stabilizing solution can be a challenge in itself. The set of stabilizing K's can be quite small. For example, for the Boeing 767 system considered in Section 5, the following is a stabilizing solution,

$$ K = \begin{bmatrix} -1.7319 & -2.1035e{-}5 \\ 4.5059e{+}1 & 2.1706e{-}4 \end{bmatrix}. $$

Changing the $(1, 2)$ entry of this K by plus or minus 10^{-5} makes the closed loop system unstable. As the feasible region can be quite localized, one would expect that finding such solutions, and moreover finding globally optimal solutions, would be quite difficult. A global search would have to search quite small regions. This may not be feasible. For example, the calculation of a function value can be fairly time consuming; in the Boeing 767 problem, which has 55 states, to calculate the value of β at a point takes approximately 0.35 seconds on a 3 GHz Pentium 4 machine.

As we have already indicated, for Problem 1, the quantity we are interested in minimizing, $\|T_{w,z}(s, K)\|_\infty$, is not defined for all K's. (Problem 2 is similar in that, while β is defined everywhere, it is 0 for all non-stabilizing K's.) In 'ordinary' constrained optimization (see [27] and references therein), it is still possible to evaluate the objective function outside the feasible region. This may be extremely helpful for finding deep local minimizers inside the feasible region. For Problems 1 and 2 we do not have this advantage. In fact, the feasible region, the set of stabilizing K's, cannot even be usefully quantified.

Finally, it is worth mentioning that finding K that minimizes $\alpha(A + BKC)$ is quite different to finding K that minimizes $-\beta(A+BKC)$. In the first case, one seeks to find a K that causes solutions of the closed loop system to decay to zero as quickly as possible. (We are assuming there exist K for which the closed loop system is stable.) No regard is given to how robustly stable $A + BKC$ is with respect to perturbations. In the second case, one optimizes robust stability. While K must stabilize the system, no regard is given to how quickly solutions decay to zero. In other words, in terms of optimality, the behaviors of the functions $\alpha(A + BKC)$ and $-\beta(A + BKC)$ are quite different.

$\boxed{4}$ A Global Optimization Algorithm

Our algorithm has two main parts. The first part is a nonsmooth gradient descent like algorithm for finding an initial stabilizing solution. This part of the algorithm is based on minimizing the function

$$g : \mathbb{R}^{m_2 \times p_2} \to \mathbb{R}, \ \ K \mapsto \alpha(A + BKC) \tag{3}$$

and is described in Subsections 4.1 and 4.2 below.

The second part of the algorithm uses the stabilizing solution found by the first part as a starting point to minimize f. A recently developed global optimization algorithm, AGOP, which is presented in [23] and [24], is used to minimize f. In the remainder of this section we present a step-by-step outline of the algorithm, as well as descriptions of its various components. We start by describing AGOP.

AGOP is designed for solving unconstrained continuous optimization problems. It uses a line search mechanism where the descent direction is defined via a dynamical systems approach. It can be applied to a wide range of functions, requiring only function evaluations to work. In particular it does not require gradient information and can be used to find minima of non-differentiable functions.

Briefly, AGOP works as follows. Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is the function to be minimized. (In our case, f is given by (2).) AGOP must first be given a set of points, say $\Omega = \{x_1, ..., x_q\} \subset \mathbb{R}^n$. Generally, a suitable choice for an initial set of points is the set of vertices of a box centered around $x = 0$. (For minimizing f given by (2), we will actually use a different choice which will be detailed later.) Suppose that $x_\star \in \Omega$ has the smallest cost of the points in Ω, that is, that $f(x_\star) \leq f(x)$ for all $x \in \Omega$. The set Ω and the values of f at each of the points in Ω allow us to generate a dynamical system; see [23] for details. This dynamical system determines a possible descent direction v at the point $x_\star$; again see [23] for details. An inexact line search along this direction provides a new point $\hat{x}_{q+1}$. A local search about $\hat{x}_{q+1}$ is then carried out. This is done using a *direct search* method called *local variation*. This is an efficient local optimization technique that does not explicitly use derivatives and can be applied to nonsmooth functions. A good survey of direct search methods can be found in [19]. Letting x_{q+1} denote the optimal solution of this local search, the set Ω is augmented to include x_{q+1}. Starting with this updated Ω, the whole process can be repeated. The process is terminated when v is approximately 0 (or a prescribed bound on the number of iterations is reached). The solution returned is the current $x_\star$, that is, the point in Ω with the smallest cost. (If f is continuously differentiable then the solution will be a local minima.)

Note that the convex hull of the set of points in the initial Ω is roughly where AGOP looks for a solution. However, because line search segments are not constrained to lie in some prescribed region, during its operation the algorithm may add to Ω points that are not in the convex hull of the original Ω. As a result, the solution produced by the algorithm may not lie in the convex hull of the initial set of points.

In our application of AGOP in this paper, Ω is initially constructed from the stabilizing solution, K^{stab}, found by the first part of the algorithm. The initial Ω consists of K^{stab} and the vertices of a particular box. This box is constructed to contain K^{stab} and to be roughly contained in the set of stabilizing K's. Applying AGOP to Ω gives a new K. This K is then used to construct (a new box and then) a new Ω and this process is repeated a number of times. While AGOP can be applied just once and then the process terminated, the iterative process described each time refines the search area and increases the likelihood that deep

solutions can be found. Of course, the price is an increase in the amount of computation required.

The above description along with the details below, are one way for automatically refining Ω. We do not claim that it is necessarily the best possible. This issue, optimal stopping criteria, and other such matters, are all interesting questions for future investigations.

The algorithm consists of the following steps.

MAIN ALGORITHM:

Step 0. Choose an initial point K^0. $K^0 = 0$ is an appropriate choice if no information is available regarding were an optimal solution might be found.

Step 1. Using K^0 as an initial point, apply a local optimization procedure to the function g given by (3) to find an initial stabilizing solution K^{stab}; see Subsection 4.1.

Step 2. Let $q = 1$ and $K^1 = K^{\mathrm{stab}}$.

Step 3. Create a box $\mathcal{B}$ around K^q; see Subsection 4.3. Let Ω consist of K^q and the vertices of $\mathcal{B}$.

Step 4. Using Ω, apply algorithm AGOP to the objective function f to get a new solution K^{q+1}. If

$$f(K^{q+1}) < f(K^q) - tol$$

then we set $q = q + 1$ and go to step 3. Otherwise the program is terminated.

4.1 Finding an Initial Stabilizing K

In this subsection we describe our method for finding an initial stabilizing controller.

Step 0: (Initialization)
Suppose we are given an initial controller matrix K^0 (if no such K^0 is available, take $K^0 = 0$). Let $\mathcal{B} = \{K \in \mathbb{R}^{m_2 \times p_2} \mid |K_{ij}| \le \rho \text{ for all } i, j\}$ be a given box containing K^0. Set $s = 0$.

Step 1: (Calculate a local descent direction)
For the function g given by (3), calculate a candidate local descent direction L at the point K^s; see Subsection 4.2.

Step 2: (Coarse line search)
Let $\delta > 0$ be a given step size parameter and let N be the smallest positive integer such that $K^s + N\delta L \notin \mathcal{B}$. Set

$$\hat{K} = \underset{K \in \{K^s + l\delta L \mid l = 0, \dots, N\}}{\arg\min} f(K),$$

and

$$K_{\mathrm{left}} = \hat{K} - \delta L \quad \text{and} \quad K_{\mathrm{right}} = \hat{K} + \delta L.$$

Step 3:   (Finer line search)
Let the parameter η be a given positive integer. Set

$$\Delta = \frac{1}{2\eta}(K_{\text{right}} - K_{\text{left}}),$$

$$K^{s+1} = \underset{K \in \{K_{\text{left}} + l\Delta \ \mid \ l = 0, \ldots, 2\eta\}}{\arg\min} f(K),$$

$$K_{\text{left}} = K^{s+1} + \Delta,$$

$$K_{\text{right}} = K^{s+1} - \Delta.$$

If $\max_{ij}|\Delta_{ij}| < \epsilon$, go to Step 4. Otherwise, go to Step 3.

Step 4:   If $s + 1 = S$, go to Step 5. Otherwise, set $s = s + 1$ and go to Step 1.

Step 5: $K^{\text{stab}} = K^S$.

4.2   Finding a Local Descent Direction

Candidate local descent directions for nonsmooth functions can be calculated in number of ways. In this subsection we present our method for finding such a direction for the function g given by (3).

Given a K, a direction L is found by comparing the value of g at nearby points along the coordinate axes. Let $\{E_{ij} \in \mathbb{R}^{m_2 \times p_2} \mid i = 1, \ldots, m_2, \ j = 1, \ldots, p_2\}$ be the standard orthonormal basis for $\mathbb{R}^{m_2 \times p_2}$ and let $\epsilon > 0$ be a given small parameter. L_{ij}, the (i, j) component of L, is calculated as follows. Define

$$a = g(K - \epsilon E_{ij}), \quad b = g(K), \quad c = g(K + \epsilon E_{ij}).$$

Then,

$$L_{ij} = \begin{cases} 0, & \text{if } (a = b = c) \text{ or } (a > b, \ b < c), \\ b - c, & \text{if } (a = b < c) \text{ or } (a \geq b > c) \text{ or } (b > a \geq c), \\ a - b, & \text{if } (a > b = c) \text{ or } (a < b \leq c) \text{ or } (b > c > a). \end{cases}$$

This method of calculating a descent direction is sort of similar to using finite-difference derivative approximation methods for differentiable functions, see for example [5]. The main difference is that right and left differences are considered and a direction chosen based on which gives greatest decrease. (Another difference is that we do not scale the L_{ij}'s by $1/\epsilon$, but this is not so important.)

4.3   Creating Appropriate Boxes

Suppose K^{stab} is a stabilizing matrix. We generate a box $\mathcal{B}$ which contains K^{stab} and which is roughly a subset of the stabilizing matrices as follows. Let $\{E_{ij} \in \mathbb{R}^{m_2 \times p_2} \mid i = 1, \ldots, m_2, \ j = 1, \ldots, p_2\}$ be the standard orthonormal basis for $\mathbb{R}^{m_2 \times p_2}$. For each (i, j), define

$$\begin{array}{lll} \gamma_{ij}^{\min} = & \arg\min_{\gamma \in \mathbb{R}} & \gamma \\ & \text{subject to} & \alpha(A + B(K^{\text{stab}} + \gamma E_{ij})C) \leq 0, \end{array} \tag{4}$$

and

$$\begin{array}{lll} \gamma_{ij}^{\max} = & \arg\max_{\gamma \in \mathbb{R}} & \gamma \\ & \text{subject to} & \alpha(A + B(K^{\text{stab}} + \gamma E_{ij})C) \leq 0. \end{array} \tag{5}$$

Note that in (4) and (5) above, $\gamma_{ij}^{\min}$ and $\gamma_{ij}^{\max}$ are not required to be exact minimizers and maximizers; approximate values are sufficient.

The box is given by

$$\mathcal{B} = \{K \mid K_{ij}^{\min} \leq K_{ij} \leq K_{ij}^{\max}\}$$

where

$$K_{ij}^{\min} = K_{ij}^{\text{stab}} + \gamma_{ij}^{\min} E_{ij},$$
$$K_{ij}^{\max} = K_{ij}^{\text{stab}} + \gamma_{ij}^{\max} E_{ij}.$$

5 Numerical Experiments

This section contains some numerical experiments for various problems from the literature. Considered are both robust stabilization problems and H_∞ synthesis problems. In addition, in the last part of this section, results of using the preliminary part of the algorithm for finding stabilizing controllers are also presented.

All computational results were obtained using a 3 GHz Pentium 4 machine. Our algorithm was coded using Matlab 7.0.

5.1 Turbo-generator: Robust Stabilization

The first system considered is a turbo-generator model from [16] (system TG1 from the $COMPl_eib$ collection [21]). For this system, $n = 10$ and $m = p = 2$. The A matrix for this system is stable with $\beta(A) = 0.00767$. Our aim is to find K that maximizes $\beta(A + BKC)$.

The results of running the algorithm are given in Table 1 below. As can be seen from the table, the best stability radius achieved was $\beta(A + BKC) = 0.0784$, which is substantially better than $\beta(A)$. The best K found was

$$K = \begin{bmatrix} -0.99935 & -1.0807 \\ -0.099408 & -0.15920 \end{bmatrix}.$$

Table 1: Turbo-generator: Robust Stabilization, $k = 0$. q denotes the iteration number, β the stability radius, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	β	T	N
0	1.52e-2	1.3	-
1	6.89e-2	21	2800
2	7.19e-2	37	4900
3	7.82e-2	41	5500
4	7.84e-2	73	8900
5	7.84e-2	73	9400

Robust stabilization of the turbo-generator model is also considered in [7]. The solution given there is

$$K = \begin{bmatrix} -0.7763 & -0.7193 \\ -0.0935 & -0.1515 \end{bmatrix},$$

for which $\beta(A + BKC) = 0.0785$. This value is basically the same as our own.

5.2 Boeing 767: Robust Stabilization

The next system considered is a model of a Boeing 767 aircraft at a flutter condition [8] (system AC10 from the $COMPl_e ib$ collection [21]). For this system, $n = 55$ and $m = p = 2$. The A matrix is unstable. In this subsection we consider for this system the problems of robust stabilization via static control and robust stabilization via low-order dynamic control.

Numerical methods capable of finding stabilizing controllers for the system have only recently appeared; see [7], [1] and [2]. As we will see below, our algorithm is also able to stabilize this system. We note however that if, rather than using the full algorithm, a stabilizing solution is sought by just applying AGOP to f, then a stabilizing solution may not be found. In fact this deficiency motivated use to create the portion at the start of the algorithm that seeks a stabilizing solution.

In [7], robust stabilization of the system is considered for $k = 0$ (the static controller case), and $k = 1$ and $k = 2$ (low-order dynamic control). We now demonstrate that our algorithm is able to find better solutions than those appearing in [7], which up to now have been the best available.

For $k = 0$, the results are given in Table 2. The stability radius achieved by the algorithm was $\beta(A + BKC) = 9.33 \times 10^{-5}$ with

$$K = \begin{bmatrix} -1.1179 & -1.6880e-5 \\ 4.3557e+1 & 1.9930e-4 \end{bmatrix}.$$

By comparison, the solution obtained in [7] produces a stability radius of $\beta(A + BKC) = 7.91 \times 10^{-5}$, which is less than our own value.

Table 2: Boeing 767: Robust Stabilization, $k = 0$. q denotes the iteration number, β the stability radius, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	β	T	N
0	9.09e-6	1.1	-
1	7.04e-5	320	2000
2	9.20e-5	480	3100
3	9.23e-5	470	3000
4	9.33e-5	480	2900
5	9.33e-5	500	3000

For $k = 1$, the results are given in Table 3. The stability radius achieved by the algorithm was 1.24×10^{-4} with

$$\begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} = \begin{bmatrix} -2.0566e-1 & -1.2201e+2 & 5.6638e-3 \\ -1.5569e-2 & -5.2806e-1 & 2.5267e-6 \\ -1.4663e-2 & 5.8900 & 3.9470e-5 \end{bmatrix}.$$

By comparison, the solution obtained in [7] produces a stability radius of 9.98×10^{-5}, which again is less than our own value.

Table 3: Boeing 767: Robust Stabilization, $k = 1$. q denotes the iteration number, β the stability radius, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	β	T	N
0	3.05e-6	1.5	-
1	1.24e-4	640	4200
2	1.24e-4	660	4300

For $k = 2$, the results are given in Table 4. The stability radius achieved by the algorithm was 2.00×10^{-4} with

$$\begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} = \begin{bmatrix} -4.1367e{+}1 & -4.6368e{+}1 & 4.7884e{+}1 & -1.3239e{-}2 \\ -3.7889e{+}1 & -4.2488e{+}1 & -3.3479e{+}1 & -8.5480e{-}3 \\ -1.3890e{-}1 & -2.1165e{-}1 & 1.1020 & -3.6489e{-}5 \\ -5.6430e{-}1 & -5.2074e{-}1 & 5.7206e{-}2 & -1.2360e{-}4 \end{bmatrix}.$$

By comparison, the solution obtained in [7] produces a stability radius of 1.02×10^{-4}. Hence, our stability radius is again better, and in this case, rather significantly better.

Table 4: Boeing 767: Robust Stabilization, $k = 2$. q denotes the iteration number, β the stability radius, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	β	T	N
0	2.43e-5	2.4	-
1	1.86e-4	1300	8400
2	2.00e-4	2200	17000
3	2.00e-4	1800	16000

Finally, before ending this subsection, let us make an observation regarding the Boeing 767 system. Examining the system matrices reveals that, while the nonzero entries in B are of the same magnitude, the entries in the first row of C are roughly 10^5 times smaller in magnitude than the entries of the second row of C. That is, the problem is poorly scaled. This issue can be overcome by multiplying the second row of C by 10^{-5}. If a controller K could be found for this re-scaled system, a controller for the original unscaled system would be K with its last column multiplied by 10^{-5}. Using this re-scaling method, just using AGOP applied to f also finds a stabilizing solution. In the results given above, we have *not* used re-scaling, and hence the full algorithm is superior in this regard.

5.3   Boeing 767: H_∞ Synthesis

In this subsection we again consider the Boeing 767 system but this time consider the problem of H_∞ synthesis.

For $k = 0$, the results are given in Table 5. The minimum value achieved by the algorithm

was $\|T_{w,z}(s,K)\|_\infty = 13.2$ with

$$K = \begin{bmatrix} -8.9180e - 1 & 2.1578e - 5 \\ 4.2991 & 2.0537e - 5 \end{bmatrix}.$$

For comparison purposes, the best result from the literature, see [2], gives H_∞ norm equal to 13.1

Table 5: Boeing 767: H_∞ Synthesis, $k = 0$. q denotes the iteration number, $\|T_{w,z}\|_\infty$ the H_∞ norm of the closed loop transfer function from w to z, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	$\|T_{w,z}\|_\infty$	T	N
0	5.46e+2	1.6	-
1	1.32e+1	630	4700
2	1.32e+1	590	4300

For $k = 1$, the results are given in Table 6. The minimum value achieved by the algorithm was $\|T_{w,z}(s,[A_K\ B_K;\ C_K\ D_K])\|_\infty = 10.4$ with

$$\begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} = \begin{bmatrix} -2.4176 & -2.6371e - 1 & 2.6845e - 4 \\ -2.9232 & 3.1731e - 1 & 1.4342e - 5 \\ -3.4393e - 1 & 2.4001e - 1 & 4.9838e - 5 \end{bmatrix}.$$

The best result from the literature, again see [2], gives H_∞ norm equal to 10.2.

Table 6: Boeing 767: H_∞ Synthesis, $k = 1$. q denotes the iteration number, $\|T_{w,z}\|_\infty$ the H_∞ norm of the closed loop transfer function from w to z, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	$\|T_{w,z}\|_\infty$	T	N
0	7.33e+2	2.1	-
1	1.31e+1	810	63000
2	1.20e+1	1300	9100
3	1.04e+1	1200	8100
4	1.04e+1	1300	9200

5.4 **Transport Airplane: H_∞ Synthesis**

The final system considered is a transport airplane [12] (system AC8 from the *COMPl$_e$ib* collection [21]). For this system, $n = 9$, $m_2 = 1$ and $p_2 = 5$. The A matrix is unstable.

The results are given in Table 7. The minimum value achieved by the algorithm was $\|T_{w,z}(s,K)\|_\infty = 2.01$ with

$$K = \begin{bmatrix} 1.3018 & -0.98672 & -1.4860 & 0.063107 & 1.4209 \end{bmatrix}.$$

In [2], the result for this problem has the same H_∞ norm.

Table 7: Transport Airplane: H_∞ Synthesis, $k = 0$. q denotes the iteration number, $\|T_{w,z}\|_\infty$ the H_∞ norm of the closed loop transfer function from w to z, T the time in seconds required for each iteration, and N the number of f evaluations required for each iteration.

q	$\|T_{w,z}\|_\infty$	T	N
0	13.2	1.3	-
1	2.13	15	3200
2	2.04	32	5200
3	2.02	31	5000
4	2.01	35	5200
5	2.01	33	5000

5.5 Stabilization

Once the algorithm finds a stabilizing solution, it does not necessarily stop immediately though nor does it try to minimize (3) as much as it can. In this final part of the numerical results section, we demonstrate that the preliminary part of the algorithm can often be successful in quickly finding stabilizing controllers. In this case, the algorithm is terminated as soon as a stabilizing solution is found.

We ran the algorithm on each of the static output feedback problems given in the $COMPl_eib$ collection which were not open loop stable. Details of the problems can be found in [21].

Results are given in Table 8 and are quite good. Stabilizing controllers were found for 43 of the 49 systems. Of the 43 that were successfully stabilized, a stabilizing solution for 35 of the systems was found in 1.0 seconds or less.

6 Conclusions

In this paper the H_∞ synthesis problem was posed as an unconstrained, nonsmooth, nonconvex minimization problem in the entries of the output feedback matrix K. A numerical method for solving this reformulation of the problem was presented and application of the algorithm to various benchmark problems produced quite positive results. In particular, the algorithm was able to significantly improve on the best results appearing in the literature for robust stabilization of the Boeing 767 model. The effectiveness of the preliminary part of the algorithm for successfully and quickly finding stabilizing controllers was also demonstrated.

7 Acknowledgements

References

[1] P. Apkarian and D. Noll, Controller design via nonsmooth multi-directional search, Submitted.

[2] P. Apkarian and D. Noll, Nonsmooth H_∞ synthesis, Submitted.

Table 8: Using the (preliminary part of the) algorithm to find stabilizing solutions. T denotes the time in seconds required to find a stabilizing solution. '-' indicates a stabilizing solution was not found.

Problem	T	$\alpha(A + BKC)$	Problem	T	$\alpha(A + BKC)$
AC1	1.9e-1	-7.24e-3	DIS2	0	-8.83e-2
AC2	1.6e-1	-7.24e-3	DIS4	7.8e-1	-1.91e-1
AC4	3.1e-2	-5.00e-2	DIS5	-	-
AC5	-	-	WEC1	3.1e-2	-2.43e-2
AC7	3.1e-2	-4.82e-3	BDT2	1.3e-1	-5.16e-3
AC8	2.3e-1	-5.89e-2	IH	6.3e-2	-1.01e-3
AC9	4.5e-1	-1.14e-3	PAS	2.1e+1	-2.95e-8
AC10	1.9e-1	-2.33e-2	TF1	3.4e+1	-1.28e-2
AC11	4.7e-2	-3.80e-4	TF2	1.0e+1	-1.00e-5
AC12	2.5e-1	-8.39e-3	TF3	2.3e+1	-1.92e-3
AC13	1.4e-1	-9.29e-5	NN1	1.1	-2.47e-2
AC14	1.1e-1	-9.29e-5	NN2	1.6e-2	-1.25e-1
AC18	3.3e-1	-3.10e-1	NN3	-	-
HE1	3.1e-2	-3.14e-2	NN5	3.1e-2	-4.39e-4
HE3	1.2	-3.40e-3	NN6	8.1e-1	-7.84e-3
HE4	1.0	-7.57e-5	NN7	8.3e-1	-7.84e-3
HE5	4.8e-1	-3.09e-5	NN9	2.5	-1.67e-3
HE6	4.7e-2	-5.62e-4	NN10	-	-
HE7	3.1e-2	-5.62e-4	NN12	-	-
JE2	1.3	-1.33e-2	NN13	2.7e-1	-1.47e-1
JE3	1.0	-1.38e-2	NN14	2.5e-1	-1.47e-1
REA1	3.1e-2	-4.22e-2	NN15	4.7e-2	-3.28e-3
REA2	1.6e-2	-3.62e-2	NN16	3.1e-2	-3.50e-1
REA3	3.1e-2	-2.07e-2	NN17	4.7e-2	-1.81e-3
REA4	-	-			

[3] P. Apkarian, D. Noll, J.-B. Thevenet and H. D. Tuan, A spectral quadratic-SDP method with applications to fixed-order H_2 and H_∞ synthesis, in *Proceedings Asian Control Conf.*, Melbourne, Australia, 2004.

[4] P. Apkarian, D. Noll and H. D. Tuan, Fixed-order H_∞ control design via a partially augmented Lagrangian method, *Int. J. Robust Nonlinear Control* 13 (2003) 1137–1148.

[5] D.P. Bertsekas, Nonlinear Programming, 2nd ed., Athena Scientific, Belmont, Mass., 1999.

[6] V. Blondel and J.H. Tsitsiklis, NP-hardness of some linear control design problems, *SIAM J. Control Optim.* 35 (1997) 2118–2127.

[7] J.V. Burke, A.S. Lewis and M.L. Overton, A nonsmooth, nonconvex optimization approach to robust stabilization by static output feedback and low-order controllers, in *Proceedings ROCOND*, Milan, 2003.

[8] E.J. Davison, *Benchmark Problems for Control System Design: Report of the IFAC Theory Committee*, IFAC, Laxenberg, 1990.

[9] L. El Ghaoui and F. Oustry and M. Ait Rami, A cone complementarity linearization algorithm for static output-feedback and related problem, *IEEE Trans. on Automatic Control* 42 (1997) 1171–1176.

[10] 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) 348–360.

[11] B. Fares, D. Noll and P. Apkarian, Robust control via sequential semidefinite programming, *SIAM J. Control Optim.* 40 (2002) 1791–1820.

[12] D. Gangsaas, K.R. Bruce, J.D. Blight and U.-L. Ly, Application of modern synthesis to aircraft control: three case studies, *IEEE Trans. on Automatic Control* 31 (1986) 995–1014.

[13] K.M. Grigoriadis and E.B. Beran, Alternating projection algorithms for linear matrix inequalities problems with rank constraints, in *Advances on Linear Matrix Inequality Methods in Control*, L. El Ghaoui and S.-I. Niculescu (eds.), SIAM, Philadelphia, 1999, pp. 251–267.

[14] K.M. Grigoriadis and R.E. Skelton, Low-order control design for LMI problems using alternating projection methods, *Automatica* 32 (1996) 1117–1125.

[15] D. Hinrichsen and A. J. Pritchard, Stability radii of linear systems, *Systems and Control Letters* 7 (1986) 1–10.

[16] Y.S. Hung and A.G.J. MacFarlane, *Multivariable feedback: A quasi-classical approach*, Lecture Notes in Control and Information Sciences, Springer-Heidelberg, Berlin, 1982.

[17] S. Ibaraki and M. Tomizuka, Rank minimization approach for solving BMI problems with random search, in *Proceedings American Control Conference*, 2001, pp. 25–27.

[18] S. Kanev, C. Scherer, M, Verhaegen and B.De Schutter, Robust output-feedback controller design via a local BMI optimization, *Automatica* 40 (2004) 1115–1127.

[19] T.G. Kolda, R.M. Lewis and V. Torczon, Optimization by direct search: new perspectives on some classical and modern methods, *SIAM Review* 45 (2003) 385–482.

[20] F. Leibfritz, An LMI-based algorithm for designing suboptimal static $\mathcal{H}_2/\mathcal{H}_\infty$ output feedback controllers, *SIAM J. Control Optim.* 39 (2001) 1711–1735.

[21] F. Leibfritz, COMPl$_e$ib: COstrained Matrix-optimization Problem library - a collection of test examples for nonlinear semidefinite programs, control system design and related problems, University of Trier, 2003.

[22] F. Leibfritz and E. M. E. Mostafa, An interior point constrained trust region method for a special class of nonlinear semidefinite programming problems, *SIAM J. Optim.* 12 (2002) 1048–1074.

[23] M.A. Mammadov, A new global optimization algorithm based on a dynamical systems approach, in *Proc. International Conference on Optimization: Techniques and Applications - ICOTA6*, December, Ballarat, Also in: *Research Report 04/04, University of Ballarat*, 2004. http://www.ballarat.edu.au/ard/itms/publications/researchPapers/Papers_2004/04-04.pdf

[24] M.A. Mammadov, A. M. Rubinov and J. Yearwood, Dynamical systems described by relational elasticities with applications to global optimisation, in *Continuous Optimization: Current Trends and Applications*, V. Jeyakumar and A. Rubinov (eds.), Springer, 2005.

[25] D. Noll, M. Torki and P. Apkarian, Partially Augmented Lagrangian Method for Matrix Inequality Constraints, *SIAM J. Optim.* 15 (2004) 161–184.

[26] R. Orsi, U. Helmke and J. B. Moore, A Newton-like method for solving rank constrained linear matrix inequalities, in *Proc. 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, 2004, pp. 3138–3144.

[27] A.M. Rubinov and X.Q. Yang, Lagrange-Type Functions in Constrained Non-Convex Optimization, Applied Optimization, Vol. 85, Kluwer Academic Publishers, 2003.

[28] J.-B. Thevenet, D. Noll and P. Apkarian Nonlinear spectral SDP method for BMI-constrained problems: applications to control design, in *Proceedings ICINCO*, Portugal, 2004.

MUSA MAMMADOV
School of Information Technology & Mathematical Sciences, University of Ballarat, Australia
E-mail address: m.mammadov@ballarat.edu.au

ROBERT ORSI
Research School of Information Sciences and Engineering, The Australian National University,
Canberra ACT 0200, Australia
E-mail address: robert.orsi@anu.edu.au