New robust control functions for the Polynomial Toolbox 3.0

Didier Henrion[1,2,3]
Michael Šebek[4]

February 13, 2003

Abstract

This report describes a bunch of new functions implemented in version 3.0 of the Polynomial Toolbox for Matlab. Calling sequences and functionalities are illustrated by numerical examples. The functions use convex optimization over linear matrix inequalities (LMIs) to solve various robust control problems.

1 Introduction

A set of new functions are being included to the new release 3.0 of the Polynomial Toolbox [PolyX 00], based on recent theoretical achievements in polynomial techniques and convex optimization. The new functions use optimization over linear matrix inequalities (LMIs) to solve various robust control problems:

- robust analysis:
  - `ptopana` - robust stability analysis of a polytope of polynomial matrices
  - `elliana` - robust stability radius of an ellipsoid of continuous-time scalar polynomials
  - `ellista` - ellipsoidal approximation of the stability domain in the coefficient space of a polynomial

[1]Corresponding author. FAX: +33 5 61 33 69 69. E-mail: `henrion@laas.fr`
[2]Laboratoire d'Analyse et d'Architecture des Systèmes, Centre National de la Recherche Scientifique, 7 Avenue du Colonel Roche, 31 077 Toulouse, cedex 4, France.
[3]Also with the Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Pod vodárenskou věží 4, 182 08 Praha, Czech Republic.
[4]Center for Applied Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 2, 166 27 Praha 6, Czech Republic.

- robust design:
  - `ptopdes` - robust stabilization of a polytope of scalar polynomials
  - `ellides` - robust stabilization of an ellipsoid of scalar polynomials
  - `ptopdes2` - robust proportional-derivative stabilization of a polytope of second-order systems
  - `ellides2` - robust proportional-derivative stabilization of an ellipsoid of second-order systems
  - `sofss` - simultaneous stabilization by scalar static output feedback
  - `hinfdes` - fixed-order H_∞ controller design

LMI problems are solved with the semidefinite programming feature of the solver SeDuMi [Sturm 99]. LMI problems are transformed into semidefinite programs with a user-friendly interface to SeDuMi [Peaucelle et al. 01].

2 Installation

The new functions require Matlab 6.5 and the Polynomial Toolbox 3.0. Both freeware SeDuMi 1.05 and its LMI interface 1.03 must be properly installed and reachable from Matlab's path. To download and install SeDuMi, refer to the instructions given at

`http://fewcal.kub.nl/sturm/software/sedumi.html`

To download and install the LMI interface to SeDuMi, refer to the instructions given at

`http://www.laas.fr/~peaucell/SeDuMiInt.html`

3 Notations

We say that a polynomial is stable when all its roots belong to a given stability region $\mathcal{D}$. We consider standard stability regions described by the quadratic scalar inequality

$$\mathcal{D} = \{s \in \mathbb{C} : \begin{bmatrix} 1 \\ s \end{bmatrix}^\star S \begin{bmatrix} 1 \\ s \end{bmatrix} < 0\}$$

where the star denotes transpose conjugate and S is a 2×2 Hermitian matrix referred to as the stability matrix. For example the choice

$$S = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

corresponds to the left half-plane (continuous-time polynomials) and

$$S = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

corresponds to the unit disk (discrete-time polynomials).

When referring to a monic polynomial of degree n

$$p(s) = p_0 + p_1 s + \cdots + p_{n-1} s^{n-1} + s^n$$

we use sometimes its coefficient vector

$$p = \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_{n-1} \end{bmatrix} \in \mathbb{R}^n.$$

4 Function `ptopana`

Function `ptopana` checks robust stability of a polytope of polynomial matrices. Given a set of polynomial matrix vertices $A_i(s)$ for $i = 1, 2, \ldots$ the function attempts to prove stability of the uncertain polynomial matrix

$$A(s) = \sum_i \lambda_i A_i(s), \quad \sum_i \lambda_i = 1, \quad \lambda_i \geq 0.$$

The underlying theory can be found in [Henrion et al. 01a] and [Henrion et al. 01b].

The function syntax is as follows:

```
ptopana(A)
ptopana(A,S)
```

where the first input argument is a cell array of polynomial matrices, and the second input argument is the stability matrix. Function `ptopana` returns `1` is the polytope is robustly stable. If the function returns `0` then we cannot conclude about robust stability.

4.1 Third degree continuous-time polytope with three vertices

With the following instructions:

```
>> A{1} = 28.3820+34.7667*s+8.3273*s^2+s^3;
>> A{2} = 0.2985+1.6491*s+2.6567*s^2+s^3;
```

```
>> A{3} = 4.0421+9.3039*s+5.5741*s^2+s^3;
>> ptopana(A)
ans =
     1
```

function `ptopana` proves robust stability of the continous-time polytope of degree 3 with 3 vertices. The root locus of the edges of the polytopic family is represented in Figure 1.

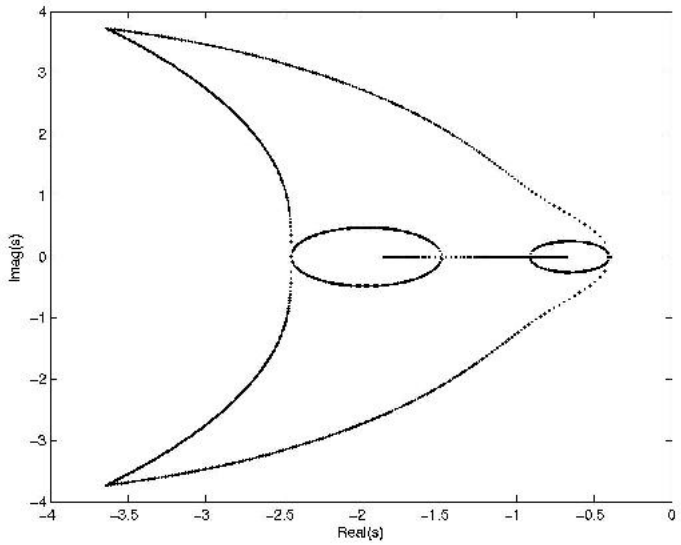

Figure 1: Root locus of the edges of the polynomial polytope.

4.2 Mechanical system

Consider the mechanical system represented in Figure 2, whose differential equations after application of the Laplace transform are given by

$$\begin{bmatrix} m_1 s^2 + d_1 s + c_1 + c_{12} & -c_{12} \\ -c_{12} & m_2 s^2 + d_2 s + c_2 + c_{12} \end{bmatrix} \begin{bmatrix} x_1(s) \\ x_2(s) \end{bmatrix} = \begin{bmatrix} 0 \\ u(s) \end{bmatrix}.$$

We assume that system parameters m_1, d_1, c_1, m_2, d_2, c_2 belong to the uncertainty hyper-rectangle $[1,3]\times[0.5,2]\times[1,2]\times[2,5]\times[0.5,2]\times[2,4]$ and we set $c_{12}=1$. This mechanical system is passive so it must be open-loop stable (when $u(s)=0$) independently of the values of the masses, springs, and dampers. However, it is a non-trivial task to know whether the open-loop system is robustly stable in some stability region $\mathcal{D}$ ensuring a certain damping. Here we choose the disk of radius 12 centered at -12:

$$\mathcal{D} = \{s \in \mathbb{C} : (s+12)^2 < 12^2\},$$

i.e. we set

$$S = \begin{bmatrix} 0 & 12 \\ 12 & 1 \end{bmatrix}$$

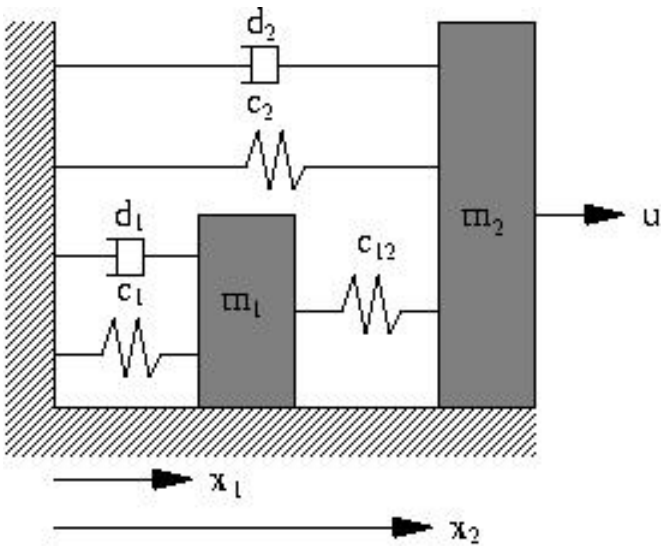

Figure 2: Mechanical system.

as the stability matrix. The robust stability analysis problem amounts then to assessing whether the second degree polynomial matrix is robustly stable in $\mathcal{D}$ for all admissible uncertainty. This is an interval polynomial matrix with $m = 2^6 = 64$ vertices. With this polynomial matrix polytope and the above stability matrix as input arguments, function `ptopana` returns 1, which proves robust stability, see the script below:

```
>> c12 = 1;
>> A = cell(1,2^6); i = 1;
>> for m1 = [1 3], for d1 = [0.5 2], for c1 = [1 2],
   for m2 = [2 5], for d2 = [0.5 2], for c2 = [2 4],
     A0 = [c1+c12 -c12; -c12 c2+c12]; A1 = [d1 0;0 d2]; A2 = [m1 0;0 m2];
     A{i} = pol([A0 A1 A2],2); i = i+1;
   end; end; end;
   end; end; end;
>> S = [0 12; 12 1];
>> ptopana(A,S)
ans =
     1
```

Therefore, the root-locus of the polynomial matrix remains in disk $\mathcal{D}$ for all admissible uncertainty. In Figure 3 we represented the roots of the 64 polynomial matrix vertices.

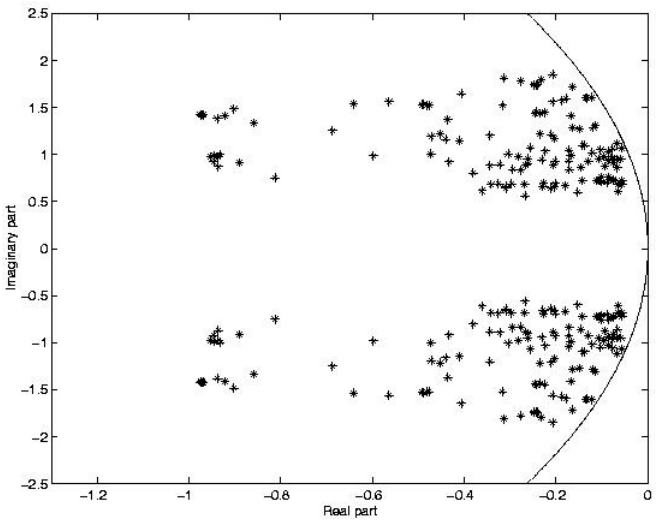

Figure 3: Roots of the 64 polynomial matrix vertices of the mechanical system.

5 Function `elliana`

Function `elliana` computes the largest radius r such that the continuous-time ellipsoid of polynomials

$$\{p(s,q) = p_0(s) + \sum_{i=1}^{n} q_i p_i(s),\ q^T q \leq r^2\},$$

remains robustly stable. In the above description, $p_0(s)$ is a given stable nominal polynomial, $p_i(s)$ are given polynomials of degree less than or equal to the degree of $p_0(s)$, and

$$q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}$$

is a real vector modelling the uncertainty, see [Ackermann 93, §7.2], [Barmish 94, Part IV] or [Bhattacharyya 95, §4.5].

The function syntax is as follows:

```
R = elliana(P0,[P1 P2 .. PN])
```

where the first input argument is the stable nominal polynomial $p_0(s)$ and the second input argument is an array of polynomials $p_i(s)$ describing the uncertainty ellipsoid.

5.1 First example

Consider as in [Barmish 94, Example 15.5.3] the ellipsoid of polynomials described by

$$\begin{aligned} p_0(s) &= 2 + 1.4s + 1.5s^2 + s^3 \\ p_i(s) &= s^{i-1}, \quad i = 1, .., 4 \end{aligned}$$

We obtain a stability radius of $r = 0.0330$ with the following script:

```
>> r = elliana(s^3+1.5*s^2+1.4*s+2,[1 s s^2 s^3])
r =
    0.0330
```

5.2 Second example

Consider as in [Ackermann 93, Example 7.2] the ellipsoid of polynomials described by

$$\begin{aligned} p_0(s) &= 129 + 166s + 237s^2 + 108s^3 + 80s^4 \\ p_1(s) &= -16 + 24s - 12s^2 + 4s^3 \\ p_2(s) &= -21 + 42s - 21s^2 \end{aligned}$$

We obtain a stability radius of $r = 1.1125$ with the following script:

```
>> r = elliana(129+166*s+237*s^2+108*s^3+80*s^4,...
       [-16+24*s-12*s^2+4*s^3, -21+42*s-21*s^2])
r =
    1.1125
```

6 Function `ellista`

Function `ellista` builds an inner ellipsoidal approximation $\mathcal{E}$ of the (generally non-convex) stability domain in the space of coefficients of a monic polynomial, based on the theoretical results of [Henrion et al. 01c]. Note that the LMI formulation implemented in function `ellista` slightly differs from the one found in the above reference: the current implementation is less conservative.

The function syntax is as follows:

```
[P,pc] = ellista(p0)
[P,pc] = ellista(p0,S)
P = ellista(pc,S,'center')
```

where the first input argument is either

- one arbitrary polynomial $p_0(s)$ in $\mathcal{E}$ (first and second calling syntaxes), or
- the center polynomial $p_c(s)$ of $\mathcal{E}$ (third syntax).

Stability region $\mathcal{D}$ can be specified through its defining matrix S as a second input argument. By default it is the standard stability region associated with the first input polynomial. A positive definite shaping matrix P as well as a center polynomial $p_c(s)$ are returned as output arugments such that

$$\mathcal{E} = \{p \in \mathbb{R} : (p - p_c)'P(p - p_c) \leq 1\}$$

is the ellipsoidal approximation of the stability domain in the space of polynomial coefficients.

6.1 Third-degree discrete-time stability domain

In the three-dimensional space of coefficients of third degree discrete-time monic polynomials, the exact stability domain is a non-convex set delimited by two triangles $V_1V_2V_3$, $V_2V_3V_4$ of vertices

$$p_{V_1} = \begin{bmatrix} 1 \\ 3 \\ 3 \end{bmatrix} \quad p_{V_2} = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \quad p_{V_3} = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \quad p_{V_4} = \begin{bmatrix} -1 \\ 3 \\ -3 \end{bmatrix}$$

supporting a hyperbolic paraboloid with saddle point

$$p_S = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.$$

With the following command, function `ellista` computes an inner approximation of the stability domain containing a stable polynomial with all roots at the origin:

```
>> [P,pc] = ellista(z^3)
P =
    2.3189    0.0000    0.4458
    0.0000    2.0180   -0.0000
    0.4458   -0.0000    1.7998
pc =
     2.5e-17 + 0.15z + 2.5e-16z^2 + z^3
```

The actual non-convex stability domain and its ellipsoidal approximation are represented in Figure 4.

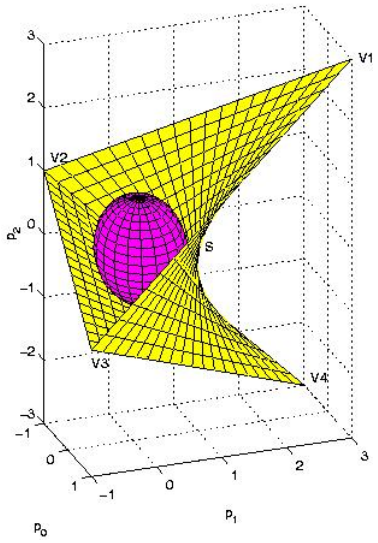

Figure 4: Hyperbolic and ellipsoidal stability domains for a third-degree discrete-time polynomial.

6.2 Second-degree continuous-time stability domains

In order to obtain ellipsoidal approximation of the stability domain of second-degree continuous-time polynomials with respective centers $(s+1)^2$, $(s+2)^2$ and $(s+3)^2$, we use the following sequence:

```
>> P1 = ellista((s+1)^2,[],'center');
>> P2 = ellista((s+2)^2,[],'center');
>> P3 = ellista((s+3)^2,[],'center');
```

We obtained the three ellipses represented in Figure 5. One can notice that the ellipse area increases with the norm of center vector coefficients, which is not surprising since the actual stability domain is the unbounded positive quadrant.

6.3 Third-degree continuous-time stability with stability margin

Let stability region $\mathcal{D}$ be the shifted left half-plane ensuring a stability margin of 2, i.e.

$$\mathcal{D} = \{s \in \mathbb{C} : \mathrm{Re}\, s < -2\}$$

which corresponds to the stability matrix

$$S = \begin{bmatrix} 4 & 1 \\ 1 & 0 \end{bmatrix}.$$

Enforcing for example $p_0(s) = (s+4)^3$ as a polynomial contained in ellipsoid $\mathcal{E}$, the following sequence computes an ellipsoidal approximation of the stability region:

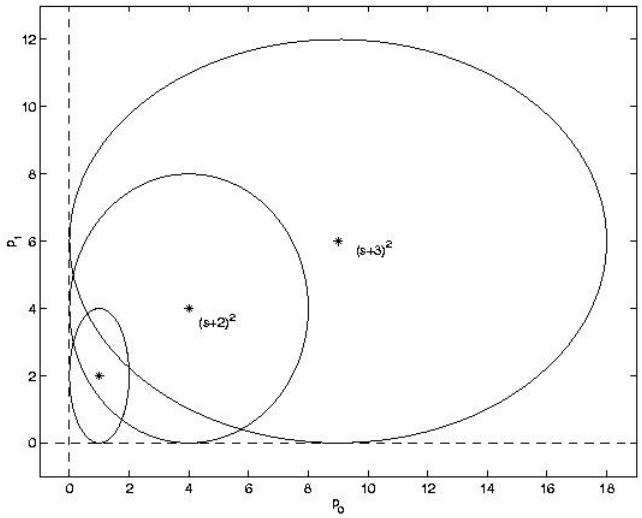

Figure 5: Continuous-time second-degree ellipsoidal stability domains for different choices of center polynomials.

```
>> [P,pc] = ellista((s+4)^3,[4 1;1 0])
P =
    0.0197   -0.0739    0.2760
   -0.0739    0.2956   -1.1824
    0.2760   -1.1824    5.0423
pc =
     2.2e+02 + 1.3e+02s + 22s^2 + s^3
```

Center polynomial $p_c(s)$ has its roots located in $-4.0076 \pm i0.0742$ and -13.7466, well inside the stability region $\mathcal{D}$.

7 Function `ptopdes`

This function attempts to stabilize a polytope of scalar plants with a fixed-order compensator. We consider a proper scalar plant

$$\frac{b(s,q)}{a(s,q)}$$

whose denominator and numerator polynomials are affected by polytopic uncertainty. The components of uncertainty parameter vector q belong to a polytope with given vertices q^i, i.e.

$$q = \sum_i \lambda_i q^i, \quad \sum_i \lambda_i = 1, \quad \lambda_i \geq 0.$$

We are seeking a controller

$$\frac{y(s)}{x(s)}$$

of fixed order with monic denominator polynomial. The controller is settled in a standard negative feedback configuration. Equivalently, polynomials $x(s)$ and $y(s)$ are sought such that the roots of polytopic characteristic polynomial

$$d(s,q) = a(s,q)x(s) + b(s,q)y(s)$$

remain in the stability region for all admissible values of uncertain parameter q in the polytope.

The calling syntax of function `ptopdes` is as follows:

```
[x,y] = ptopdes(a,b,c)
[x,y] = ptopdes(a,b,c,S)
[x,y] = ptopdes(a,b,c,S,pid)
```

The first and second input arguments are cell arrays of denominator and numerator polynomials $a(s,q^i)$ and $b(s,q^i)$ respectively. The third argument $c(s)$ is called the central polynomial. It plays a key role in the design process, as illustrated by the following numerical examples. See also [Henrion et al. 02c] for more information. Input polynomial $c(s)$ fixes the order of the sought controller, i.e. $\deg x(s) = \deg c(s) - \deg a(s)$. Stability region $\mathcal{D}$ can be specified through its defining matrix S as a fourth input argument. By default it is the standard stability region associated with the first input polynomial. Note that roots of central polynomial $c(s)$ must lie within the stability region $\mathcal{D}$. Finally, if a last input argument is present, then controller denominator is set to $x(s) = s$. So if the controller has order one (resp. two), it will be a PI (resp. PID).

7.1 Helicopter

The simplified linearized model of a laboratory experiment representing a helicopter is given by the plant

$$\frac{b(s,q)}{a(s,q)} = \frac{q_1}{s(0.1s+1)(s+q_2)}$$

affected by interval uncertainty

$$q_1 \in [20,\ 60],\ q_2 \in [0,\ 1].$$

Here uncertainty polytope Q is a two-dimensional box with $2^2 = 4$ vertices

$$q^1 = [20\ \ 0],\ q^2 = [20\ \ 1],\ q^3 = [60\ \ 0],\ q^4 = [60\ \ 1].$$

The nominal parameter vector is the center of the box $q^0 = [40\ \ 0.5]$. We are seeking a second-order monic controller

$$\frac{y(s)}{x(s)}$$

robustly stabilizing the interval plant.

In order to call function `ptopdes`, we need a so-called central closed-loop characteristic polynomial, central polynomial for short. A nominally stabilizing second-order controller, obtained by some design method, is given by

$$\frac{y^0(s)}{x^0(s)} = \frac{2 + 2.2s + 2.2s^2}{10s + s^2}.$$

The nominal characteristic polynomial reads

$$\begin{aligned} d(s, q^0) &= a(s, q^0)x^0(s) + b(s, q^0)y^0(s) \\ &= 80 + 88s + 93s^2 + 11s^3 + 2.05s^4 + 0.10s^5. \end{aligned}$$

Calling the Matlab function `ptopdes` with the 4 plant vertex numerators and denominators and the central polynomial $c(s) = d(s, q^0)$ as input arguments (see the script below), we obtain the following robustly stabilizing controller

$$\frac{y(s)}{x(s)} = \frac{0.9947 + 0.3007s + 1.574s^2}{1.611 + 6.232s + s^2}.$$

```
% parameter intervals
>> q1int = [20 60]; q2int = [0 1];
% vertices
>> a = cell(1,4); b = cell(1,4); i = 1;
>> for q1 = q1int, for q2 = q2int,
    a{i} = (0.1*s+1)*s*(s+q2); b{i} = q1; i = i+1;
   end; end;
% nominal plant
>> q1 = 40; q2 = 0.5;
>> a0 = (0.1*s+1)*s*(s+q2); b0 = q1;
% nominal controller & char poly
>> x0 = 10*s+s^2; y0 = 2+2.2*s+2.2*s^2; c = a0*x0+b0*y0;
% robust controller
>> [x,y] = ptopdes(a,b,c)
x =
     1.6 + 6.2s + s^2
y =
     0.99 + 0.3s + 1.6s^2
```

The function `ptopdes` attempts to minimize the Euclidean norm of the vector of controller coefficients. Here we obtain a controller of norm 6.78. As we can see on the robust root locus on Figure 6, stability is ensured but some poles are very near the imaginary axis. To ensure some robust stability margin, we can shift the stability region to the left. For example, we choose the shifted half plane

$$\mathcal{D} = \{s \in \mathbb{C} : \mathrm{Re}\, s < -0.4\},$$

i.e. we set

$$S = \begin{bmatrix} 0.8 & 1 \\ 1 & 0 \end{bmatrix}$$

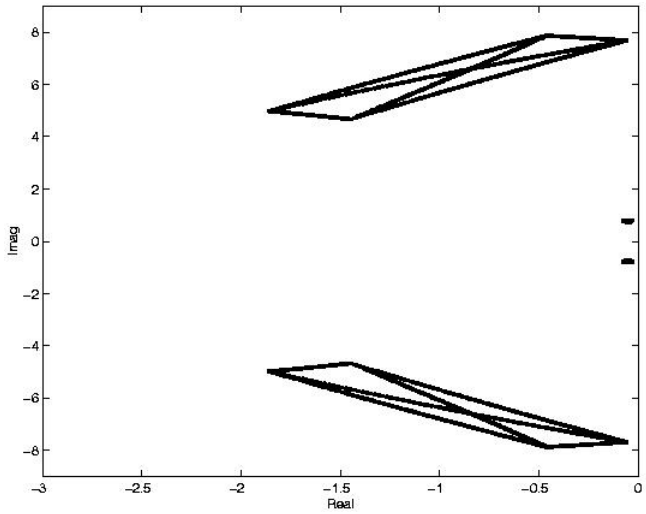

Figure 6: Helicopter. Edges of robust root locus without stability margin.

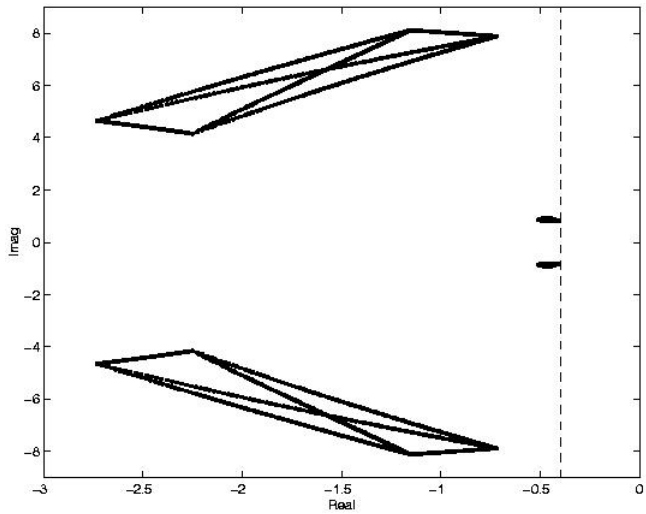

Figure 7: Helicopter. Edges of robust root locus with stability margin.

as the stability matrix.

In the above script, we just have to replace the instruction

```
[x,y] = ptopdes(a,b,c);
```

with

```
[x,y] = ptopdes(a,b,c,[0.8 1;1 0]);
```

Running the modified script, we obtain a controller

$$\frac{y(s)}{x(s)} = \frac{1.865 + 2.061s + 1.992s^2}{4.335 + 10.50s + s^2}$$

of norm 11.91 greater than the previous one. This could be expected since more control effort is needed to shift the poles farther from the imaginary axis. We can check on the robust root locus on Figure 7 that the required robust stability margin is ensured indeed.

7.2 F4E Aircraft

We consider Ackermann's model of the longitudinal motion of an F4E fighter aircraft [Ackermann 93, §1.4]. The input is the elevator position, the output is the pitch rate, and the system is linearized around four representative flight conditions in the Mach-altitude envelope:

Mach 0.5	5000 ft	$a^1(s) = -52.75 + 22.00s + 15.84s^2 + s^3$	$b^1(s) = -163.8 - 185.4s$
Mach 0.85	5000 ft	$a^2(s) = -122.5 + 34.93s + 17.12s^2 + s^3$	$b^2(s) = -789.1 - 507.8s$
Mach 0.9	35000 ft	$a^3(s) = -14.64 + 17.51s + 15.33s^2 + s^3$	$b^3(s) = -101.8 - 158.3s$
Mach 1.5	35000 ft	$a^4(s) = 269.1 + 43.60s + 15.74s^2 + s^3$	$b^4(s) = -251.4 - 304.2s.$

We are seeking a static output feedback controller simultaneously stabilizing the four plants with a stability margin of 0.5.

It is easy to see that the first plant can be stabilized with the controller polynomials $x^1(s) = 1$ and $y^1(s) = -1$. The resulting central polynomial

$$c(s) = a^1(s)x^1(s) + b^1(s)y^1(s) = 111.1 + 207.4s + 15.84s^2 + s^3$$

has roots -0.5588 and $-7.6410 \pm j11.85$ ensuring the stability margin. With the above four vertex plants $a^i(s)$, $b^i(s)$, the above central polynomial $c(s)$ and the stability matrix

$$S = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

i	roots of $d^i(s)$
1	$-0.5112,\ -7.664 \pm j10.80$
2	$-1.234,\ -7.943 \pm j19.84$
3	$-0.5000,\ -7.415 \pm j9.629$
4	$-1.717,\ -7.011 \pm j15.32$

Table 1: F4E aircraft. Closed-loop poles.

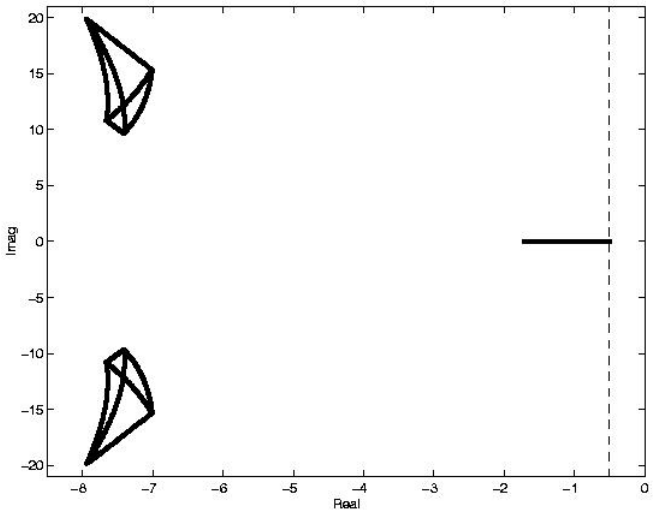

Figure 8: F4E aircraft. Edges of robust root locus.

as input arguments, function `ptopdes` returns the controller

$$\frac{y(s)}{x(s)} = -0.8692.$$

One can check that the above controller simultaneously stabilizes the four plants with the required stability margin. The roots of characteristic polynomials $d^i(s) = a^i(s)x(s) + b^i(s)y(s)$ are given in Table 1. Actually the above controller does not only stabilize simultaneously the four plants but also any plant within the convex hull of these four vertices, see the edges of the robust root locus on Figure 8.

7.3 Oblique wing aircraft

We consider the model of an experimental oblique wing aircraft studied in [Barmish 94, Example 11.5.1]

$$\frac{b(s,q)}{a(s,q)} = \frac{q_0 + q_1 s}{q_2 + q_3 s + q_4 s^2 + q_5 s^3 + s^4}$$

with

$q_0 \in [90, 166], q_1 \in [54, 74], q_2 \in [-0.1, 0.1], q_3 \in [30.1, 33.9], q_4 \in [50.4, 80.8], q_5 \in [2.8, 4.6]$

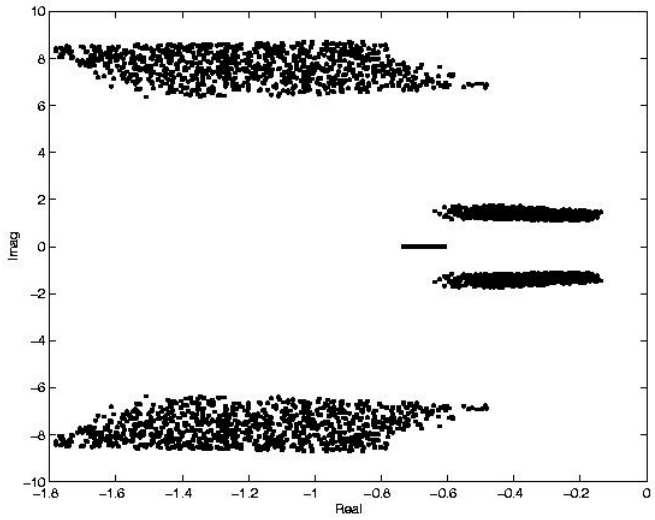

Figure 9: Oblique wing aircraft. Robust root locus.

that must be stabilized with a PI controller

$$\frac{y(s)}{x(s)} = K_p + \frac{K_i}{s}.$$

The interval plant corresponds to a polytope $a^i(s)$, $b^i(s)$ with $2^6 = 64$ vertices. One can check easily that the choice $K_p = 1$ and $K_i = 1$ stabilizes the vertex plant

$$\frac{b^1(s)}{a^1(s)} = \frac{90 + 54s}{-0.1 + 30s + 50s^2 + 2.8s^3 + s^4}$$

obtained by setting all the parameters to their minimum allowed values. With the choice

$$c(s) = sa^1(s) + (1 + s)b^1(s)$$

as the central polynomial, function `ptopdes` called with the script

```
>> i = 1; a = cell(2^6,1); b = cell(2^6,1);
>> for q0 = [90 166], for q1 = [54 74], for r0 = [-0.1 0.1],
   for r1 = [30.1 33.9], for r2 = [50.4 80.8], for r3 = [2.8 4.6],
      b{i} = q1*s+q0; a{i} = s^4+r3*s^3+r2*s^2+r1*s+r0; i = i+1;
   end; end; end;
   end; end; end;
>> c = s*a{1}+(1+s)*b{1};
>> [x,y] = ptopdes(a,b,c,[],'pi')
```

fails in finding a robustly stabilizing controller. In the above syntax the last input argument indicates that we are seeking a PI controller.

Considering now the second vertex plant

$$\frac{b^2(s)}{a^2(s)} = \frac{90 + 54s}{-0.1 + 30s + 50s^2 + 4.6s^3 + s^4}$$

where we changed only the coefficient of s^3 in the denominator polynomial, the choice

$$c(s) = sa^2(s) + (1+s)b^2(s)$$

as a third input argument to function `ptopdes` now proves successful. We obtain the robustly stabilizing PI controller

$$\frac{y(s)}{x(s)} = 0.8641 + \frac{0.6354}{s}.$$

The robust root locus, obtained by taking 1000 random plants within the uncertainty polytope, is represented on Figure 9.

7.4 Robot

We consider the problem of designing a robust controller for the approximate ARMAX model of a PUMA 762 robotic disk grinding process [Tong and Sinha 94]. From the results of identification and because of the nonlinearity of the robot, the coefficients of the numerator of the plant transfer function change for different positions of the robot arm. We consider variations of up to 20% around the nominal value of the parameters. The fourth-order discrete-time model is given by

$$\frac{b(z^{-1},q)}{a(z^{-1},q)} = \frac{(0.0257+q_1) + (-0.0764+q_2)z^{-1} + (-0.1619+q_3)z^{-2} + (-0.1688+q_4)z^{-3}}{1 - 1.914z^{-1} + 1.779z^{-2} - 1.0265z^{-3} + 0.2508z^{-4}}$$

where

$$|q_1| \leq 0.00514,\ |q_2| \leq 0.01528,\ |q_3| \leq 0.03238,\ |q_4| \leq 0.03376.$$

The characteristic polynomial of the closed-loop system is given by

$$d(z,q) = z^{12}[(1-z^{-1})a(z^{-1},q)x(z^{-1}) + z^{-5}b(z^{-1},q)y(z^{-1})]$$

where the term $1-z^{-1}$ is introduced in the controller denominator to maintain the steady state error to zero when parameters are changed. With the input central polynomial

$$c(z) = z^{19}$$

function `ptopdes` finds the seventh-order robust controller

$$\frac{y(z^{-1})}{x(z^{-1})} = \frac{-0.2863 + 0.2928z^{-1} + 0.0221z^{-2} - 0.1558z^{-3} + 0.0809z^{-4} + 0.1420z^{-5} - 0.1254z^{-6} + 0.0281z^{-7}}{1 + 1.1590z^{-1} + 0.9428z^{-2} + 0.4996z^{-3} + 0.3044z^{-4} + 0.4881z^{-5} + 0.4003z^{-6} + 0.3660z^{-7}}.$$

The robust root locus, obtained by taking 1000 random plants within the uncertainty polytope, is represented on Figure 10.

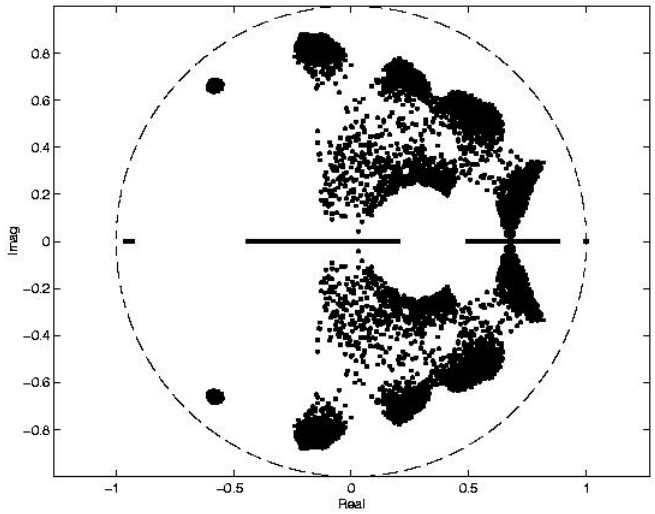

Figure 10: Robot. Robust root locus.

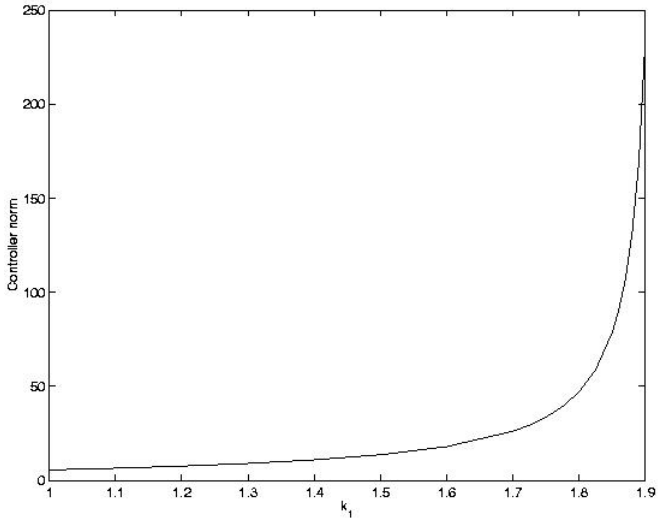

Figure 11: Norm of the first order controller as a function of upper gain k_1 with the central polynomial $c(s) = (s+1)^3$.

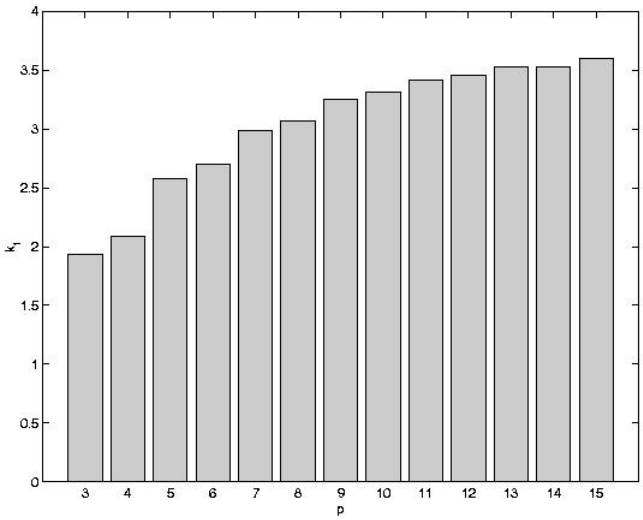

Figure 12: Maximum value of upper gain k_1 for which a first order robust controller is found with the central polynomial $c(s) = (s+1)^p$.

7.5 Stability margin optimization

We consider as in [Doyle et al. 92, §11.3] the problem of robustly stabilizing the plant

$$\frac{b(s,q)}{a(s,q)} = \frac{q(s-1)}{(s+1)(s-2)}$$

for all real gains q in the interval $[1,\ k_1]$. The uncertain plant polytope is therefore made of 2 vertices only. In [Doyle et al. 92] it is shown that a robustly stabilizing controller (of arbitrarily high order) exists if and only if $k_1 < 4$. The design method proposed there is based on coprime factorization and H_∞ model matching. It is solved with the help of Nevanlinna-Pick interpolation, which has the drawback of producing high-order controllers. In [Doyle et al. 92] a controller of eighth order is computed for $k_1 = 3.5$.

From the Hurwitz stability criterion there is no static controller stabilizing the plant, so we try the central polynomial $c(s) = (s+1)^p$ with $p \geq 3$ as input argument to `ptopdes` in order to seek a controller of order $p - 2 \geq 1$. When $p = 3$ we represent in Figure 11 the Euclidean norm of the first order controller obtained by function `ptopdes` as a function of k_1. Recall that the function is designed to minimize the Euclidean norm over all candidate controllers. In Figure 12 we reported as a function of degree p the maximum value of k_1 for which a robust controller is found with function `ptopdes`. For values of p greater than 15 we are reaching the limits of the SeDuMi solver, and the function fails for numerical reasons.

In the sequel we describe a heuristic to improve the stability margin with low-order controllers. Let $k_1 = 2$. We know from the results above that $c(s) = (s+1)^3$ is not a suitable choice of central polynomial for this value of k_1. It may mean that the poles of central polynomial $c(s)$ are not correctly chosen. So we try to move one pole nearest to the imaginary axis, just as in $c(s) = (s+1)^2(s+0.1)$ but function `ptopdes` fails as well. We try the opposite direction, i.e. $c(s) = (s+1)^2(s+10)$ and function `ptopdes` now

successfully returns a robustly stabilizing first-order controller

$$\frac{y(s)}{x(s)} = \frac{254.9 + 348.1s}{-327.9 + s}.$$

With this choice of central polynomial $c(s)$ function `ptopdes` finds a first-order controller up to $k_1 = 2.38$, so we have improved the stability margin without increasing the controller order. Pursuing this idea and trying to move poles of $c(s)$, we have been able to stabilize robustly the system for $k_1 = 2.59$ and $c(s) = (s+0.5)(s+1)(s+100)$ with the first-order controller

$$\frac{y(s)}{x(s)} = \frac{1292 + 1773s}{-1731 + s}.$$

This stability margin may perhaps be improved with further attempts.

Proceeding similarly with higher order controllers, we have been able to stabilize robustly the plant for $k_1 = 3.5$ with the third-order controller

$$\frac{y(s)}{x(s)} = \frac{240.8 + 755.5s + 871.7s^2 + 420.1s^3}{-423.5 - 739.8s - 409.5s^2 + s^3}$$

with the choice

$$c(s) = (s + 0.5)^3(s + 10)(s + 100)$$

as a central polynomial. Following this procedure, we may think about designing a heuristic to select poles of $c(s)$ and improve the stability margin without increasing the controller order.

8 Function `ellides`

This function attempts to stabilize a scalar plant affected by ellipsoidal uncertainty with a fixed-order compensator. We consider a proper scalar plant

$$\frac{b(s,q)}{a(s,q)} = \frac{b^0(s) + q'b^1(s)}{a^0(s) + q'a^1(s)}$$

whose denominator and numerator polynomials are affected by norm-bounded uncertainty: real parameter vector q satisfies $q'q \leq 1$ and $a^1(s)$, $b^1(s)$ are polynomial row vectors of the same dimension as q.

We are seeking a controller

$$\frac{y(s)}{x(s)}$$

of fixed order with monic denominator polynomial. The controller is settled in a standard negative feedback configuration. Equivalently, polynomials $x(s)$ and $y(s)$ are sought such that the roots of polytopic characteristic polynomial

$$d(s,q) = a(s,q)x(s) + b(s,q)y(s)$$

remain in the stability region for all q such that $q'q \leq 1$.

The calling syntax of function `ellides` is as follows:

```
[x,y] = ellides(a0,b0,a1,b1,c)
[x,y] = ellides(a0,b0,a1,b1,c,S)
[x,y] = ellides(a0,b0,a1,b1,c,S,pid)
```

The first and second input arguments are nominal polynomials. The third and fourth input arguments are uncertainty polynomials. The fifth argument is called the central polynomial. It plays the same role as in function `ptopdes`, see above. Stability region $\mathcal{D}$ can be specified through its defining matrix S as a sixth input argument. By default it is the standard stability region associated with the first input polynomial. Finally, if a last input argument is present, then controller denominator is set to $x(s) = s$. So if the controller has order one (resp. two), it will be a PI (resp. PID).

8.1 Mixing tanks

We consider the two mixing tanks arranged in cascade with recycle stream shown in Figure 13 and described in [Crisalle et al. 94]. The controller must be designed to maintain the

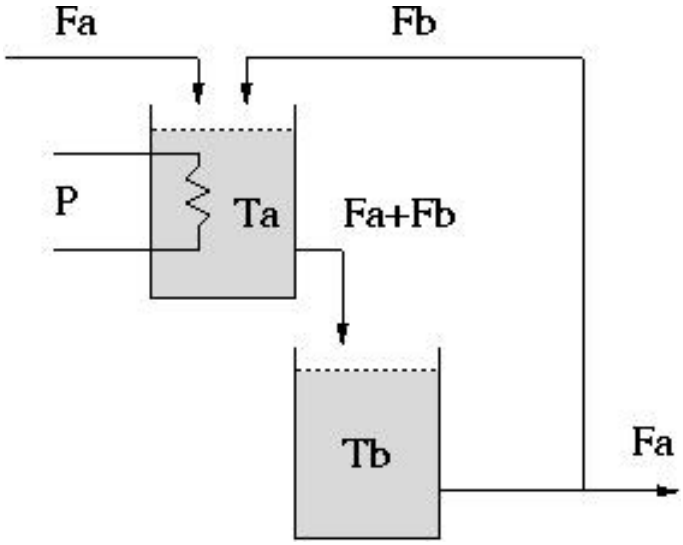

Figure 13: Two-tank system.

temperature T_b of the second tank at a desired set point by manipulating the power P delivered by the heater located in the first tank. The only available measurement is temperature T_b. The identification of the nominal plant model is carried out using a standard least-squares method [Crisalle et al. 94]. The discrete-time nominal plant is given by

$$\frac{b^0(z)}{a^0(z)} = \frac{b_0^0 + b_1^0 z}{a_0^0 + a_1^0 z + z^2}$$

with nominal plant vector

$$p_c = \begin{bmatrix} b_0^0 \\ b_1^0 \\ a_0^0 \\ a_1^0 \end{bmatrix} = \begin{bmatrix} 0.0038 \\ 0.0028 \\ 0.2087 \\ -1.1871 \end{bmatrix}.$$

An ellipsoidal uncertainty model is readily available as a by-product of the least-squares identification technique [Crisalle et al. 94]. The positive definite matrix P characterizing the uncertainty ellipsoid $\{p \,:\, (p-p_c)'P(p-p_c) \leq 1\}$ is given by

$$P = 10^5 \begin{bmatrix} 2.4179 & 0.0568 & 0.0069 & 0 \\ 0.0568 & 2.4121 & 0.0045 & 0.0062 \\ 0.0069 & 0.0045 & 0.0015 & 0.0014 \\ 0 & 0.0062 & 0.0014 & 0.0015 \end{bmatrix}.$$

Equivalently, uncertainty polynomial vectors are given by

$$a^1(s) = 10^{-2} \begin{bmatrix} -0.05575 + 0.03987z \\ 0.0002497 - 0.02517z \\ 19.73 - 13.77z \\ -13.77 + 19.62z \end{bmatrix}, \quad b^1(s) = 10^{-3} \begin{bmatrix} 2.036 - 0.02397z \\ -0.02397 + 2.037z \\ -0.5575 + 0.002497z \\ 0.3987 - 0.2517z \end{bmatrix}.$$

Now suppose that we are seeking a first-order controller

$$\frac{y_0 + y_1 z}{x_0 + z}$$

robustly stabilizing the plant for all admissible models within the uncertainty ellipsoid.

With the choice

$$c(z) = (0.1 + z)^3$$

as an input central polynomial, function `ellides` returns

$$\frac{y(z)}{x(z)} = \frac{6.068 + 6.981z}{0.3524 + z}$$

as a first-order robustly stabilizing controller, see the script below:

```
>> pc = [0.0038 0.0028 0.2087 -1.1871]';
>> P = 1e5*[2.4179 0.0568 0.0069 0;0.0568 2.4121 0.0045 0.0062
            0.0069 0.0045 0.0015 0.0014;0 0.0062 0.0014 0.0015];
>> Q = P^(-.5);
>> b0 = [1 z 0 0]*pc; a0 = [0 0 1 z]*pc+z^2;
>> b1 = ([1 z 0 0]*Q).'; a1 = ([0 0 1 z]*Q).';
>> c = (0.1+z)^3;
>> [x,y] = ellides(a0,b0,a1,b1,c)
x =
     0.35 + z
y =
     6.1 + 7z
```

The robust root-locus of closed-loop characteristic polynomial obtained by describing randomly the uncertainty ellipsoid is represented in Figure 14. We can check that indeed all characteristic polynomial roots stay in the unit disk for all admissible uncertainty.

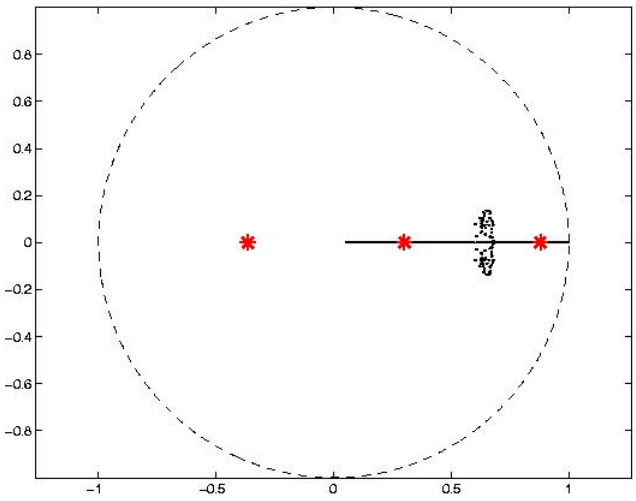

Figure 14: Root-locus within the unit disk.

Enforcing now pole location in the stability region

$$\mathcal{D} = \{z \in \mathbb{C} : |z| \leq 0.7\},$$

i.e. setting

$$S = \begin{bmatrix} -(0.7)^2 & 0 \\ 0 & 1 \end{bmatrix}$$

as the stability matrix in function `ellides`, we obtain

$$\frac{y(z)}{x(z)} = \frac{-35.69 + 137.3z}{0.5913 + z}$$

and the robust root-locus of figure 15.

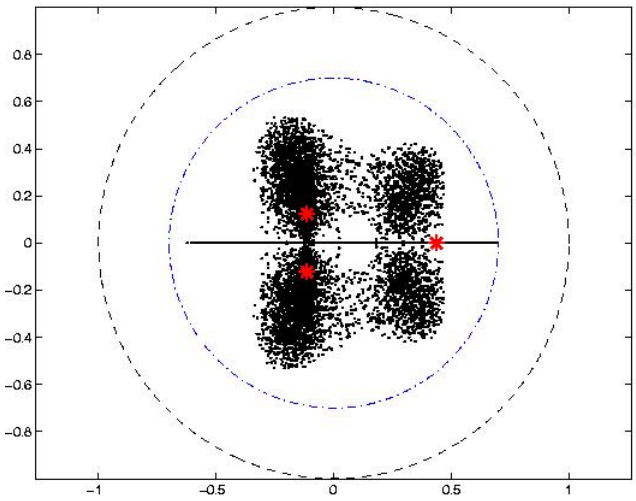

Figure 15: Root-locus within $|z| \leq 0.7$.

9 Function `ptopdes2`

Consider the multivariable linear system described by the second-order dynamical equations

$$\begin{aligned}(A_0 + A_1 s + A_2 s^2)x &= Bu \\ y &= Cx\end{aligned}$$

controlled by a proportional-derivative (PD) output-feedback controller of the form

$$u = -(F_0 + F_1 s)y$$

so that the closed-loop system behavior is captured by the quadratic polynomial matrix

$$N(s) = (A_0 + BF_0C) + (A_1 + BF_1C)s + A_2 s^2.$$

We assume that the second-order system is affected by polytopic uncertainty, i.e. quadratic matrix $A(s) = A_0 + A_1 s + A_2 s^2$ belongs to a polytope with given polynomial matrix vertices $A^1(s),\ A^2(s)\ , \ldots$

The calling syntax of function `ptopdes2` is as follows:

```
[F0,F1] = ptopdes2(A,B,C,D)
[F0,F1] = ptopdes2(A,B,C,D,S)
```

The first input argument is the quadratic polynomial matrix $A(s)$. The second input argument is the constant input matrix B (default identity). The third input argument is the constant output matrix C (default identity). The fourth input argument is the central (or nominal) closed-loop quadratic polynomial matrix, see the above description of function `ptopdes` for more information. Stability region $\mathcal{D}$ can be specified through its defining matrix S as a fifth input argument. By default it is the standard stability region associated with the first input polynomial. If S is a cell array of stability matrices S^j, then the stability region $\mathcal{D}$ is the intersection of all stability regions $\mathcal{D}^j$ corresponding to matrices S^j. Function `ptopdes2` attempts to find constant feedback matrices F_0 and F_1 such that closed-loop polynomial matrix $N(s)$ is robustly stable (i.e. its zeros are located with region $\mathcal{D}$) in the presence of polytopic uncertainty.

9.1 Mechanical structure

We consider the mechanical system shown on Figure 16, consisting of five material points linked by elastic springs [Barb et al. 01]. The points can slide without friction along their respective axes. Mass, distance to the origin at the equilibrium, and spring stiffness are given for each point in Table 2.

The system is controlled by two external forces acting at masses 1 and 5. System matrices

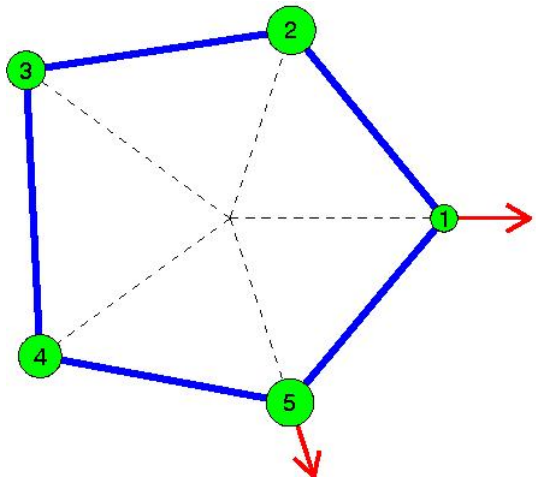

Figure 16: Five masses linked by elastic springs.

Point	Mass	Distance	Spring	Stiffness
1	0.5093	0.8034	1-2	1.461
2	0.9107	0.7430	2-3	1.369
3	0.7224	0.9456	3-4	1.088
4	0.8077	0.8810	4-5	1.203
5	0.8960	0.7282	5-1	1.468

Table 2: System data.

are given by

$$A_0 = \begin{bmatrix} 2.565 & 1.080 & 0 & 0 & 1.089 \\ 0.6038 & 0.8206 & 0.4766 & 0 & 0 \\ 0 & 0.6009 & 1.504 & 0.4808 & 0 \\ 0 & 0 & 0.4300 & 1.114 & 0.5131 \\ 0.6190 & 0 & 0 & 0.4626 & 0.8352 \end{bmatrix}, \quad A_1 = 0_5, \quad A_2 = I_5$$

and

$$B = \begin{bmatrix} 0 & 1.964 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1.116 & 0 \end{bmatrix}, \quad C = I_3.$$

Open-loop poles are all purely imaginary and located at $\pm i1.783$, $\pm i1.380$, $\pm i1.145$, $\pm i0.5675$ and $\pm i0.3507$.

A stabilizing PD controller is obtained in [Barb et al. 01] with a nearly optimal linear-quadratic robust design method:

$$\begin{aligned} F_0^0 &= \begin{bmatrix} 0.03960 & -0.02200 & 0.3685 & 0.8069 & 0.4099 \\ 0.3993 & 0.6453 & 0.4886 & 0.2269 & 0.03220 \end{bmatrix}, \\ F_1^0 &= \begin{bmatrix} 0.01520 & -0.3694 & 0.06470 & -0.04980 & 1.317 \\ 1.186 & -0.5896 & -0.2165 & -0.3263 & 0.02680 \end{bmatrix}. \end{aligned}$$

Poles of closed-loop quadratic matrix polynomial

$$D(s) = (A_0 + BF_0^0C) + (A_1 + BF_1^0C)s + A_2s^2$$

are located at $-0.1067 \pm i1.406$, -0.1405, $-0.1809 \pm i0.5350$, $-0.2174 \pm i1.099$, $-0.8157 \pm i1.450$ and -1.016. Feedback matrix $F^0 = [F_0^0 \quad F_1^0]$ has norm $f^0 = 1.859$.

In view of the closed-loop poles, we choose

$$\mathcal{D} = \{s \in \mathbb{C} : \mathrm{Re}\, s < -0.1\}$$

as the stability region. With the above polynomial matrix $D(s)$ as central system matrix, we invoke function `ptopdes2` as follows

```
>> A2 = eye(5); A1 = zeros(5);
>> A0 = [2.5647 1.0797 0 0 1.0890
         0.6038 0.8206 0.4766 0 0
         0 0.6009 1.5044 0.4808 0
         0 0 0.4300 1.1142 0.5131
         0.6190 0 0 0.4626 0.8352];
>> A = pol([A0 A1 A2],2);
>> B = [0 1.9637;0 0;0 0;0 0;1.1161 0];
>> F00 = -[-0.0396 0.0220 -0.3685 -0.8069 -0.4099
          -0.3993 -0.6453 -0.4886 -0.2269 -0.0322];
```

```
>> F01 = [0.0152 -0.3694 0.0647 -0.0498 1.3167
         1.1859 -0.5896 -0.2165 -0.3263 0.0268];
>> D = pol([A0+B*F00 A1+B*F01 A2],2);
>> S = [0.2 1;1 0];
>> [F0,F1] = ptopdes2(A,B,[],D,S)
```

Running the above script, we obtain the feedback matrices

$$F_0 = \begin{bmatrix} -0.1610 & -0.1136 & -0.03508 & 0.08337 & -0.05075 \\ -0.2706 & 0.04941 & 0.1440 & 0.06144 & -0.1366 \end{bmatrix},$$
$$F_1 = \begin{bmatrix} -0.1169 & -0.3153 & 0.2319 & 0.1873 & 0.5418 \\ 0.5383 & -0.2237 & -0.004471 & -0.2137 & 0.06340 \end{bmatrix}.$$

Poles of the new closed-loop quadratic matrix polynomial

$$N(s) = (A_0 + BF_0C) + (A_1 + BF_1C)s + A_2s^2$$

are located at $-0.1090 \pm i1.404$, $-0.1348 \pm i0.5436$, $-0.1445 \pm i1.1110$, $-0.1823 \pm i0.2301$ and $-0.2603 \pm i1.457$, well inside region $\mathcal{D}$. Feedback matrix $F = [F_0 \quad F_1]$ has largest singular value $f = 0.7593 < f^0$. Consequently, new feedback F requires less control effort and is less prone to saturation than original feedback F^0.

9.2 Vibrating rod

We consider as in [Datta et al. 97] the finite difference model of an axially vibrating non-conservative rod. The model is parametrized by the number of nodes n, and system matrices are given by $A_0 = 1000FF'$, $A_1 = FGF'$ and $A_2 = 2(I + SS') + S + S'$ where $S = [\delta_{i+1,j}]$ is a shift matrix of size n , δ_{ij} is the Kronecker delta, $F = I_n - S$, $G = 0.01\ \mathrm{diag}\{\sin\frac{i\pi}{2n}\}$ for $i = 1, \ldots, n$. We assume that all the inputs and the outputs are available for feedback. For example, when $n = 4$, system matrices are:

$$A_0 = 1000 \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}, \quad A_1 = 0.01 \begin{bmatrix} 1.090 & -0.7071 & 0 & 0 \\ -0.7071 & 1.631 & -0.9239 & 0 \\ 0 & -0.9239 & 1.924 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

and

$$A_2 = \begin{bmatrix} 4 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}.$$

Open-loop system poles are located at $-7.681 \cdot 10^{-5} \pm i5.102$, $-9.644 \cdot 10^{-4} \pm i16.10$, $-3.259 \cdot 10^{-3} \pm i29.24$ and $-8.195 \cdot 10^{-3} \pm i42.28$.

We choose the strip

$$\mathcal{D} = \{s \in \mathbb{C} : -2 < \mathrm{Re}\, s < -0.5\}$$

as the intersection of two basic stability regions and

$$D(s) = (s+1)^2 I_n$$

as an (arbitrary) central system matrix with zeros in $\mathcal{D}$. When $n = 4$ function `ptopdes2` returns a stabilizing PD compensator with feedback matrices

$$F_0 = 1000 \begin{bmatrix} -1.990 & 1.001 & 0.001493 & 0.0001069 \\ 1.001 & -1.989 & 0.9999 & -0.0001651 \\ 0.001493 & 0.9999 & -1.989 & 1.002 \\ 0.0001068 & -0.0001649 & 1.002 & -0.9955 \end{bmatrix},$$

$$F_1 = \begin{bmatrix} 10.24 & 1.704 & 0.6290 & 0.04552 \\ 1.704 & 10.86 & 1.456 & -0.0702 \\ 0.6289 & 1.457 & 10.61 & 2.486 \\ 0.04536 & -0.06999 & 2.486 & 4.931 \end{bmatrix}.$$

Closed-loop system poles are then located at $-1.218 \pm i0.8525$, $-1.222 \pm i0.8506$, $-1.239 \pm i0.8344$ and $-1.719 \pm i1.257$, well inside the assigned stability region.

9.3 Mass-spring system

Consider the undamped mass-spring example [Nichols and Kautsky 01, Example 1], where system matrices are given by:

$$A_0 = \begin{bmatrix} 40 & -40 & 0 \\ -40 & 80 & -40 \\ 0 & -40 & 80 \end{bmatrix}, \quad A_1 = 0_3, \quad A_2 = 10 I_3, \quad B = \begin{bmatrix} 1 & 2 \\ 3 & 2 \\ 3 & 4 \end{bmatrix}, \quad C = I_3.$$

Following [Nichols and Kautsky 01], a choice of nominally stabilizing PD controller matrices assigning closed-loop poles to -1, -2, -3, -4, -5 and -6 is as follows:

$$F_0^0 = \begin{bmatrix} 1.257 & 44.62 & -120.2 \\ -56.18 & -42.28 & 227.7 \end{bmatrix}, \quad F_1^0 = \begin{bmatrix} -86.18 & 27.23 & 16.52 \\ 85.49 & -13.02 & 4.992 \end{bmatrix}.$$

Now suppose that each diagonal entry in mass matrix A_2 belongs to an independent uncertainty interval [9, 11]. As a result, the system matrix belongs to a polytope with $2^3 = 8$ vertices. Let

$$\mathcal{D} = \{s \in \mathbb{C} : \operatorname{Re} s < -0.5\}$$

be the stability region, and let

$$D(s) = (A_0 + BF_0^0 C) + (A_1 + BF_1^0 C)s + A_2 s^2$$

be the central system matrix when diagonal entries in A_2 are equal to their nominal value 10. Function `ptopdes2` yields

$$F_0^R = \begin{bmatrix} -7.992 & -11.21 & 22.39 \\ -3.851 & 3.637 & 12.42 \end{bmatrix}, \quad F_1^R = \begin{bmatrix} -21.55 & 12.41 & 7.180 \\ 22.23 & -10.93 & 7.768 \end{bmatrix}$$

as robustly stabilizing feedback matrices. In Figure 17 we represent the closed-loop robust root locus for 10000 randomly chosen systems in the admissible uncertainty range. Closed-loop poles of the 8 polytope vertices are represented by red stars.

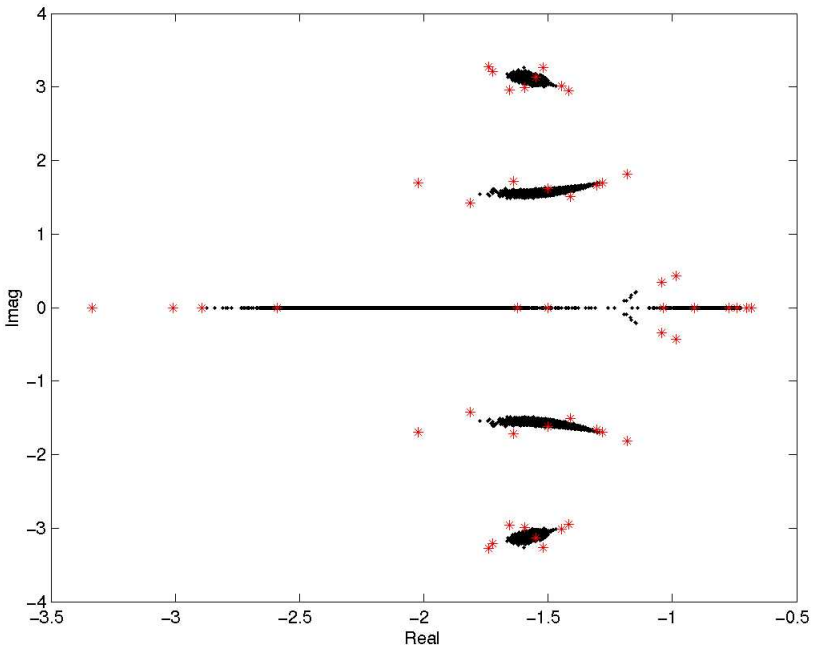

Figure 17: Robust root-locus of the mass-spring system.

10 Function `ellides2`

Now we assume that the second-order system introduced in the description of function `ptopdes2` is affected by ellipsoidal uncertainty. In other words, quadratic matrix $A(s)$ is subject to additive norm-bounded (unstructured) uncertainty

$$A(s) = A^0(s) + \Delta A^1(s), \quad \sigma_{\max}(\Delta'\Delta) \leq 1$$

where Δ is a uncertainty matrix of arbitrary column dimension, $A^0(s)$ is the nominal quadratic system matrix, $A^1(s)$ is the quadratic uncertainty matrix, and $\sigma_{\max}$ denotes the maximum singular value.

The calling syntax of function `ellides2` is as follows:

```
[F0,F1] = ellides2(A0,A1,B,C,D)
[F0,F1] = ellides2(A0,A1,B,C,D,S)
```

The first two input arguments are system matrix $A^0(s)$ and uncertainty matrix $A^1(s)$, respectively. The third input argument is the constant input matrix B (default identity). The fourth input argument is the constant output matrix C (default identity). The fifth input argument is the central (or nominal) closed-loop quadratic polynomial matrix, see [Henrion et al. 02d] and the above description of function `ptopdes2` for more information. Stability region $\mathcal{D}$ can be specified through its defining matrix S as a sixth input argument. By default it is the standard stability region associated with the first input polynomial.

If S is a cell array of stability matrices S^j, then the stability region $\mathcal{D}$ is the intersection of all stability regions $\mathcal{D}^j$ corresponding to matrices S^j. Function `ellides2` attempts to find constant feedback matrices F_0 and F_1 such that closed-loop polynomial matrix $N(s) = A^0(s) + \Delta A^1(s) + B(F_0 + F_1 s)C$ is robustly stable (i.e. its zeros are located within region $\mathcal{D}$) in the presence of ellipsoidal uncertainty Δ.

10.1 Wing in airstream

In [Tisseur and Higham 01] the authors consider an eigenvalue problem arising from the analysis of the oscillations of a wing in an airstream. Quadratic system matrix coefficients are given by:

$$A_0 = \begin{bmatrix} 121.0 & 18.90 & 15.90 \\ 0 & 2.700 & 0.1450 \\ 11.90 & 3.640 & 15.50 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 7.660 & 2.450 & 2.100 \\ 0.2300 & 1.040 & 0.2230 \\ 0.6000 & 0.7560 & 0.6580 \end{bmatrix}$$

and

$$A_2 = \begin{bmatrix} 17.60 & 1.280 & 2.890 \\ 1.280 & 0.8240 & 0.4130 \\ 2.890 & 0.4130 & 0.7250 \end{bmatrix}.$$

The system is open-loop unstable since its poles are located at $0.09427 \pm i2.553$, $-0.8848 \pm i8.442$ and $-0.9180 \pm i1.761$.

We choose $\mathcal{D} = \mathcal{D}^1 \cap \mathcal{D}^2$ with

$$\mathcal{D}^1 = \{s \in \mathbb{C} : -\mathrm{Re}\, s < 0\}, \quad \mathcal{D}^1 = \{s \in \mathbb{C} : -2 < -\mathrm{Re}\, s\}$$

as the stability region, i.e. we set

$$S^1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad S^2 = \begin{bmatrix} -4 & -1 \\ -1 & 0 \end{bmatrix}$$

as stability matices. Let $D(s) = (s+1)^2 I_3$ be the central closed-loop matrix with roots in $\mathcal{D}$. Macro `ellides2` called with no uncertainty matrix returns the following stabilizing feedback matrices:

```
>> A2 = [17.6 1.28 2.89; 1.28 0.824 0.413; 2.89 0.413 0.725];
>> A1 = [7.66 2.45 2.1; 0.23 1.04 0.223; 0.6 0.756 0.658];
>> A0 = [121 18.9 15.9; 0 2.7 0.145; 11.9 3.64 15.5];
>> A = pol([A0 A1 A2],2);
>> D = diag([(s+1)^2 (s+1)^2 (s+1)^2]);
>> S = cell(2,1);
>> S{1} = [0 1;1 0]; % Re(s) < 0
>> S{2} = [-4 -1;-1 0]; % Re(s) > -2
>> [F0,F1] = ellides2(A,[],[],[],D,S)
F0 =
   -4.8884  -13.1087   -2.4724
```

```
    1.9863   -1.0819    0.9174
    1.5609   -1.3376  -12.9224
F1 =
   14.0911   -3.9281    0.0044
    1.5266    0.6136    0.6725
    1.0762   -0.7899   -0.2328
>> N = pol([A0+F0 A1+F1 A2],2);
>> roots(N)
ans =
  -1.0498 + 2.6799i
  -1.0498 - 2.6799i
  -0.7443 + 1.5821i
  -0.7443 - 1.5821i
  -0.7736
  -0.4570
```

If a failure affects the second actuator, function `ellides2` is still able to compute a stabilizing PD feedback:

```
>> B = [1 0;0 0;0 1]; C = eye(3);
>> [F0,F1] = ellides2(A,[],B,C,D,S)
F0 =
   -4.3948  -15.1142   -0.4306
    1.9550   -1.5836  -12.8369
F1 =
   14.9815   -3.9831    0.7908
    2.1173   -0.4122    0.4698
>> N = pol([A0+B*F0*C A1+B*F1*C A2],2);
>> roots(N)
ans =
  -1.0335 + 2.6151i
  -1.0335 - 2.6151i
  -0.3391 + 1.7762i
  -0.3391 - 1.7762i
  -1.8296 + 0.5039i
  -1.8296 - 0.5039i
```

Similarly, assuming that all the actuators are available, but that a failure affects the second sensor, function `ellides2` returns

```
>> B = eye(3); C = [1 0 0;0 0 1];
>> [F0,F1] = ellides2(A,[],B,C,D,S)
F0 =
   -4.7800    3.0768
   13.5196    2.4315
    5.2709  -11.7931
```

```
F1 =
   15.7655    1.5617
    4.5934    0.7399
    3.2495    0.7613
>> N = pol([A0+B*F0*C A1+B*F1*C A2],2);
>> roots(N)
ans =
  -0.9126 + 2.6002i
  -0.9126 - 2.6002i
  -1.9667
  -1.3883
  -0.3567 + 0.4326i
  -0.3567 - 0.4326i
```

Finally, we suppose that the damping matrix is subject to additive norm-bounded uncertainty, i.e. $A^1(s) = \delta s I_3$. Function `ellides2` was then able to robustly stabilize the system:

```
>> A1 = 0.19*s*eye(3);
>> [F0,F1] = ellides2(A,A1,[],[],D,S)
F0 =
   -4.7768  -15.0286   -0.0750
    1.1389   -1.0147    0.3262
    2.9717   -2.8173  -13.0886
F1 =

   14.5147   -3.2621    0.2163
    0.3704    0.7278    0.3836
    2.1353   -0.3827    0.0369
```

for all uncertainty with worst-case norm $\delta = 0.19$. In Figure 18 we represent the closed-loop robust root locus for 10000 randomly chosen systems in the admissible uncertainty range. Nominal closed-loop poles are represented by red stars.

11 Function `sofss`

Let

$$\frac{b_i(s)}{a_i(s)}, \quad i = 1, 2 \dots$$

denote a set of scalar plants, where $a_i(s)$ and $b_i(s)$ are scalar polynomials of degree n. Function `sofss` solves the problem of finding a scalar static feedback gain k that simultaneous stabilizes the plants, i.e. such that the roots of all the characteristic polynomials

$$c_i(s) = a_i(s) + k b_i(s), \quad i = 1, 2 \dots$$

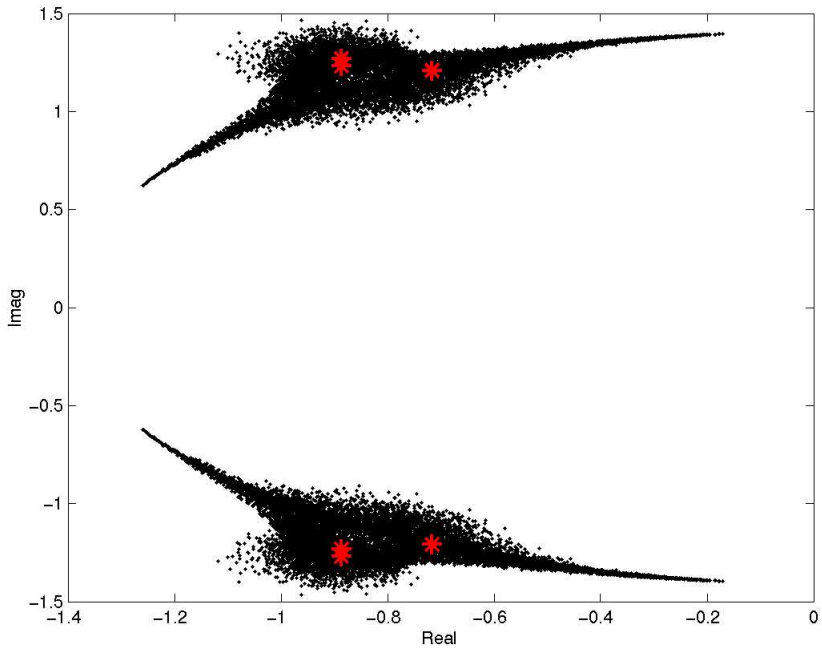

Figure 18: Robust root-locus of the wing.

belong to the left half plane.

Even though the problem can be solved using LMIs, function `sofss` solves the problem with standard numerical algebra. The calling syntax of function `sofss` is as follows:

```
inter = sofss(a,b)
```

where input arguments are cell arrays of denominator and numerator polynomials, respectively. The output argument is an two-dimensional arrays, where each row is an open interval $]k_j, k_{j+1}[$ for $j = 1, 2 \ldots$ Any feedback gain k chosen within these intervals will simultaneously stabilize the plants.

11.1 Reactor

Consider the continuous stirred tank reactor model studied in [Howitt and Luus 91]. The non-linear model is

$$\begin{aligned} x_1 &= (x_2 + 0.5)\exp(Ex_1/(x_1 + 2)) - (2 + u)(x_1 + 0.25) \\ x_2 &= 0.5 - x_2 - (x_2 + 0.5)\exp(Ex_1/(x_1 + 2)) \end{aligned}$$

where E is a parameter related to the activation energy. During the life of the reactor, some representative values of E are 20, 25 and 30. Assuming that only $y = x_1$ is available

for feedback, the $N = 3$ linearized systems of order $n = 2$ to be simultaneously stabilized are given by

$$\begin{aligned} b_1(s)/a_1(s) &= (0.5 - 0.25s)/(11 - 5s + s^2) \\ b_2(s)/a_2(s) &= (-0.5 - 0.25s)/(-2.25 - 2.25s + s^2) \\ b_3(s)/a_3(s) &= (-0.5 - 0.25s)/(-3.5 - 3.5s + s^2). \end{aligned}$$

Calling function `sofss` with the following script

```
>> b1=0.5-0.25*s;a1=11-5*s+s^2;
>> b2=-0.5-0.25*s;a2=-2.25-2.25*s+s^2;
>> b3=-0.5-0.25*s;a3=-3.5-3.5*s+s^2;
>> sofss({a1 a2 a3},{b1 b2 b3})
ans =
  -22.0000  -20.0000
```

we obtain that the three plants are simultaneously stabilizable by a static output feedback $u = ky$ for any value of k such that $-22 < k < -20$.

11.2 F4E aircraft

We consider as in Section 7.2 the problem of simultaneously stabilizing four operating points of the longitudinal short period mode of the F4E fighter aircraft. Applying function `sofss`, we obtain the interval $]-\infty, -0.3219[$, i.e. the four plants are simultaneously stabilizable by a static output feedback for any finite value of k such that $k < -0.3219$. In Figure 19 we represent the root locus of the four plants for $-1 < k < 2$ (in black) and $-1 < k < -0.3219$ (in red).

12 Function `hinfdes`

The scalar H_∞ design problem to be solved in this paper can be formally stated as follows. Given a set of polynomials $n_i^k(s)$, $d_i^k(s)$ for $i = 1, 2, \ldots$, $k = 1, 2, \ldots$, as well as a set of positive real numbers γ^k, seek polynomials $x_i(s)$ of given degrees such that

$$\left\| \frac{\sum_i n_i^k(s) x_i(s)}{\sum_i d_i^k(s) x_i(s)} \right\|_\infty < \gamma^k, \quad k = 1, 2, \ldots$$

In the above inequalities

$$\|S\|_\infty = \sup_{s \in \partial\mathcal{D}} |S(s)|$$

denotes the peak value of the magnitude of rational transfer function S when evaluated along the one-dimensional boundary $\partial\mathcal{D}$ of a given stability region $\mathcal{D}$.

The above H_∞ design paradigm covers all the standard frequency domain specifications arising in scalar control problems. For example, in the feedback system of figure 20

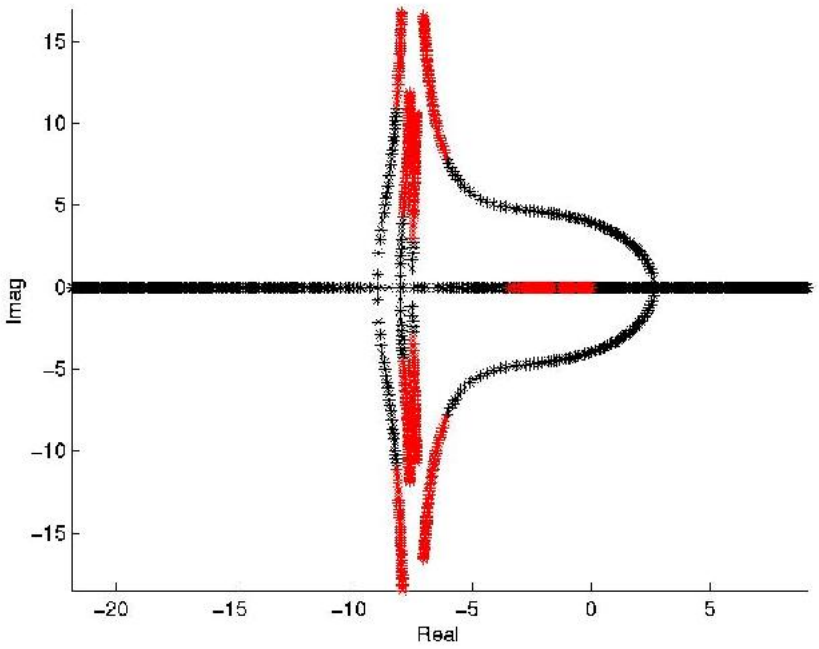

Figure 19: Robust root-locus of the F4E aircraft.

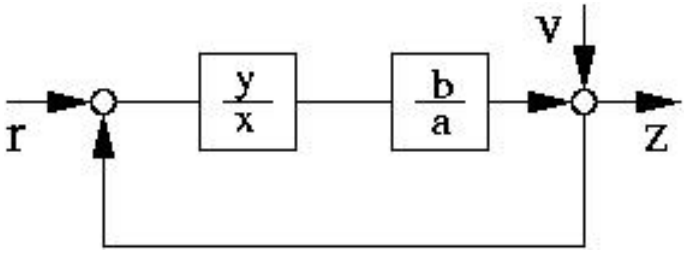

Figure 20: Standard feedback configuration.

the sensitivity of the control system output z to disturbances v is characterized by the sensitivity function

$$S = \frac{1}{1 + \frac{b}{a}\frac{y}{x}} = \frac{ax}{ax + by}$$

where plant polynomials a and b are given, and controller polynomials x and y must be found, see [Kwakernaak 93]. As shown in [Doyle et al. 92], robustness of the closed-loop plant to model uncertainty may be characterized by the complementary sensitivity function

$$T = 1 - S = \frac{by}{ax + by}$$

which is also the closed-loop system transfer function. As recalled in [Åström et al. 98], simplified yet sensible design specifications for a control law can be formulated as

$$\|S| \ < \gamma_S, \quad \|T\| < \gamma_T$$

where typical values of γ_S range between 1.2 and 2.0 and typical values of γ_T range between 1.0 and 1.5. This H_∞ control problem, as well as many others, can be formulated using the general paradigm proposed above.

12.1 Optimal robust stability

Consider the optimal robust stability problem of section 11.1 in [Doyle et al. 92], where the open-loop plant in figure 20 is given by

$$\frac{b}{a} = \frac{s-1}{(s+1)(s-0.5)}$$

and we seek a controller y/x minimizing γ_T under the following weighted H_∞ constraint on the closed-loop transfer function

$$\|WT\|_\infty = \left\| \left(\frac{s+0.1}{s+1} \right) \left(\frac{by}{ax+by} \right) \right\|_\infty < \gamma_T.$$

The following Matlab code seeks a first order controller for $\gamma_T = 1.9$:

```
a = (s+1)*(s-0.5); b = (s-1); gammaT = 1.9;
c = (s+0.1)*(s+1)^2*(s+3); % central polynomial
lmi = hinfdes([],'init',[1 1]); % seek first order controller
lmi = hinfdes(lmi,(s+0.1)*[0 b],(s+1)*[a b],c,gammaT); % H-inf spec
out = hinfdes(lmi,'solve'); % solve LMI
x = out(1); y = out(2);
```

Central polynomial c is the key design parameter, and together with upper bound γ_T they capture the whole degrees of freedom. Roots in c are just an indication on where closed-loop poles should be located: generally, roots of characteristic polynomial $ax + by$

will be located around roots of c, but they may also differ significantly due to structural constraints. The H_∞ design procedure then consists in iteratively playing with the roots of c, while lowering upper bound γ_T.

In table 3 we show different choices of roots for c, denoted by $\sigma(c)$ (4 roots = 2 for the open-loop system, 1 for the weighting function, 1 for the controller), together with actual poles of closed-loop transfer function T (3 roots) denoted by $\sigma(ax+by)$, upper bounds γ_T and the actual weighted norms $\|WT\|_\infty$ achieved by the computed controllers. Each design requires about 1 second of CPU time on our computer.

$\sigma(c)$	$\sigma(ax+by)$	γ_T	$\|WT\|_\infty$
-1,-1,-1,-1	$-1.04 \pm i1.08$, -0.230	2.9	2.11
-1,-1,-1,-0.1	$-0.731 \pm i0.566$, -0.118	2.3	1.74
-2,-1,-1,-0.1	$-1.133 \pm i0.586$, -0.114	2.1	1.54
-3,-1,-1,-0.1	$\mathbf{-1.383 \pm i0.642}$, **-0.0932**	**1.9**	**1.47**
-10,-1,-1,-0.1	-6.775, -1.063, -0.1059	1.8	1.31
-500,-1,-1,-0.1	-1700, -0.992, -0.103	1.7	1.21

Table 3: Optimal robust stability. Roots of central polynomial, characteristic polynomial, H_∞ upper bound and achieved H_∞-norm.

We can see that a good strategy is to start with a central polynomial with all its roots in -1, and a loose upper bound on γ_T. Decreasing γ_T, some closed-loop poles move away from -1, which gives indications on how to move roots of the central polynomial. At the bottom of the table, we can see that by allowing a very fast root in the central polynomial, γ_T can be decreased significantly close to the theoretical infimum of 1.20. Yet the closed-loop system also features a very fast pole, and the resulting controller $y/x = (-2046.2 - 2039.7s)/(3744.0 + s)$ results impractical.

A good tradeoff here is indicated in boldface letters in table 3, where a weighted H_∞-norm of 1.47 is achieved with the first-order controller

$$\frac{y}{x} = \frac{-3.0456 - 3.2992s}{5.6580 + s}.$$

Note however that the sensitivity function has very poor norm $\|S = 1 - T\|_\infty = 13.1$, due to the fact that no specifications were enforced on S. As a result, the above controller can be very sensitive to perturbations, or fragile, as pointed out in [Keel and Bhattacharyya 97].

A more sensible design approach would then enforce an additional H_∞ specification on S, such as

$$\|S\|_\infty = \left\| \frac{ax}{ax+by} \right\|_\infty < \gamma_S$$

for some suitable value of γ_S. However, as shown in [Åström 00], for this numerical example the ratio between the unstable open-loop pole and zero is small so there is no controller that will give a reasonably robust closed-loop system.

Adding a line to the above Matlab code to enforce an additional specification on $\|S\|_\infty$, we obtain (after about 2 seconds of CPU time) with $c(s) = (s+1)^3(s+100)$, $\gamma_T = 4$ and

$\gamma_S = 4$ the following first-order controller

$$\frac{y}{x} = \frac{-873.30 - 816.37s}{1202.4 + s}$$

producing $\|S\|_\infty = 3.44$ and $\|WT\|_\infty = 2.24$.

12.2 Flexible beam

Consider the flexible beam example of section 10.3 in [Doyle et al. 92]. The open-loop plant is given by

$$\frac{b}{a} = \frac{-6.4750s^2 + 4.0302s + 175.7700}{5s^4 + 3.5682s^3 + 139.5021s^2 + 0.0929s}$$

and we are seeking a controller y/x. For the closed-loop plant to approximate a standard second-order system with settling time at 1% of 8 seconds and overshoot less then 10%, the following frequency domain specification on the weighted sensitivity function is enforced:

$$\|WS\|_\infty = \left\| \frac{s^2 + 1.2s + 1}{s(s + 1.2)} \frac{ax}{ax + by} \right\|_\infty < \gamma_S.$$

Suppose we are looking for a second-order controller. The open-loop plant has poles 0, $-0.6660 \cdot 10^{-3}$, and $-0.3565 \pm i5.270$, and the weighting function has poles at 0 and -1.2. As explained in [Henrion 03], the central polynomial must mirror open-loop stable poles, so an initial choice of central polynomial features roots $-0.6660 \cdot 10^{-3}$, $-0.3565 \pm i5.270$, -1.2 plus two roots at -10^{-2} corresponding to the open-loop plant integrator and the weighting function integrator, plus two roots at -1 (arbitrary) corresponding to the controller poles. With this choice of central polynomial and $\gamma_S = 5$ the H_∞ LMI problem is solved in 15 seconds but the resulting step response is too slow.

After a series of attempts, an acceptable step response was obtained with the roots $\sigma(c) = \{-0.6660 \cdot 10^{-3}, -10^{-2}, -0.3565 \pm i5.270, -0.1, -1, -1, -1\}$ corresponding to the central polynomial $c(s) = 0.1858 \cdot 10^{-4} + 0.3000 \cdot 10^{-1}s + 3.178s^2 + 37.33s^3 + 94.06s^4 + 90.50s^5 + 33.45s^6 + 3.824s^7 + s^8$.

```
% Matlab script for the flexible beam example
b = (-6.4750*s^2+4.0302*s+175.77)/5; % plant numerator
a = (5*s^4+3.5682*s^3+139.5021*s^2+0.0929*s)/5; % monic plant denominator
lmi = hinfdes([],'init',[2 2]); % seek second-order controller
wn = s^2+1.2*s+1; wd = s*(s+1.2); % weighting function
c = (a+shift(a,-1)*1e-2)*(s+1)*(s+0.1)*(s+1)^2; % central polynomial
gammaS = 5; % H-inf upper bound
% H-inf spec |(wn*a*x)/(wd*(a*x+b*y))| < gammaS
lmi = hinfdes(lmi,wn*[a 0],wd*[a b],c,MS);
sol = hinfdes(lmi,'solve');
if ~isempty(sol) x = sol(1); y = sol(2); end;
```

With $\gamma_S = 5$ the above script returns the controller

$$\frac{y}{x} = \frac{0.77489 \cdot 10^{-4} + 0.16572 \cdot 10^{-1} s + 0.36537 s^2}{0.41025 \cdot 10^{-1} + 1.0437 s + s^2}$$

producing

$$\|S\|_\infty = 1.27, \quad \|T\|_\infty = 1.01$$

and a step response with settling time at 1% of 11.3 seconds and overshoot of 4%. Bode magnitude plots of S and T are given in figure 21, and the step response is shown in figure 22.

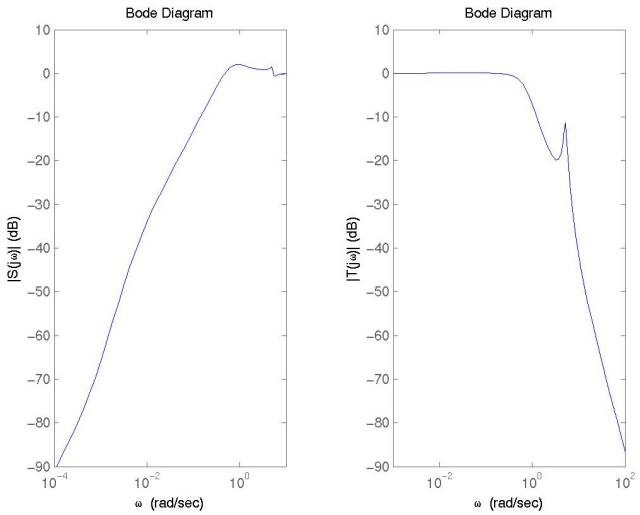

Figure 21: Flexible beam. Bode magnitude plots of S and T.

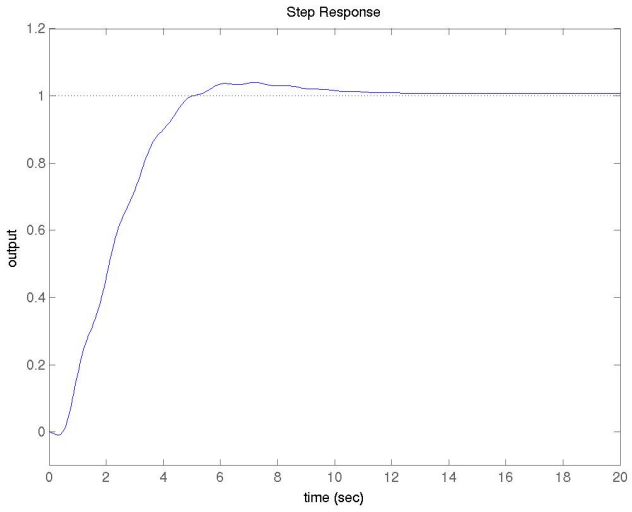

Figure 22: Flexible beam. Step response.

13 Extensions

Some directions for further work:

1. Maximization of uncertainty radius (instead of minimization of controller norm) in `ellides` and `ellides2` (easy)
2. Extend `ptopdes`, `ellides`, `ptopdes2`, `ellides2` to intersections of stability regions (already implemented in `ptopdes2` and `ellides2` but not yet documented) (easy)
3. Extend `elliana` to other stability regions (medium)
4. The current version of function `ptopana` is based on [Henrion et al. 01a], but it is likely that the more recent results of [Henrion et al. 02a] are less computationally demanding (medium)
5. Extend `ptopdes` to polynomial matrices, based on the results of [Henrion et al. 02a] (medium)
6. Implement functions `ptopspr` and `ellispr` to perform SPR design with polytopic and ellipsoidal uncertainty, based on [Henrion 02] and [Henrion 01] (medium)
7. Extend some analysis and design routines to LMI (or even QMI) stability regions, see [Henrion et al. 01b] (medium)
8. Extend `hinfdes` to multivariable systems (medium)
9. Implement `simdes`, simultaneous stabilization of a set of scalar plants with the cone complementarity LMI algorithm, see [Henrion et al. 99] (difficult)

Acknowledgment

This work was supported by the Grant Agency of the Czech Republic under Project No. 102/02/0709, and by the CNRS STIC Departement Young Researcher Grant No. 01N80/0474. Comments by Zdeněk Hurák, Michal Kvasnica, Thomas Gauchet and Dimitri Peaucelle were appreciated.

References

[Ackermann 93] J. Ackermann. Robust Control. Systems with Uncertain Physical Parameters. *Springer Verlag*, Berlin, 1993.

[Åström et al. 98] K. J. Åström, H. Panagopoulos, T. Hägglund. Design of PI controllers based on non-convex optimization. *Automatica*, Vol. 34, No. 5, pp. 585–601, 1998.

[Åström 00] K. J. Åström. Limitations on Control System Performance. *European Journal of Control*, Vol. 6, No. 1, pp. 2–20, 2000.

[Barb et al. 01] F. Dan Barb, A. Ben-Tal, A. Nemirovski. Robust dissipativity of interval uncertain systems. Technical report, Delft University of Technology, The Netherlands, 2001.

[Barmish 94] B. R. Barmish. New Tools for Robustness of Linear Systems. *MacMillan*, New York, 1994.

[Bhattacharyya 95] S. P. Bhattacharyya, H. Chapellat, L. H. Keel. Robust Control: The Parametric Approach, *Prentice Hall*, Upper Saddle River, 1995.

[Crisalle et al. 94] O. D. Crisalle, H. M. Mahon, D. Bonvin. Study of Robust Control Designs using the Critical Direction Method for Ellipsoidal Uncertainties. *Proceedings of the IEEE Southcon94 Conference*, pp. 173–180, Orlando, Florida, 1994.

[Datta et al. 97] B. N. Datta, S. Elhay, Y. M. Ram. Orthogonality and partial pole assignment for the symmetric definite quadratic pencil. *Linear Algebra and its Applications*, Vol. 257, pp. 29–48, 1997.

[Doyle et al. 92] J. C. Doyle, B. A. Francis and A. R. Tannenbaum. Feedback Control Theory. *MacMillan*, New York, 1992.

[El Ghaoui 99] L. El Ghaoui, S. I. Niculescu (Editors). Advances in Linear Matrix Inequality Methods in Control. *SIAM*, Philadelphia, 1999.

[Franklin et al. 86] G. J. Franklin, J. D. Powell and A. Emani-Naeini. Feedback Control of Dynamic Systems. *Addison-Wesley*, Reading, 1986.

[Henrion et al. 99] D. Henrion, S. Tarbouriech, M. Šebek. Rank-one LMI Approach to Simultaneous Stabilization of Linear Systems. *Systems and Control Letters*, Vol. 38, No. 2, pp. 79–89, 1999.

[Henrion 01] D. Henrion. Discrete Robust SPR Design via Semidefinite Programming. LAAS-CNRS Research Report No. 01150, Toulouse, France, March 2001.

[Henrion et al. 01a] D. Henrion, D. Arzelier, D. Peaucelle, M. Šebek. An LMI Condition for Robust Stability of Polynomial Matrix Polytopes. *IFAC Automatica*, Vol. 37, pp. 461–468, 2001.

[Henrion et al. 01b] D. Henrion, O. Bachelier, M. Šebek. D-Stability of Polynomial Matrices. *International Journal of Control*, Vol. 74, No. 8, pp. 845–856, 2001.

[Henrion et al. 01c] D. Henrion, D. Peaucelle, D. Arzelier, M. Šebek. Ellipsoidal Approximation of the Stability Domain of a Polynomial. *Proceedings of the European Control Conference*, Porto, Portugal, pp.384–389, September 2001.

[Henrion et al. 02a] D. Henrion, D. Arzelier, D. Peaucelle. Positive Polynomial Matrices and Improved LMI Robustness Conditions. *Proceedings of the IFAC World Congress on Automation*, Barcelona, Spain, July 2002. To appear in *Automatica*, 2003.

[Henrion et al. 02b] D. Henrion, M. Šebek, V. Kučera. An Algorithm for Static Output Feedback Simultaneous Stabilization of Scalar Plants. *Proceedings of the IFAC World Congress on Automation*, Barcelona, Spain, July 2002.

[Henrion 02] D. Henrion. LMIs for Robust SPR Design. *IEEE Transactions on Circuits and Systems, Part I: Fundamental Theory and Applications*, Vol. 49, No. 7, pp. 1017–1020, 2002.

[Henrion et al. 02c] D. Henrion, M.Šebek, V. Kučera. Positive Polynomials and Robust Stabilization with Fixed-Order Controllers. LAAS-CNRS Research Report No. 02325, Toulouse, France. To appear in *IEEE Transactions on Automatic Control*, 2003.

[Henrion et al. 02d] D. Henrion, M. Šebek, V. Kučera. Robust pole placement for second-order systems: an LMI approach. LAAS-CNRS Research Report No. 02324, July 2002.

[Henrion 03] D. Henrion. LMI optimization for fixed-order H_∞ controller design. To be registred as a LAAS-CNRS Research Report, February 2003.

[Howitt and Luus 91] G. D. Howitt, R. Luus. Simultaneous Stabilization of Linear Single-Input Systems by Linear State Feedback Control. *International Journal of Control*, Vol. 54, No. 4, pp. 1015–1030, 1991.

[Keel and Bhattacharyya 97] L. H. Keel, S. P. Bhattacharyya. Robust, fragile or optimal ? *IEEE Transactions on Automatic Control*, Vol. 42, No. 8, pp. 1098–1105, 1997.

[Kwakernaak 93] H. Kwakernaak. Robust control and H_∞ optimization – Tutorial Paper. *Automatica*, Vol. 29, No. 2, pp. 255–273, 1993.

[Nichols and Kautsky 01] N. K. Nichols, J. Kautsky. Robust eigenstructure assignment in quadratic matrix polynomials: nonsingular case. *SIAM Journal on Matrix Analysis and Applications*, Vol. 23, No. 1, pp. 77–102, 2001.

[Peaucelle et al. 01] D. Peaucelle, D. Henrion, Y. Labit. SeDuMi Interface: A user-friendly free Matlab package for defining LMI problems. *Proceedings of the IEEE Conference on Computer-Aided Control System Design*, Glasgow, Scotland, September 2002. See `www.laas.fr/~peaucell/SeDuMiInt.html`

[PolyX 00] PolyX, Ltd. The Polynomial Toolbox for Matlab. Prague, Czech Republic. Version 2.5 realeased in 2000. See `www.polyx.cz`

[Sturm 99] J. F. Sturm. Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, Vol. 11–12, pp. 625–653, 1999. See `fewcal.kub.nl/sturm/software/sedumi.html`

[Tisseur and Higham 01] F. Tisseur, N. J. Higham. Structured pseudospectra for polynomial eigenvalue problems, with applications. *SIAM Journal on Matrix Analysis and Applications*, Vol. 23, No. 1, pp. 187–208, 2001.

[Tong and Sinha 94] Y. Tong and N. K. Sinha. A Computational Technique for the Robust Root Locus. *IEEE Transactions on Industrial Electronics*, Vol. 41, No. 1, pp. 79–85, 1994.