Polynomial Methods and Convex Optimization for the Control of Input Constrained Systems*

Didier Henrion[1]
Sophie Tarbouriech[1]
Vladimír Kučera[2,3]

Abstract

A polynomial approach is pursued for locally stabilizing discrete-time linear systems subject to input constraints. Using the Youla-Kučera parametrization and geometric properties of polyhedra and ellipsoids, the problem of simultaneously deriving a stabilizing controller and the corresponding stability region is cast into a standard LMI problem. The approach is illustrated by a numerical example underlining the use of powerful CACSD tools such as the Polynomial Toolbox, Lmitool or Simulink for Matlab.

Keywords

Input Constraints, Polynomial Approach, LMIs, CACSD

1 Introduction

The problem of control constraints appears in most practical control systems. Due to technological and safety reasons, the actuators cannot drive an unlimited energy to the controlled plant. The presence of input bounds can be source of parasitic equilibrium points and limit cycles, or can even lead the closed-loop system to an unstable behavior.

[1]LAAS-CNRS, 7 Avenue du Colonel Roche, 31 077 Toulouse, Cedex 4, France

[2]Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Pod vodárenskou věží 4, 182 08 Prague 8, Czech Republic

[3]Trnka Laboratory for Automatic Control, Faculty of Electrical Engineering, Czech Technical University, Technická 2, 166 27 Prague 6, Czech Republic

*This work was supported by the Barrande Project No. 97/005-97/026, by the Grant Agency of the Czech Republic under contract No. 102/99/1368 and by the Ministry of Education of the Czech Republic under contract No. VS97/034. This work has been coordinated within the Network of Excellence for Industrial Applications of Polynomial Methods (EUROPOLY) No. CP97:7010.

In past years, this fact has motivated study of both analysis and design techniques taking control bounds into account.

Control limitation may be handled implicitly, or a posteriori, through the so-called anti-windup strategies [12]. Alternatively, input constraints may be handled explicitly, or a priori, pursuing either saturation avoidance methods or saturation allowance methods. Saturation avoidance methods consist in preventing input saturation, so that the closed-loop system stays in a region of linear behavior. They may rely upon positive invariance [2] or convex programming [3]. Saturation allowance methods consist in letting the saturation occur. The closed-loop system is therefore non-linear. Significant results have lately emerged in the scope of global stabilization [20, 19] and semi-global stabilization [16]. They inherently require stability of the open-loop system. Relaxing this stability assumption, results addressing the local stabilization problem have also been obtained [21, 8, 7, 6].

In this paper, we focus on a saturation avoidance method mixing some of the above mentioned techniques. On the one hand, we use the Youla-Kučera parametrization of all stabilizing controllers in the context of the polynomial approach to control systems [14, 15]. On the other hand, we use geometric properties of polyhedra and ellipsoids, traditionally invoked when studying positive invariance, to come up with a convex programming formulation of the constrained stabilization problem, as in [3]. To the authors knowledge, our development is the first application of the polynomial approach to the control of linear systems subject to input constraints.

The main emphasis of this paper is more on practical issues rather than on theoretical achievements. The interested reader may refer to [9, 10] for comprehensive and rigorous mathematical descriptions of our techniques. In the present paper, we rather illustrate the application of our techniques through Computer-Aided Control Systems Design (CACSD) tools.

The outline of the paper is as follows. In Section 2, we state the control problems to be solved. In Section 3, we describe systematic design methods. Our techniques are exemplified in Section 4 through a numerical example. We underline and illustrate the use of CACSD tools. The paper winds up with some concluding remarks in Section 5.

Notations: I_n is the $n \times n$ identity matrix, A^T is the transpose of matrix A, $\|x\|$ is the Euclidean norm of vector x, vector inequality $x \geq y$ is meant componentwise, $A \succeq B$ means that matrix $A - B$ is positive semidefinite, $P(\mathtt{d})$ denotes a polynomial matrix in the indeterminate d.

2 Problem Statement

Consider a multi-input multi-output (MIMO) observable discrete-time linear system

$$\begin{array}{rcl} \xi_{k+1} & = & F\xi_k + Gu_k \\ z_k & = & H\xi_k \end{array} \qquad (1)$$

where $\xi_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, $z_k \in \mathbb{R}^p$ stand for the state, input and output vectors, respectively. Input vector u_k is subject to hard constraints

$$-u^- \leq u_k \leq u^+ \tag{2}$$

where u^-, u^+ are given vectors with positive entries. Moreover, initial system state ξ_0 is supposed to belong to a given subset of the state-space. An outer convex approximation of this subset is supposed to be an ellipsoid

$$\begin{array}{rcl} \mathcal{E}_\Pi & = & \{\xi : (\xi-\xi_c)^T \Pi (\xi - \xi_c) \leq 1\} \\ & = & \{\Pi^{-\frac{1}{2}}\eta + \xi_c : \|\eta\| \leq 1\} \end{array} \tag{3}$$

where Π is a positive semi-definite matrix and ξ_c is a vector. Performing z-transform, linear system (1) can equivalently be written in a transfer function setting as

$$z(\mathtt{d}) = S(\mathtt{d})u(\mathtt{d}) + T(\mathtt{d})\xi_0 \tag{4}$$

where $\mathtt{d}$ stands for the usual backward shift operator [14] and rational matrices

$$\begin{array}{rcl} S(\mathtt{d}) & = & H(I_n - F\mathtt{d})^{-1}G\mathtt{d} \\ T(\mathtt{d}) & = & H(I_n - F\mathtt{d})^{-1} \end{array}$$

can be expressed using polynomial matrix fraction descriptions (MFDs, see [11] or [24]) as

$$\begin{array}{rcl} S(\mathtt{d}) & = & B(\mathtt{d})A^{-1}(\mathtt{d}) \\ & = & \bar{A}^{-1}(\mathtt{d})\bar{B}(\mathtt{d}) \\ T(\mathtt{d}) & = & \bar{A}^{-1}(\mathtt{d})\bar{C}(\mathtt{d}). \end{array} \tag{5}$$

System (4) should have no unstable hidden modes, so the matrix couples $A(\mathtt{d})$,$B(\mathtt{d})$ and $\bar{A}(\mathtt{d})$,$\bar{B}(\mathtt{d})$ in the above MFDs are assumed to be right and left coprime, respectively, for $|\mathtt{d}| \leq 1$. System (4) is to be driven by a dynamic controller

$$u(\mathtt{d}) = -K(\mathtt{d})z(\mathtt{d}) \tag{6}$$

where rational matrix $K(\mathtt{d})$ can be expressed using polynomial MFDs as

$$\begin{array}{rcl} K(\mathtt{d}) & = & Y(\mathtt{d})X^{-1}(\mathtt{d}) \\ & = & \bar{X}^{-1}(\mathtt{d})\bar{Y}(\mathtt{d}). \end{array} \tag{7}$$

Equivalently, dynamic controller (6) can be expressed in state-space setting as

$$\begin{array}{rcl} \eta_{k+1} & = & \bar{F}\eta_k + \bar{G}z_k \\ u_k & = & \bar{H}\eta_k + \bar{J}z_k \end{array} \tag{8}$$

where

$$K(\mathtt{d}) = \bar{H}(I_r - \bar{F}\mathtt{d})^{-1}\bar{G}\mathtt{d} + \bar{J}.$$

With these notations, two different problems are studied in this paper.

Problem 1 *Find dynamic controller (8) such that linear system (1) subject to input constraints (2) is stabilized when initialized within ellipsoid (3).*

Problem 2 *Find dynamic controller (8) and the largest ellipsoid (3) such that linear system (1) subject to input constraints (2) is stabilized when initialized within this set.*

3 Design Algorithms

3.1 Stabilization

With the help of the Youla-Kučera parametrization of all stabilizing controllers [13, 25] and a lemma for inscribing an ellipsoid into a polyhedron [23], a design technique is first developed to tackle Problem 1. The interested reader is referred to [9, 10] for a comprehensive treatment. The technique can be sketched as follows.

Let $\hat{X}(\mathtt{d})$, $\hat{Y}(\mathtt{d})$, $\hat{\bar{X}}(\mathtt{d})$, $\hat{\bar{Y}}(\mathtt{d})$ denote a particular solution to the double polynomial Bézout identity

$$\begin{bmatrix} \bar{X}(\mathtt{d}) & \bar{Y}(\mathtt{d}) \\ -\bar{B}(\mathtt{d}) & \bar{A}(\mathtt{d}) \end{bmatrix} \begin{bmatrix} A(\mathtt{d}) & -Y(\mathtt{d}) \\ B(\mathtt{d}) & X(\mathtt{d}) \end{bmatrix} = \begin{bmatrix} I_m & 0 \\ 0 & I_n \end{bmatrix} \tag{9}$$

where $A(\mathtt{d})$, $B(\mathtt{d})$ and $\bar{A}(\mathtt{d})$, $\bar{B}(\mathtt{d})$ are given coprime polynomial matrices. Then all linear controllers

$$\begin{aligned} u(\mathtt{d}) &= -Y(\mathtt{d})X^{-1}(\mathtt{d})z(\mathtt{d}) \\ &= -\bar{X}^{-1}(\mathtt{d})\bar{Y}(\mathtt{d})z(\mathtt{d}) \end{aligned}$$

that stabilize the linear system

$$\begin{aligned} z(\mathtt{d}) &= B(\mathtt{d})A^{-1}(\mathtt{d})u(\mathtt{d}) \\ &= \bar{A}^{-1}(\mathtt{d})\bar{B}(\mathtt{d})u(\mathtt{d}) \end{aligned}$$

are parametrized by

$$\begin{aligned} X(\mathtt{d}) &= \hat{X}(\mathtt{d}) + B(\mathtt{d})Q(\mathtt{d}) \\ Y(\mathtt{d}) &= \hat{Y}(\mathtt{d}) - A(\mathtt{d})Q(\mathtt{d}) \\ \bar{X}(\mathtt{d}) &= \hat{\bar{X}}(\mathtt{d}) + \bar{Q}(\mathtt{d})\bar{B}(\mathtt{d}) \\ \bar{Y}(\mathtt{d}) &= \hat{\bar{Y}}(\mathtt{d}) - \bar{Q}(\mathtt{d})\bar{A}(\mathtt{d}) \end{aligned} \tag{10}$$

where $Q(\mathtt{d})$, $\bar{Q}(\mathtt{d})$ are arbitrary stable rational matrices, or converging matrix power series in $\mathtt{d}$.

Let $A^j(\mathtt{d})$ and $Y^j(\mathtt{d})$ stand for the j-th row in matrices $A(\mathtt{d})$ and $Y(\mathtt{d})$, respectively. Define $q(\mathtt{d}) = q_0 + q_1\mathtt{d} + q_2\mathtt{d}^2 + \cdots$ as the transpose of the column vector obtained by stacking together the columns of parameter matrix $Q(\mathtt{d})$. Let

$$\begin{aligned} V^j(\mathtt{d}) &= \bar{C}(\mathtt{d}) \otimes [A^j(\mathtt{d})]^T &&= V_0^j + V_1^j\mathtt{d} + V_2^j\mathtt{d}^2 + \cdots \\ W^j(\mathtt{d}) &= -\hat{Y}^j(\mathtt{d})\bar{C}(\mathtt{d}) &&= W_0^j + W_1^j\mathtt{d} + W_2^j\mathtt{d}^2 + \cdots \end{aligned}$$

where $\otimes$ denotes the Kronecker product. Define

$$\bar{M}_i^j(q) = W_i^j + \sum_{k=0,1,2\ldots} q_k V_{i-k}^j$$

as a row vector parameterized by row vectors q_k for $k = 0, 1, 2, \ldots$ and let

$$M(q) = \begin{bmatrix} \bar{M}_0(q) \\ -\bar{M}_0(q) \\ \bar{M}_1(q) \\ -\bar{M}_1(q) \\ \vdots \end{bmatrix} \qquad \bar{M}_i(q) = \begin{bmatrix} \bar{M}_i^1(q) \\ \vdots \\ \bar{M}_i^m(q) \end{bmatrix} \qquad \mu = \begin{bmatrix} u^+ \\ u^- \\ u^+ \\ u^- \\ \vdots \end{bmatrix}.$$

Then we can show the following central result.

Theorem 1 *Problem 1 is solved if and only if there exists q such that*

$$\|\Pi^{-\frac{1}{2}}(M^i(q))^T\| + M^i(q)\xi_c \leq \mu^i, \quad i = 1, 2, \ldots \tag{11}$$

where $M^i(q)$ and μ^i stand for the i-th row in matrix $M(q)$ and vector μ, respectively. A stabilizing controller is retrieved from parameter q. A stability region for the closed-loop system is the polyhedron

$$\mathcal{P}_M = \{\xi_0 : M(q)\xi_0 \leq \mu\}. \tag{12}$$

Remark 1 (Polynomial Youla-Kučera Parameter) In general, sequence $Q(\mathtt{d})$ can be expressed as an infinite stable power series. However, for practical purpose, we only consider polynomial, or finite-impulse-response $Q(\mathtt{d})$ in the sequel.

Remark 2 (Convex Programming Problem) When $Q(\mathtt{d})$ is a matrix polynomial, relations (11) become a finite-dimensional second-order cone programming problem [17] that must be solved for vector coefficients q_k. We denote by N the row dimension of matrix $M(q)$.

In view of the above remarks, we can now describe an algorithm for solving Problem 1.

Algorithm 1 (Stabilization)

Input: System (1) with input constraints (2). An ellipsoid (3) of initial conditions that must be stabilized. Degree δ of Youla-Kučera parameter $Q(\mathtt{d})$.
Output: A stabilizing dynamic controller (8), if one exists.

Step 1: Compute polynomial matrices $A(\mathtt{d})$, $B(\mathtt{d})$, $\bar{A}(\mathtt{d})$, $\bar{B}(\mathtt{d})$, $\bar{C}(\mathtt{d})$ in MFDs (5).
Step 2: Solve Bézout equation (9) for polynomial matrices $\hat{X}(\mathtt{d})$, $\hat{Y}(\mathtt{d})$, $\hat{\bar{X}}(\mathtt{d})$, $\hat{\bar{Y}}(\mathtt{d})$.
Step 3: Assume polynomial matrix $Q(\mathtt{d})$ has degree δ. Build matrix $M(q)$ and vector μ.
Step 4: Solve second-order cone programming problem (11) for a parameter q. If the problem is infeasible, conclude that there is no Youla-Kučera parameter of degree δ that solves the problem and exit.
Step 5: Report $Q(\mathtt{d})$ into relations (10) to build dynamic controller polynomial matrices $X(\mathtt{d})$, $Y(\mathtt{d})$. Derive the corresponding state-space realization (8).

Remark 3 (Increasing Controller Order) Clearly, if the above algorithm fails for some degree δ, the designer may try higher values of δ. Note that this will most probably entail increasing the controller order.

3.2 Largest Stability Regions

The control law designed in Theorem 1 is valid when the closed-loop system state is initialized in a given convex subset of the state-space. Now we assume that this region is not given beforehand. As stated in Problem 2, a natural objective is then to find the largest possible stability region for which a stabilizing control law can be designed. In [10] we show that this problem proves very intricate in the general case, so that we will be satisfied with an approximate, but tractable solution. For simplicity, we assume that input constraints are symmetric, i.e. $u^+ = u^-$. If this is not the case, then some input centering step must be performed, see [10] for details.

Algorithm 2 (Largest Inscribed Ellipsoid)

Input: System (1) with input constraints (2). Degree δ of Youla-Kučera parameter $Q(\mathtt{d})$.
Output: A stabilizing dynamic controller (8) and the corresponding polyhedron (12) of stabilizable initial conditions, if some exist.

Steps 1,2,3 and 5: See Algorithm 1.
Step 4: Solve either the semidefinite optimization problem

$$\begin{array}{ll} \min & \mathrm{trace}\Pi \\ \text{s.t.} & \left[\begin{array}{cc} \Pi & (M^i(q))^T \\ M^i(q) & (\mu^i)^2 \end{array}\right] \succeq 0, \quad i = 1,2,\ldots,N \end{array} \tag{13}$$

of the semidefinite optimization problem

$$\begin{array}{ll} \min & \lambda \\ \text{s.t.} & \lambda I_n \succeq \Pi \\ & \left[\begin{array}{cc} \Pi & (M^i(q))^T \\ M^i(q) & (\mu^i)^2 \end{array}\right] \succeq 0, \quad i = 1,2,\ldots,N \end{array} \tag{14}$$

for a semidefinite positive matrix Π and a parameter q. If the problem is infeasible, conclude that there is no Youla-Kučera parameter of degree δ that solves the problem and exit.

4 Illustration

We will now illustrate the ideas exposed in the paper on a numerical example found in the control literature. In the sequel, the computations involving polynomials and polynomial matrices are readily performed with the Polynomial Toolbox Version 2.0 for MATLAB [18]. The semidefinite programming problems are implemented with the user-friendly interface LMITOOL 2.0 for MATLAB [5] and solved with the interior-point methods of the SDPHA package 3.0 [4]. Vertices of polytope $\mathcal{P}_M$ are very efficiently computed with the software QHULL [1]. Finally, closed-loop simulations are performed on Simulink 2.0 for MATLAB.

We consider the discrete-time linear system studied in [22]. Its state-space model reads

$$\begin{aligned} \xi_{k+1} &= \left[\begin{array}{cc} 0.8 & 0.5 \\ -0.4 & 1.2 \end{array}\right] \xi_k + \left[\begin{array}{c} 0 \\ 1 \end{array}\right] u_k \\ z_k &= \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right] \xi_k. \end{aligned}$$

System input is constrained to

$$-7 \leq u_k \leq 7.$$

Corresponding MATLAB matrices are reproduced in Script 0.

```
>> F = [0.8 0.5; -0.4 1.2];
>> G = [0; 1];
>> H = eye(2);
>> umin = 7; umax = umin;
```

Script 0. Input data.

Suppose that we aim at solving Problem 2, i.e. determining the largest ellipsoid of stabilizable initial conditions. We closely follow the steps of Algorithm 2 and illustrate the use of CACSD tools.

Step 1: Compute MFD polynomial matrices $A(\mathtt{d})$, $B(\mathtt{d})$, $\bar{A}(\mathtt{d})$, $\bar{B}(\mathtt{d})$ such that

$$B(\mathtt{d})A^{-1}(\mathtt{d}) = \bar{A}^{-1}(\mathtt{d})\bar{B}(\mathtt{d}) = H(I_n - F\mathtt{d})^{-1}G\mathtt{d}.$$

For example, this may be performed by MATLAB Scripts 1a and 1b. Macros `inv`, `rat2lmf` and `lmf2rmf` are provided in the Polynomial Toolbox 2.0.

```
>> [invF, detF] = inv(eye(2) - F*d); % inverse of a polynomial matrix
>> Num = H*invF*G*d; Den = ones(2)*detF; % transfer function matrices
>> [Bb, Ab] = rat2lmf(Num, Den) % rational matrices to left MFD
Bb =
      0.43d
     -0.69d
Ab =
     -1 + d      0.43
     -0.34     -0.69 + d
```

Script 1a. Direct computation of the left MFD.

```
>> [B, A] = lmf2rmf(Bb, Ab)
B =
     0.43d^2
     0.86d - 0.69d^2
A =
     0.86 - 1.7d + d^2
```

Script 1b. Right MFD from left MFD.

Step 2: Solve Bézout equation (9) for polynomial matrices $\hat{X}(\mathtt{d})$, $\hat{Y}(\mathtt{d})$, $\hat{\bar{X}}(\mathtt{d})$, $\hat{\bar{Y}}(\mathtt{d})$. First, we solve the equation

$$\hat{\bar{X}}(\mathtt{d})A(\mathtt{d}) + \hat{\bar{Y}}(\mathtt{d})B(\mathtt{d}) = I_m.$$

Second, we solve the equation

$$\begin{bmatrix} \hat{\bar{Y}}(\mathtt{d}) \\ \bar{A}(\mathtt{d}) \end{bmatrix} \hat{X}(\mathtt{d}) + \begin{bmatrix} -\hat{\bar{X}}(\mathtt{d}) \\ \bar{B}(\mathtt{d}) \end{bmatrix} \hat{Y}(\mathtt{d}) = \begin{bmatrix} 0 \\ I_n \end{bmatrix}$$

MATLAB Script 2 solves these equations. It is based on the Polynomial Toolbox equation solvers `xaybc` and `axbyc`.

```
>> % Solve first equation
>> [Xhb, Yhb] = xaybc(A, B, 1)
Constant polynomial matrix: 1-by-1
Xhb =
     1.2
Constant polynomial matrix: 1-by-2
Yhb =
     1     2.3
>> % Solve second equation
>> [Xh, Yh] = axbyc([Yhb; Ab], [-Xhb; Bb], [zeros(1,2);eye(2)])
Xh =
    -0.8 - 0.44d    -0.5 - d
     0.4 + 0.7d     -1.2 + 1.6d
Yh =
     0.096 + d    -2.8 + 2.3d
```

Script 2. Solving Bézout equation.

Step 3: We choose $\delta = 0$, i.e. a constant matrix parameter $Q(\mathtt{d})$. Then we build matrix $M(q)$ and vector μ. We collect coefficients q_k of polynomial row-vector $q(\mathtt{d})$ into a row-vector denoted by q. Matrices $S = [S^1\ S^2\ \cdots\ S^N]$ and $T = [T^1\ T^2\ \cdots\ T^N]$ such that $M^i(q) = qS^i + T^i$ are obtained by straightforward MATLAB operations. The matrices are given in Script 3.

```
>> S
S =
   -0.8918    0.3716    1.7836   -0.7432   -1.0345    0.4310
   -0.2973   -0.5945    0.5945    1.1891   -0.3448   -0.6897

>> T
T =
   -0.8800   -2.0000    1.8560    1.1600         0         0

>> mu = [umax umax umax]
mu =
     7     7     7
```

Script 3. Matrices S, T and vector μ.

Step 4: We solve LMI optimization problem (13) for a semidefinite positive matrix Π and a parameter q. We use the user-friendly interface LMITOOL 2.0 for MATLAB [5] to specify the LMI, as shown in Figure 1. A MATLAB script called `ellipsoid` is then created that solves the LMI.

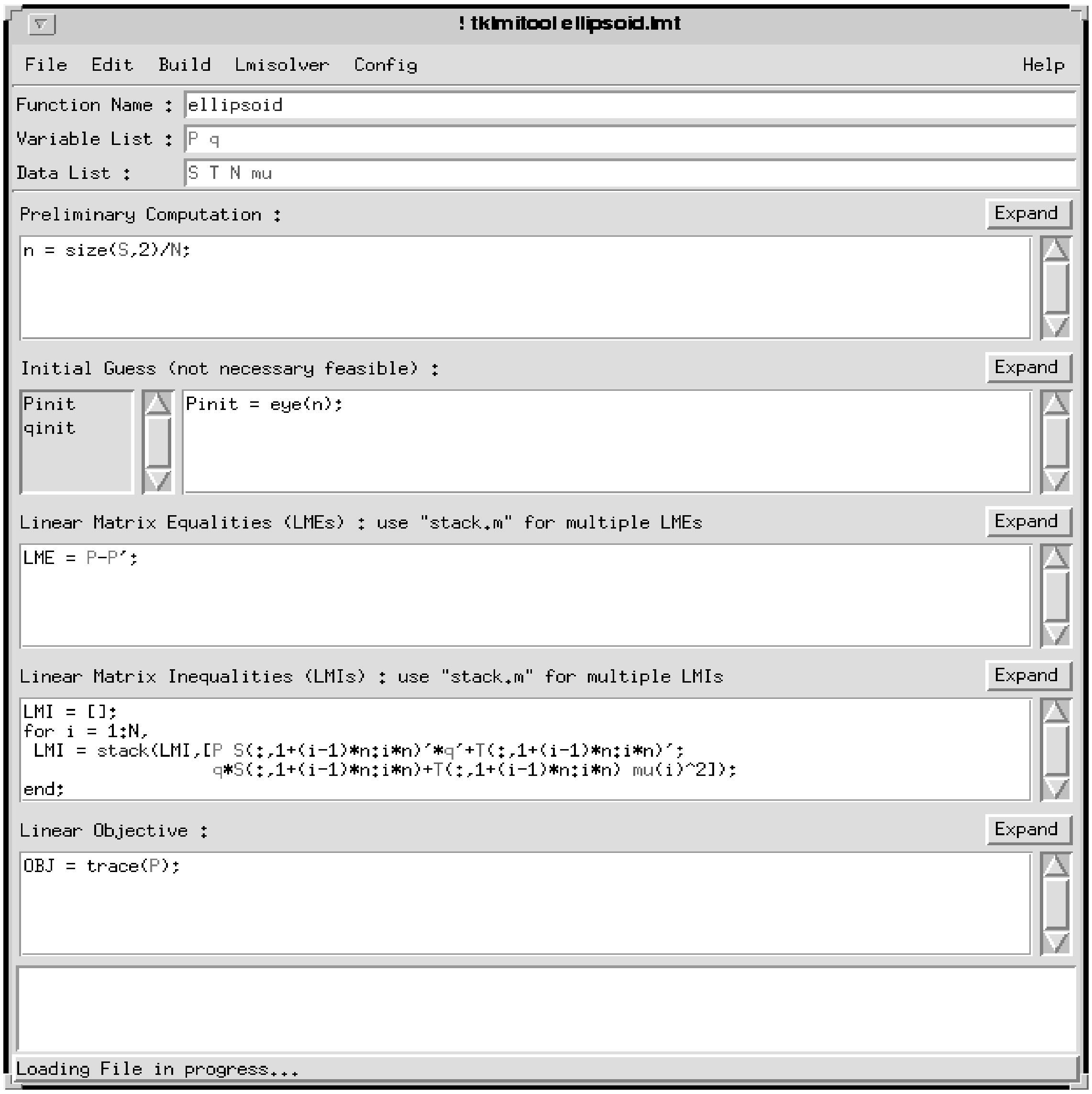

Figure 1. Building the LMI.

The LMI problem is solved with the interior-point methods of the SDPHA package 3.0 [4], as illustrated in MATLAB Script 4.

```
>> [P, q] = ellipsoid(S, T, 3, mu)
--------- Welcome to LMISOLVER ---------
------ You are using SDPHA method ------
 FORMATTING PHASE:
 Forming LMEs, LMIs and objective... Done.
 Eliminating equality constraints via SVD... Formatting done.
 condition no. of A = 1.99e+00
***************************************************************************
       SDPHA:  Homogeneous path-following algorithm
***************************************************************************
 version  predcorr  gam  expon  use_corrprim   rho
   2        1      0.00    1        1          Inf
it  step p_infeas d_infeas   gap         obj     pathres  sigma  rco   tau
------------------------------------------------------------------------------
 0 0.0e+00 1.2e+00 9.7e-01 9.0e+00  7.650000e+01 0.0e+00
 1 9.6e-01 7.7e-01 6.4e-01 9.5e+01  1.106802e+01 4.5e-01 0.132 4.7e-04  0.06670
 2 6.7e-01 2.3e-01 1.9e-01 3.4e+01 -8.111151e+00 8.5e-01 0.401 7.7e-04 0.07364
 3 8.2e-01 2.4e-02 2.0e-02 2.9e+00  2.848435e+00 6.8e-01 0.382 1.1e-03  0.12595
 4 9.6e-01 8.4e-04 6.9e-04 9.6e-02  1.959367e+00 4.6e-01 0.039 3.2e-03  0.12974
 5 7.9e-01 1.4e-04 1.2e-04 1.6e-02  1.962156e+00 7.8e-01 0.310 4.6e-04  0.16150
 6 7.9e-01 2.3e-05 1.9e-05 2.6e-03  1.956546e+00 6.6e-01 0.358 2.3e-04  0.20734
 7 9.7e-01 7.0e-07 5.8e-07 8.1e-05  1.956517e+00 2.2e-01 0.043 2.8e-05  0.21604
 8 9.9e-01 9.4e-09 7.7e-09 1.1e-06  1.956499e+00 1.5e-01 0.014 1.0e-06  0.21906
 9 9.9e-01 1.3e-10 1.0e-10 1.4e-08  1.956498e+00 3.7e-01 0.017 1.3e-08  0.22279

 Stop: relative gap < 1.00e-08
------------------------------------
 number of iterations   =  9
 gap                    = 1.41e-08
 Total CPU time         = 0.3
 CPU time per iteration = 0.0
 flops per iteration    = 3.42e+04
------------------------------------
 OBJECTIVE =
    0.0435
P =
    0.0143    0.0072
    0.0072    0.0292
q =
   -0.2029   -1.4794
```

Script 4. Solving the LMI optimization problem.

Step 5: Parameter matrix $Q(\mathtt{d})$ and controller matrices $X(\mathtt{d})$, $Y(\mathtt{d})$, $\bar{X}(\mathtt{d})$ and $\bar{Y}(\mathtt{d})$ are recovered with macros `pol`, `+`, `-` and `*` of the Polynomial Toolbox, see MATLAB Script 5.

```
>> Q = pol(q, 0, 'd')
Constant polynomial matrix: 1-by-2
Q =
    -0.2    -1.5

>> X = Xh + B*Q
X =
    -0.8 - 0.44d - 0.087d^2    -0.5 - d - 0.64d^2
     0.4 + 0.53d + 0.14d^2     -1.2 + 0.32d + d^2

>> Y = Yh - A*Q
Y =
     0.27 + 0.67d + 0.2d^2    -1.6 - 0.23d + 1.5d^2

>> Xb = Xhb + Q*Bb
Xb =
     1.2 + 0.93d

>> Yb = Yhb - Q*Ab
Yb =
     0.3 + 0.2d     1.4 + 1.5d
```

Script 5. Controller construction.

Now we can display the objects we have just created. We wrote simple MATLAB scripts to display ellipsoids and polyhedra. Computation of polyhedron vertices is performed with the software QHULL [1]. Polyhedron $\mathcal{P}_M$ and inscribed ellipsoid $\mathcal{E}_\Pi$ are represented in Figure 2.

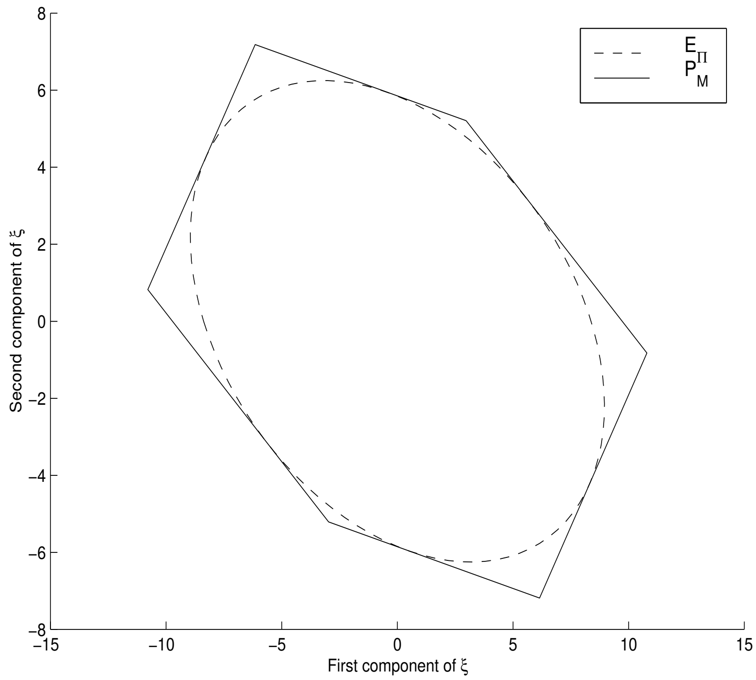

Figure 2. Polyhedron $\mathcal{P}_M$ and ellipsoid $\mathcal{E}_\Pi$.

To illustrate the fact that every initial condition ξ_0 picked up in polyhedron $\mathcal{P}_M$ is steered to the origin while verifying input constraints, we build a SIMULINK model of the constrained discrete-time system. As shown in Figure 3, a SIMULINK block for polynomial matrix MFDs is provided in the Polynomial Toolbox.

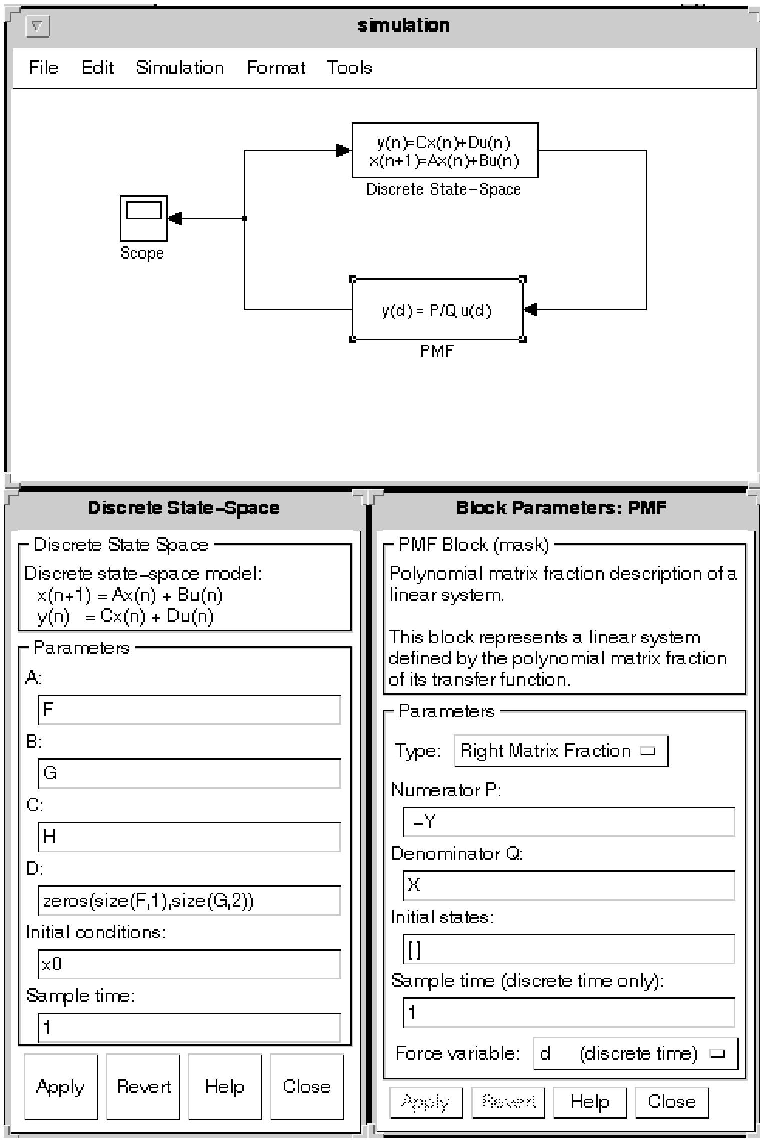

Figure 3. Simulink model.

Closed-loop simulations were carried out for different initial conditions ξ_0 chosen in polyhedron $\mathcal{P}_M$. Corresponding input profiles are represented in Figure 4. One can check that the system reaches the origin in three steps. Moreover, the input constraints are always satisfied.

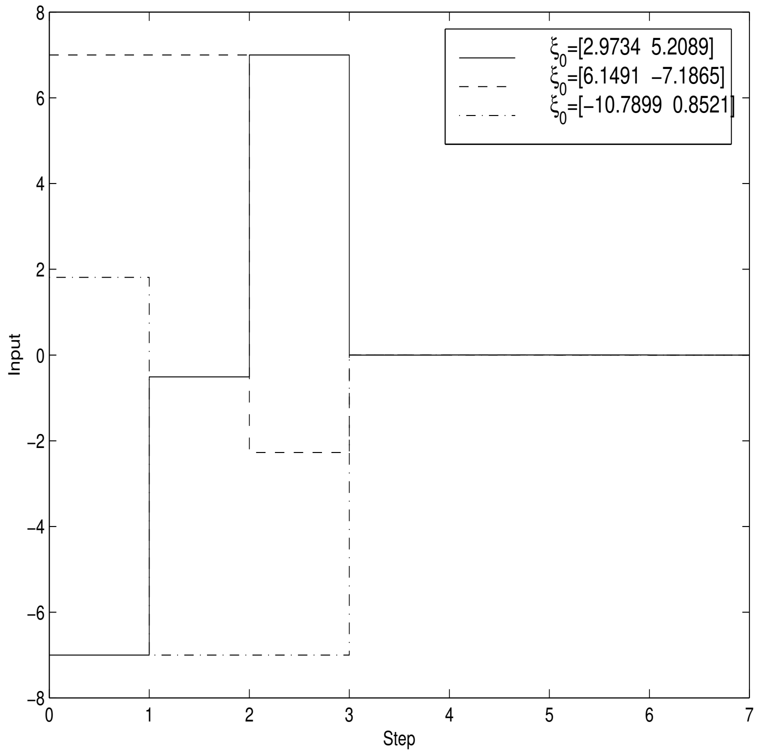

Figure 4. Closed-loop simulations.

5 Conclusion

We have proposed and illustrated practical and easily implementable algorithms for computing a stabilizing control law and the largest associated stability region for linear systems subject to input constraints. The strength of our technique is that it only resorts to convex programming, in contrast to other local stabilization methods relying upon non-convex bilinear problems [21, 8, 6].

Our work hints at several directions for future research. It can be extended to

- polyhedral or ellipsoidal constraints on system state, input or output
- uncertain systems, following the techniques outlined in [24]
- continuous-time systems
- performance requirements around the origin and trade-off between performance and the size of the stability domain, as in [7].

Related results will be published in forthcoming reports.

References

[1] C. B. Barber, D. P. Dobkin, H. T. Huhdanpaa "The Quickhull Algorithm for Convex Hulls", *ACM Transactions on Mathematical Software*, Vol. 22, No. 4, pp. 469–483, 1996. See also the home page for QHULL at `www.geom.umn.edu/software/qhull`.

[2] F. Blanchini "Set Invariance in Control - A Survey", submitted for publication, 1998.

[3] S. P. Boyd, V. Balakrishnan, C. H. Barratt, N. M. Khraishi, X. Li, D. G. Meyer, S. A. Norman "A New CAD Method and Associated Architectures for Linear Controllers", *IEEE Transactions on Automatic Control*, Vol. 33, No. 3, pp. 268–283, 1988.

[4] N. Brixius, R. Sheng, F. A. Potra "SDPHA: a MATLAB Implementation of Homogeneous Interior-Point Algorithms for Semidefinite Programming", to appear in *Optimization Methods and Software*, 1999. See also the home page for SDPHA at `www.math.uiowa.edu/~brixius/SDPHA`.

[5] L. El Ghaoui, J. L. Commeau "LMITOOL 2.0 Package: An Interface to Solve LMI Problems", *E-Letters on Systems, Control and Signal Processing*, Issue 125, January 1999. See also the LMITOOL home page at `www.ensta.fr/~gropco/lmi/lmitool.html`.

[6] J. M. Gomes da Silva Jr., S. Tarbouriech "Local Stabilization of Discrete-time Linear Systems with Saturating Controls: An LMI-based Approach", *Proceedings of the American Control Conference*, AACC, pp. 92–96, Philadelphia, Pennsylvania, 1998.

[7] D. Henrion, G. Garcia, S. Tarbouriech "Piecewise-linear Robust Control of Systems with Input Saturation", *Proceedings of the American Control Conference*, AACC, pp. 2703–2707, Philadelphia, Pennsylvania, 1998. Also to appear in *European Journal of Control*, Vol. 5, No. 1, 1999.

[8] D. Henrion, S. Tarbouriech, G. Garcia "Output Feedback Robust Stabilization of Uncertain Linear Systems with Saturating Controls", *Proceedings of the Conference on Decision and Control*, IEEE, pp. 193–198, San Diego, California, 1997. Also to appear in *IEEE Transactions on Automatic Control*, Vol. 45, No. 4, 2000.

[9] D. Henrion, S. Tarbouriech, V. Kučera "Control of Linear Systems Subject to Input Constraints: a Polynomial Approach. Part I – SISO Plants", *LAAS Technical Report*, No. 99060, Toulouse, France, 1999.

[10] D. Henrion, S. Tarbouriech, V. Kučera "Control of Linear Systems Subject to Input Constraints: a Polynomial Approach. Part II – MIMO Plants", to be registered as a *LAAS Technical Report*, Toulouse, France, 1999.

[11] T. Kailath "Linear Systems", *Prentice Hall*, Englewood Cliffs, New Jersey, 1980.

[12] M. V. Kothare, P. J. Campo, M. Morari, C. N. Nett "A Unified Framework for the Study of Anti-windup Designs", *Automatica*, Vol. 30, pp. 1869–1883, 1994.

[13] V. Kučera "Stability of Discrete Linear Feedback Systems", *Proceedings of the IFAC World Congress*, Boston, Massachussetts, Paper No. 44-1, 1975.

[14] V. Kučera "Discrete Linear Control: The Polynomial Equation Approach", *John Wiley and Sons*, Chichester, England, 1979.

[15] V. Kučera "Analysis and Design of Discrete Linear Control Systems", *Prentice Hall*, London, 1991.

[16] Z. Lin, A. Saberi "Semi-global Exponential Stabilization of Linear Systems subject to Input Saturation via Linear Feedback", *Systems and Control Letters*, Vol. 21, pp. 225–239, 1993.

[17] M. S. Lobo, L. Vandenberghe, S. Boyd, H. Lebret "Applications of Second-order Cone Programming", *Linear Algebra and its Applications*, Vol. 284, pp. 193–228, 1998.

[18] M. Šebek, H. Kwakernaak, D. Henrion, S. Pejchová "Recent Progress in Polynomial Methods and Polynomial Toolbox for MATLAB Version 2.0", *Proceedings of the Conference on Decision and Control*, IEEE, pp. 3661-3668, Tampa, Florida, December 1998. See also the Polynomial Toolbox home page at `www.polyx.cz`.

[19] R. Suárez, J. Alvarez-Ramírez, J. Solís-Daun "Linear Systems with Bounded Inputs: Global Stabilization with Eigenvalue Placement", *International Journal of Robust and Nonlinear Control*, Vol. 7, No. 9, pp. 835–846, 1997.

[20] H. J. Sussmann, E. D. Sontag, Y. Yang "A General Result on the Stabilization of Linear Systems using Bounded Controls", *IEEE Transactions on Automatic Control*, Vol. 39, No. 12, pp. 2411–2425, 1994.

[21] S. Tarbouriech, G. Garcia (Editors) "Control of Uncertain Systems with Bounded Inputs", *Lecture Notes in Control and Information Sciences*, Vol. 227, Springer Verlag, London, 1997.

[22] M. Vassilaki, J. C. Hennet, G. Bitsoris "Feedback Control of Linear Discrete-time Systems under State and Control Constraints", *International Journal of Control*, Vol. 47, No. 6, pp. 1727–1735, 1988.

[23] L. Vandenberghe, S. Boyd "Semidefinite Programming", *SIAM Review*, Vol. 38, No. 1, pp. 49–95, 1996.

[24] M. Vidyasagar "Control Systems Synthesis: a Factorization Approach", *MIT Press*, Boston, Massachussetts, 1985.

[25] D. C. Youla, H. A. Jabr, J. J. Bongiorno "Modern Wiener-Hopf Design of Optimal Controllers. Part II: the Multivariable Case", *IEEE Transactions on Automatic Control*, Vol. 21, pp. 319–338, 1976.