Solving strict polynomial inequalities by Bernstein expansion

J. Garloff and B. Graf

Dedicated to Professor Dr. Karl Nickel on the occasion of his 75th birthday

1 Introduction

Many interesting control system design and analysis problems can be recast as systems of inequalities for multivariate polynomials in real variables. In particular, for linear time-invariant systems, important control issues such as robust stability and robust performance can be reduced to such systems. Typically, the variables in the (multivariate) polynomials come from plant (controlled system) and compensator (controller) parameters. In this chapter, we describe a method for solving such systems of inequalities. By solving we mean that we end up with a collection of axis-parallel boxes in the parameter space whose union provides an inner approximation of the solution set, i.e., the polynomial inequalities are fulfilled for each parameter vector taken from such a box. This method is based on the expansion of a multivariate polynomial into Bernstein polynomials. It provides an alternative to symbolic methods like quantifier elimination whose application to control problems was demonstrated in [1]. The number of operations required by quantifier elimination methods is still doubly exponential in the number of variables, so that only relatively small problems can actually be solved, whereas Bernstein expansion has been applied to larger robust stability problems [2, 3, 4]. However, it should be noted that in contrast to symbolic methods Bernstein expansion requires a priori bounds on the design parameter range. This is not a hard restriction since the designer often can estimate the interesting parameter range.

We mention a third approach, the *probabilistic approach*, e.g.,[5, 6], to solve problems in control theory which can be formulated as systems of strict inequalities. Here again bounds on the parameter range must be known. This approach is applicable to very complex systems but it provides only 'probabilistic' answers.

Notation: For compactness, we define a *multi-index* I as an ordered l-tupel of nonnegative integers $(i_1, \ldots, i_l)$. We will use multi-indices e.g. to shorten power products: For $x = (x_1, \ldots, x_l) \in \mathbf{R}^l$ we set $x^I = x_1^{i_1} x_2^{i_2} \cdot \ldots \cdot x_l^{i_l}$. For simplicity, we sometimes suppress the brackets in the notation of multi-indices. We write $I \leq N$ if $N = (n_1, \ldots, n_l)$ and if $0 \leq i_k \leq n_k$, $k = 1, \ldots, l$. Further, let $S = \{I : I \leq N\}$. Then we can write an l-variate polynomial p in the form

$$p(x) = \sum_{I \in S} a_I x^I, \quad x \in \mathbf{R}^l, \tag{1}$$

and refer to N as the *degree* of p and to

$$\hat{n} = \max\{n_i : i = 1, \ldots, l\}. \tag{2}$$

as the *total degree* of p. Also, we write I/N for $(i_1/n_1, \ldots, i_l/n_l)$ and $\binom{N}{I}$ for $\binom{n_1}{i_1} \cdot \ldots \cdot \binom{n_l}{i_l}$.

Problem statement: Let $p_1, \ldots, p_n$ be l-variate polynomials and let an axis-parallel box Q in the $\mathbf{R}^l$ be given. We want to find

$$\Sigma := \{x \in Q : p_i(x) > 0, i = 1, \ldots, n\}; \tag{3}$$

the set Σ is called the *solution set* of the system of polynomial inequalities.

This chapter is organized as follows: The next section contains a short review of quantifier elimination methods and their application to control problems. In Section 3 we recall the Bernstein expansion and apply it to the solution of systems of polynomial inequalities in Section 4. Our algorithm is explained in Section 5. Numerical examples are presented in Section 6 and conclusions are given in Section 7.

2 Quantifier Elimination

In many practical control problems some of the polynomial variables are quantified by the logic quantifier $\forall$ (for all) or $\exists$ (there exists). Typically, $\forall$ quantifies the plant parameters (for robust design) and $\exists$ quantifies the controller parameters (to define the feasible controller-parameter set). In addition, the polynomial inequalities are combined by the Boolean operators $\wedge$ (and) and $\vee$ (or). Examples from control theory can be found in [1, 7, 8].

The problem to find an equivalent expression involving only unquantified variables is called the *quantifier elimination* (QE) *problem*. In 1948, Tarski [9] showed that there is a procedure that solves this problem in a finite number of steps. Although Tarski gave a constructive proof, the resulting algorithm is inpractical even with the power of the today's computers. One of the first attempts to use QE methods to solve control design problems was made in 1975 by Anderson et al. [10] to solve the static output-feedback stabilization problem. However, the computational complexity and lack of software severely limited the interest in their results. In 1975, Collins [11] introduced a more efficient approach, the *cylindrical algebraic decomposition* (CAD). For an introduction to CAD see the excellent exposition [8]. Given a set of l-variate polynomials, the CAD algorithm decomposes the $\mathbf{R}^l$ into components over which the polynomials have constant signs. For systems of polynomial inequalities (and even equations) the CAD method has value of its own: Once a decomposition is found, a solution (if there exists one) to any system determined by the given polynomials can be found. However, utilizing this method the QE problem could be solved only for very small problems.

In [12, 13, 14] significantly more efficient QE algorithms based on partial CAD were presented. Software packages have been written for the implementation of the new algorithms, e.g., the software package QEPCAD for *quantifier elimination by partial cylindrical algebraic decomposition* by Hoon Hong from the Research Institute for Symbolic Computation in Linz (Austria) with contributions by G.E. Collins, J.R. Johnson, and M.J. Encarnación.

The CAD algorithms always completely solve any QE problem. However, the number of operations required is still doubly exponential (for details cf. [15]) so that only problems with modest size can be handled. We mention two papers [16, 17] treating the special case of systems of strict polynomial inequalities – the proper subject of this chapter. In [16, 17] algorithms are described which allow to decide whether such a system has a solution and which are much faster than the general CAD algorithm. The simplified CAD algorithm in [17] finds a finite set of solutions such that any other solution can be connected by a continuous path of solutions with one of the solution set. We were told from Wolfram Research, Inc., that this algorithm is included in the 3.0 version of *Mathematica* [18] in the Standard Add-on Package [19] covering manipulating and solving algebraic inequalities and that the upcoming version of *Mathematica* will contain in the kernel algorithms for deciding existence of solutions of systems of polynomial equations and inequalities based on CAD and QE methods.

3 Bernstein Expansion

3.1 Bernstein Transformation of a Polynomial

In this subsection we expand a given multivariate polynomial (1) into Bernstein polynomials to obtain bounds for its range over an l-dimensional box. This approach was used in the univariate case for the first time in [20] and subsequently in a series of papers, e.g., [21, 22]. Generalizations to the multivariate case were given in [23, 24, 25, 26]. For a nearly complete bibliography see [27]. Without loss of generality we consider the unit box $U = [0,1]^l$ since any nonempty box of $\mathbf{R}^l$ can be mapped affinely onto this box.

The ith *Bernstein polynomial* of degree n is defined as

$$b_{n,i}(x) = \binom{n}{i} x^i (1-x)^{n-i}, \quad 0 \le i \le n,$$

for an arbitrary $x \in \mathbf{R}$. In the multivariate case, the Ith Bernstein polynomial of degree N is defined by

$$B_{N,I}(x) = b_{n_1,i_1}(x_1) \cdot \ldots \cdot b_{n_l,i_l}(x_l), \quad x = (x_1, \ldots, x_l) \in \mathbf{R}^l. \tag{4}$$

The transformation of a polynomial from its power form (1) into its *Bernstein form* results in

$$p(x) = \sum_{I \in S} b_I(U) B_{N,I}(x), \tag{5}$$

where the *Bernstein coefficients* $b_I(U)$ of p on U are given by

$$b_I(U) = \sum_{J \le I} \frac{\binom{I}{J}}{\binom{N}{J}} a_J, \quad I \in S. \tag{6}$$

We collect the Bernstein coefficients in an array $B(U)$, i.e., $B(U) = (b_I(U))_{I \in S}$. A similar notation will be employed for other sets of related coefficients. In [23] a method was presented for calculating the Bernstein coefficients efficiently by a difference table scheme (which is similar to the sweep procedure, cf. Sect. 3.2) that avoids the binomial coefficients and products appearing in (6).

In the following, we will use a special subset of the index set S comprising those indices which correspond to the indices of the vertices of the array $B(U)$, i.e.,

$$S_0 = \{0, n_1\} \times \cdots \times \{0, n_l\}.$$

We list two useful properties of the Bernstein coefficients, e.g., [22, 23, 28].

3.1. Lemma. *Let p be a polynomial (1) of degree N. Then the following properties hold for its Bernstein coefficients $b_I(U)$ (6):*

(i) Sharpness of special coefficients:

$$\forall\ I \in S_0 : b_I(U) = p(I/N) \tag{7}$$

(ii) Convex hull property:

$$\forall\ x \in U : \min_{I \in S} b_I(U) \le p(x) \le \max_{I \in S} b_I(U) \tag{8}$$

with equality in the left (resp., right) inequality if and only if $\min_{I \in S} b_I(U)$ *(resp.,* $\max_{i \in S} b_I(U)$*) is assumed at a Bernstein coefficient $b_I(U)$ with $I \in S_0$.*

Formula (7) follows immediately from (6). Property (8) relies on two fundamental properties of the Bernstein polynomials, viz. their nonnegativity on the unit box U and the fact that they form a partition of unity.

3.2 Sweep Procedure

In this subsection we follow the exposition in [4]. We define a sweep in rth direction ($1 \leq r \leq l$) similarly to de Casteljau's algorithm in Computer Aided Geometric Design, e.g., [28], as recursively applied linear interpolation. Let D be any subbox of U generated by sweep operations (at the beginning we have $D = U$, then subsequently D is obtained by successively halving). Starting with $B^{(0)}(D) = B(D)$ we set for $k = 1, \ldots, n_r$

$$b^{(k)}_{i_1,\ldots,i_r,\ldots,i_l}(D) = \begin{cases} b^{(k-1)}_{i_1,\ldots,i_r,\ldots,i_l}(D) & : i_r = 0, \ldots, k-1 \\ \frac{1}{2}(b^{(k-1)}_{i_1,\ldots,i_r-1,\ldots,i_l}(D) + b^{(k-1)}_{i_1,\ldots,i_r,\ldots,i_l}(D)) & : i_r = k, \ldots, n_r. \end{cases} \quad (9)$$

To obtain the new coefficients, we apply formula (9) for $i_j = 0, \ldots, n_j$, $j = 1, \ldots, r-1, r+1, \ldots, l$.

Then the Bernstein coefficients on D_0 [1], where the subbox D_0 is given by

$$D_0 = [\underline{d}_1, \bar{d}_1] \times \cdots \times [\underline{d}_r, (\underline{d}_r + \bar{d}_r)/2] \times \cdots \times [\underline{d}_l, \bar{d}_l],$$

are obtained as $B(D_0) = B^{(n_r)}(D)$. At no extra cost we get as intermediate values the Bernstein coefficients $B(D_1)$ on the neighbouring subbox D_1[1]

$$D_1 = [\underline{d}_1, \bar{d}_1] \times \cdots \times [(\underline{d}_r + \bar{d}_r)/2, \bar{d}_r] \times \cdots \times [\underline{d}_l, \bar{d}_l]$$

[1] i.e., the Bernstein coefficients of the polynomial shifted from this subbox to U

since for $k = 0, \ldots, n_r$ the following relation holds [24]: $b_{i_1,\ldots,n_r-k,\ldots,i_l}(D_1) = b^{(k)}_{i_1,\ldots,n_r,\ldots,i_l}(D)$.

It is important to note that by the sweep procedure the explicit transformation of the subboxes generated by the sweeps back to U is avoided. Let $\hat{n}$ denote the total degree (2) of polynomial (1). Since we have to perform formula (9) $n_r(n_r+1)/2$ times, we need altogether $O(\hat{n}^{l+1})$ additions and multiplications.

3.3 Selection of the Sweep Direction

The definition of the sweep procedure shows that we are free in choosing the sweep direction. In order to increase the probability for finding a nonpositive sharp Bernstein coefficient proving that the polynomial under consideration is not positive, we suggest to sweep in that coordinate direction in which the first partial derivative is largest. Our selection rule profits from the easy calculation of the partial derivatives of a polynomial in Bernstein form, e.g., [28, 29].

To shorten some expressions in the sequel we associate with an index $I = (i_1, \ldots, i_r, \ldots, i_l)$ the index $I_{r,k} = (i_1, \ldots, i_r + k, \ldots, i_l)$, where $0 \leq k + i_r \leq n_r$. Then the first partial derivative with respect to x_r of p (5) is given by the following formula ($1 \leq r \leq l$):

$$\frac{\partial p}{\partial x_r}(x) = n_r \sum_{I \leq N_{r,-1}} [b_{I_{r,1}}(D) - b_I(D)] B_{N_{r,-1},I}(x).$$

To decide which sweep direction to choose we estimate

$$\max_{x \in D} \left| \frac{\partial p}{\partial x_r}(x) \right|$$

from above by

$$\tilde{I}_r = \max_{I \leq N_{r,-1}} n_r \, |b_{I_{r,1}}(D) - b_I(D)|.$$

We choose that r_0 with maximum value

$$\tilde{I}_{r_0} = \max_{j=1,\ldots,l} \tilde{I}_j. \tag{10}$$

4 Approximation of the Solution Set

In this section we use a similar approach as in [2, 30]. The algorithm which we will describe in the next section was applied in [31] to approximate the stability region of a polynomial family with polynomial parameter dependency. Since we are able to describe the solution set Σ only in simplest cases, we are seeking for a good approximation to it. We obtain an inner approximation of Σ by the union of some subboxes of Q on which all polynomials p_i are positive. Similarly, an outer approximation is given by the union of some subboxes of Q with the property that on each there is a polynomial p_i being nonpositive there. The boundary $\partial\Sigma$ of Σ can be approximated by the union of some subboxes of Q on which each polynomial p_i assumes positive as well as nonpositive values, cf. Figure 1.

Figure 1: Inner and outer approximation of Σ and approximation of $\partial\Sigma$ (Figure taken from [32])

By Σ_i and Σ_o we denote the set of subboxes which provide the inner and outer approximation, respectively. For a fixed positive number ε, $\Sigma_b(\varepsilon)$ is the list of subboxes with volume less than ε the union of which approximates the boundary $\partial\Sigma$. In our algorithm we check (non)positivity of a polynomial by the sign of its Bernstein coefficients. From Lemma 3.1 we obtain immediately:

4.1. Lemma. Positivity test of a multivariate polynomial

Let p be an l-variate polynomial and let b_I be its Bernstein coefficients on Q. Then it holds:

$$\min_{I \in S} b_I > 0 \Longrightarrow \; p(x) > 0 \;\forall\; x \in Q, \tag{11}$$

$$\max_{I \in S} b_I \leq 0 \Longrightarrow \; p(x) \leq 0 \;\forall\; x \in Q, \tag{12}$$

$$\exists\; I \in S_0 : b_I > 0 \Longrightarrow \; \exists\; x \in Q : \; p(x) > 0, \tag{13}$$

$$\exists\ I \in S_0 : b_I \leq 0 \Longrightarrow\ \exists\ x \in Q :\ p(x) \leq 0. \tag{14}$$

According to Lemma 4.1, the sets Σ_i, Σ_o, and $\Sigma_b(\varepsilon)$ consist of the subboxes $\tilde{Q}$ generated by sweeps which fulfil the following conditions:

Σ_i : The Bernstein coefficients $(b_I^{(i)})_{I\in S}$ of all polynomials $p_i, i = 1, 2, \ldots, n$, are positive; then by (11) all polynomials $p_i, i = 1, \ldots, n$, are positive on $\tilde{Q}$.

Σ_o : The Bernstein coefficients $(b_I^{(i^*)})_{I\in S}$ of a polynomial p_{i^*} are all nonpositive; then by (12) the polynomial p_{i^*} is nonpositive on $\tilde{Q}$.

$\Sigma_b(\varepsilon)$: The volume of $\tilde{Q}$ is less than ε and each polynomial p_i possesses sharp positive and nonpositive Bernstein coefficients; then according to (13) and (14), each polynomial p_i assumes on $\tilde{Q}$ positive and nonpositive values. [2]

Of course, if it turns out that a polynomial p_i is positive over a subbox $\tilde{Q}$ of Q we can discard this polynomial from the list of polynomials to be checked further for positivity on any subbox of $\tilde{Q}$.

5 The Algorithm

The procedure **Test** checks a subbox $\tilde{Q}$ of Q to which list this box will be appended. This procedure returns *AP* (for *all positive*) if $\tilde{Q}$ will be added to Σ_i, *EN* (for *exists a nonpositive polynomial*) if it will belong to Σ_o, and *UD* (for *undecided*) if $\tilde{Q}$ will be appended to $\Sigma_b(\varepsilon)$.

The procedure **TerminateSearch** terminates the search for subboxes of Q on which the polynomials p_i are positive or nonpositive if all subboxes have a volume less than

[2] In order to avoid to introduce a fourth list, we collect for simplicity in $\Sigma_b(\varepsilon)$ all subboxes generated by sweeps having volume less than ε which belong neither to Σ_i nor to Σ_o.

ε. For a fixed recursion depth d (i.e., a fixed number of sweeps performed on the initial box), we obtain boxes with volume $2^{-d}\,\mathrm{vol}(Q)$.

On the other hand, to achieve boxes with volume less than ε we have to choose the depth d as the smallest integer number greater than

$$\frac{\ln(\mathrm{vol}(Q)) - \ln(\varepsilon)}{\ln(2)}.$$

In our algorithm we use a maximum recursion depth d which leads to parameter boxes with volume less than a given ε.

The main procedure **SolutionSet** returns a collection of subboxes on which all polynomials p_i are nonnegative. These subboxes are listed in a stack Σ_i. The stack BC consists of the Bernstein coefficients of the polynomials $p_i, i = 1, \ldots, n$, denoted by $B(D)$. We assume that the standard operations *MakeStack, Push, Pop,* and *Isempty* are implemented, e.g., [33, 34].

Procedure 1 SolutionSet($B(D)$)

```
begin  BC =MakeStack(); Σ_i = MakeStack();
       Push(BC,B(D));
  while (¬ Isempty(BC)) do
    B(D) = Pop(BC);
    {B(D_0), B(D_1)} = Sweep(B(D));
    t0=Test(B(D_0)); t1=Test(B(D_1));
    if (t0=UD) then
      if (¬ TerminateSearch()) then
        Push(BC,B(D_0));
      end if
    else if (t0=AP) then
      Push(Σ_i,D_0);
    end if
    if (t1=UD) then
      if (¬ TerminateSearch()) then
        Push(BC,B(D_1));
      end if
    else if (t1=AP) then
      Push(Σ_i,D_1);
    end if
  end while
end
```

6 Examples

All examples were run (on a PC equipped with a Pentium 133) with the maximum recursion depth $d = 15$. For simplicity, we apply the sweep selection rule (10) only to the first of the list of the polynomials so that the direction of the sweeps is completely determined by this polynomial. We denote by $\square\,\Sigma$ the smallest box in Q containing Σ.

The first two examples utilize one of the Liénard-Chipart stability criteria, cf. p. 155 in [35], which states that a polynomial

$$p(s) = a_0 s^m + a_1 s^{m-1} + \ldots + a_{m-1} s + a_m \quad \text{with } a_0 > 0$$

is stable, i.e., all its zeros have negative real parts, if and only if

$$a_m > 0, a_{m-2} > 0, a_{m-4} > 0, \ldots,$$

and

$$\Delta_1 > 0, \Delta_3 > 0, \Delta_5 > 0, \ldots,$$

where Δ_j is the leading principal minor of order j of the Hurwitz matrix, i.e.,

$$\Delta_j = \det(h_{ik})_{i,k=1,\ldots,j}$$

with

$$h_{ik} = a_{2k-i}, \quad i, k = 1, \ldots, m,$$

where by convention $a_r = 0$ if $r < 0$ or $r > m$.

Example 1: Our first example involves only two parameters, i.e., $l = 2$, so that we are able to visualize the approximations Σ_i, Σ_o, and $\Sigma_b(\varepsilon)$ obtained by our algorithm. We consider the static output-feedback problem presented in [10], cf. [1], which leads to the problem to find the set of all parameters v, w such that the closed-loop characteristic polynomial

$$p(s) = s^3 + vs^2 + (w - 5v - 13)s + w.$$

is stable. The Liénard-Chipart criterion gives us the conditions

$$\begin{aligned} v, w &> 0, \\ -5v^2 - 13v + vw - w &> 0. \end{aligned}$$

The results for $v \in [2, 10], w \in [40, 50]$ are presented in Figure 2. The white region is the inner approximation and the grey domain is the outer approximation to Σ. The approximation of $\partial\Sigma$ is given by the black region. The algorithm finds $\square\,\Sigma = [2, 5.59375] \times [41.9922, 50]$ in 0.27 s.

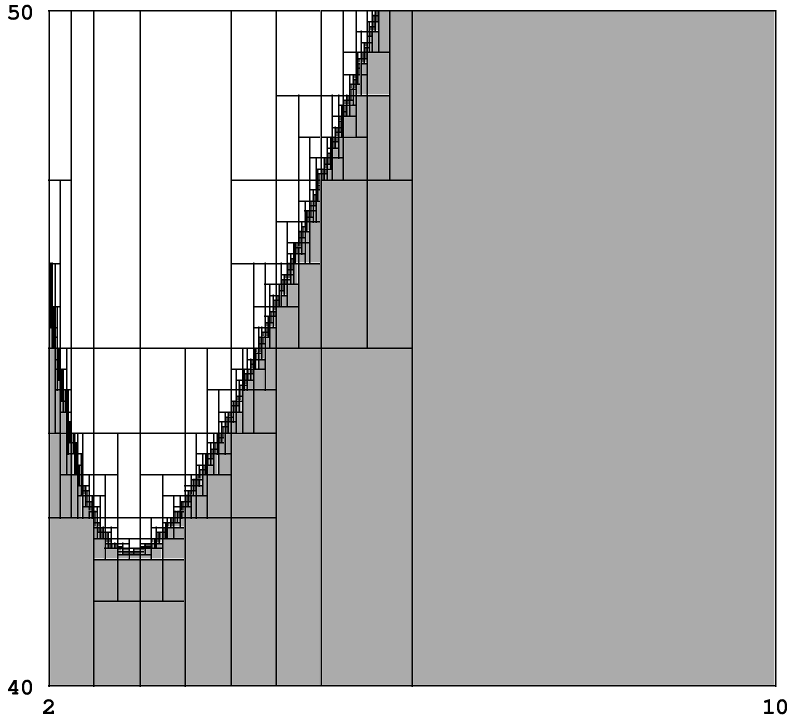

Figure 2: Solution set of Example 1.

Example 2: The problem taken from [1] is to find a (stable) second-order compensator with three parameters which simultaneously stabilize the three different plants with the following transfer functions

$$\frac{2-s}{(s^2-1)(s+2)}\,,\ \frac{2-s}{s^2(s+2)}\,,\ \frac{2-s}{(s^2+1)(s+2)}.$$

In order to reduce the number of parameters to be considered we assume a second-order compensator of the form

$$\frac{A(s+B)^2}{(s+D)^2}$$

with $D > 0$. To achieve stability, we utilize the Liénard-Chipart criterion. After some simplifications, we obtain the following set of inequalities in the parameters A, B, and D :

$$A, B, D > 0$$

$$AB^2 - D^2 > 0$$

$$-AB + A + D^2 - D - 1 > 0$$

$$AB - AD - 2A + D^3 + 4D^2 + 4D > 0$$

$$AB^3 - AB^2D - 4AB^2 + 2ABD + 4AB + 2BD^3 + 5BD^2 + 2BD - D^3 - 4D^2 - 4D > 0$$

$$AB - 2A - BD^2 - 4BD - 4B + 2D^2 + 3D - 2 > 0$$

We have choosen $A \in [100, 120], B \in [0, 2], D \in [10, 20]$. Note that the software package QEPCAD, cf. Sect. 2, needs already 2 hours of CPU time to solve the existence problem, i.e., to show that there is a solution of the above system of inequalities [1]. In 1.3 s our algorithm finds $\square\,\Sigma = [100, 120] \times [1.15625, 1.60938] \times [12.1875, 17.3438]$. Figure 3 shows the set of acceptable values of B and D for fixed $A = 110$ obtained in 0.5 s.

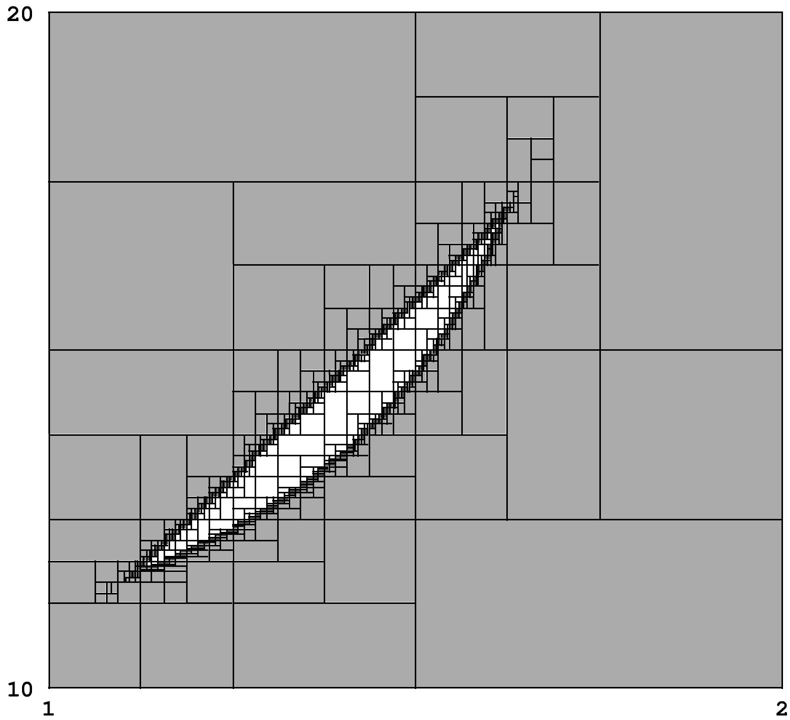

Figure 3: The set of acceptable values of B, D for $A = 110$.

Example 3: [2] We consider a plant which is assumed to be an unstable first-order system with transfer function

$$\frac{p_1}{1 - s/p_2},$$

where p_1 and p_2 are uncertainty parameters. To control the plant a proportional plus integral (PI) compensator is used with transfer function

$$C(s, q) = q_1 \frac{1 + s/q_2}{s},$$

where q_1 and q_2 are the design parameters. The controller to be designed has to meet some performance specifications represented by the following polynomial inequalities:

(1.) Closed loop stability: $\quad q_2 > 0, -q_1 > 0, -p_1 q_1 - q_2 > 0$

(2.) Tracking error (at steady state) for unit ramp: $p_1^2q_1^2 - 2500 > 0$

(3.) Closed loop bandwith: $aw^4 + bw^2 + c$, where

$$a = -q_2^2$$

$$b = p_1^2p_2^2q_1^2 - 2p_1p_2q_1q_2^2 - 2p_1p_2^2q_1q_2 - p_2^2q_2^2$$

$$c = p_1^2p_2^2q_1^2q_2^2$$

(4.) Resonance peak of the closed loop transfer function: $aw^4 + bw^2 + c$, where

$$a = 1.96q_2^2$$

$$b = 0.96p_1^2p_2^2q_1^2 + 1.96p_2^2q_2 + 3.92p_1p_2^2q_1q_2 + 3.92p_1p_2q_1q_2^2$$

$$c = 0.96p_1^2p_2^2q_1^2q_2^2$$

(5.) Control effort: $aw^4 + bw^2 + c$, where

$$a = 400q_2^2 - q_1^2,$$

$$b = 400p_1^2p_2^2q_1^2 + 800p_1p_2^2q_1q_2 + 800p_1p_2q_1q_2^2 - p_2^2q_1^2 + 400p_2^2q_2^2 - q_1^2q_2^2,$$

$$c = 400p_1^2p_2^2q_1^2q_2^2 - p_2^2q_1^2q_2^2.$$

The design parameters (q_1, q_2) are taken from $[-300, 0] \times [0, 15]$, the plant parameters p_1, p_2 are chosen from $[0.8, 1.25]$, and the variable w varies in $[0, 300]$. Our algorithm found in 8.436 s the following parameter intervals:

$q_1 \in [-300, -56.25], q_2 \in [0.11, 15], p_1, p_2 \in [0.8, 1.25], w \in [0, 18.75]$. We note that the software package QEPCAD was not able to solve this problem [7]. For the solution using QEPCAD of the simplified model involving only the design parameter q_1 with the simple output feedback $C(s, q) = q_1$ see [7].

7 Conclusions

Bernstein expansion provides a method for testing a multivariate polynomial for positivity over a box and therefore for finding an inner approximation of the solution set of a system of strict polynomial inequalities. Compared to quantifier elimination methods, Bernstein expansion is not so widely applicable:

- Only strict inequalities can be handled. However, many problems in linear control theory can be reduced to such systems.

- Bernstein expansion requires a priori bounds on the parameter range. However, the designer has often a region of special interest.

The applicability of quantifier elimination methods is severely limited by the number of the variables. So many problems of practical importance are beyond the capabilities of these methods. The development of both better algorithms and of fast algorithms for special classes of problems in a very active area for research so that it is hoped that the solution of significantly more complicated problems will be possible in near future. Bernstein expansion can handle presently more complex problems. But its efficiency drastically decreases if the number of parameters exceeds about seven. Quantifier elimination provides an explicit description of the solution set which is complicated in general. From the point of view of the designer the description of the entire solution set is often not necessary. What the designer really wants is a good inner approximation of the solution set or even only a large box inside this set. But that is what Bernstein expansion provides.

Acknowledgement

Support from the Ministry of Science and Research Baden-Württemberg and from the Ministry of Education, Science, Research, and Technology of the Federal Republic of Germany under the contract no. 1704596 is gratefully acknowledged.

References

1. ABDALLAH, C., DORATO, P., LISKA, R., STEINBERG, S., and YANG, W.: 'Applications of quantifier elimination theory to control theory'. Technical report EECE95-007, University of New Mexico, Department of Electrical and Computer Engineering, Albuquerque, 1995, cf. Proceedings of the 4th IEEE Mediteranean Symposium on Control and Automation, Maleme, Crete, Greece, 1996, pp. 340-345

2. FIORIO, G., MALAN, S., MILANESE, M., and TARAGNA, M.: 'Robust performance design of fixed structure controllers for systems with uncertain parameters', Proc. of the 32nd Conference on Decision and Control, San Antonio, Texas, 1993, pp. 3029-3031

3. GARLOFF, J., GRAF, B., and ZETTLER, M.: 'Speeding up an algorithm for checking robust stability of polynomials', *in* BANYASZ, Cs. (Ed.):' Proceedings of the 2nd IFAC Symposium on Robust Control Design' (Elsevier Sci., Oxford, 1998) pp. 183-188

4. ZETTLER, M., and GARLOFF, J.: 'Robustness analysis of polynomials with polynomial parameter dependency using Bernstein expansion', *IEEE Trans. Automat. Contr.*, 1998, **43**, (3), pp. 425-431

5. BAI, E.-W., TEMPO, R., and FU, M.: 'Worst-case properties of the uniform distribution and randomized algorithms for robustness analysis', Proc. of the American Control Conference, Albuquerque, NM, 1997

6. YOON, A., and KHARGONEKAR, P.: 'Computational experiments using randomized algorithms for robust stability analysis', submitted to *Automatica*

7. DORATO, P., YANG, W., and ABDALLAH, C.: 'Robust multi-objective feedback design by quantifier elimination', *J. Symbolic Comput.,* 1997, **24**, (2), pp. 153-159

8. JIRSTRAND, M.: 'Algebraic methods for modeling and design in control', *Linköping Studies in Science and Technology* Thesis No. 540 (Linköping University, Linköping, Schweden, 1996)

9. TARSKI, A.: 'A decision method for elementary algebra and geometry' (University of California Press, Berkeley, 1951, 2nd edn.)

10. ANDERSON, B.D.O., BOSE, N.K., and JURY, E.I.: 'Output feedback stabilization and related problems – solution via decision methods', *IEEE Trans. Automat. Contr.,* 1975, AC-**20**, (1), pp. 53-65

11. COLLINS, G.E.: 'Quantifier elimination for real closed fields by cylindrical algebraic decomposition', *in* BRAKHAGE, H.(Ed.): '2nd GI Conf. Automation Theory and Formal Languages', *Lecture Notes in Computer Science* vol. 33 (Springer, Berlin, 1975) pp. 134-183

12. HONG, H.: 'Improvements in CAD-based quantifier elimination', Ph. D. Dissertation (Ohio State University, Ohio, 1990)

13. COLLINS, G.E., HONG, H.: 'Partial cylindrical algebraic decomposition for quantifier elimination', *J. Symbolic Comput.,* 1991, **12**, (3), pp. 299-328

14. HONG, H.: 'Simple solution formula construction in cylindrical algebraic decomposition based quantifier elimination', *in* WANG, P.S. (Ed.): 'ISSAC'92, Intern. Symp. Symbolic and Algebraic Computation' (ACM Press, New York, 1992) pp. 177-188

15. BASU, S., POLLACK, R., and ROY, M.-F.: 'On the combinatorial and algebraic complexity of quantifier elimination', *in* GOLDWASSER, S. (Ed.): Proceedings 35th Annual Symposium of the IEEE Computer Soc. on *Foundations of Computer Science*, Santa Fe, New Mexico, 1994, pp. 632-641

16. McCALLUM, S.: 'Solving polynomial strict inequalities using cylindrical algebraic decomposition', *The Computer Journal*, 1993, **36**, (5), pp. 432-438

17. STRZEBONSKI, A.W: 'An algorithm for systems of strong polynomial inequalities', *The Mathematica J.*, 1994, **4**, (4), pp.74-77

18. WOLFRAM, S.: 'The Mathematica book' (Cambridge University Press, Cambridge, 1996, 3rd edn.)

19. WOLFRAM RESEARCH, INC.: 'Mathematica 3.0 Standard Add-on Packages (Cambridge University Press, Cambridge, 1996)

20. CARGO, G.T., and SHISHA, O.: 'The Bernstein form of a polynomial', *J. Res. Nat. Bur. Standards Sect. B*, 1966, **70B**, pp. 79-81

21. LANE, J.M., and RIESENFELD, R.F.: 'Bounds on a polynomial', *BIT*, 1981, **21**, pp. 112-117

22. RIVLIN, T.J.: 'Bounds on a polynomial', *J. Res. Nat. Bur. Standards Sect. B*, 1970, **74B**, pp. 47-54

23. GARLOFF, J.: 'Convergent bounds for the range of multivariate polynomials', *in* NICKEL, K. (Ed.): 'Interval mathematics 1985', *Lecture Notes in Computer Science* vol. 212 (Springer, Berlin, 1986), pp. 37-56

24. GARLOFF, J.: 'The Bernstein algorithm', *Interval Computations*, 1993, **2**, pp. 154-168

25. LANE, J.M., and RIESENFELD, R.F.: 'A theoretical development for the computer generation and display of piecewise polynomial surfaces', *IEEE Trans. Pattern Anal. Mach. Intel.*, 1980, **2**, pp. 35-46

26. MALAN, S., MILANESE, M., TARAGNA, M., and GARLOFF J.: 'B^3 algorithm for robust performances analysis in presence of mixed parametric and dynamic perturbations', Proc. of the 31st Conference on Decision and Control, Tucson, Arizona, 1992, pp. 128-133

27. HUNGERBÜHLER, R., and GARLOFF, J.: 'Bounds for the range of a bivariate polynomial over a triangle', *Reliable Computing*, 1998, **4**, (1), pp 3-13

28. FARIN, G.: 'Curves and surfaces for CAGD. A practical guide' (Academic Press, New York, 1993, 3rd edn.)

29. FAROUKI, R.T., and RAJAN, V.T.: 'Algorithms for polynomials in Bernstein form', *Computer Aided Geometric Design*, 1988, **5**, pp. 1-26

30. CANALE, M., FIORIO, G., MALAN, S., MILANESE, M., and TARAGNA, M.: 'Robust tuning of PID controllers via uncertainty model identification', Proc. 1997 European Control Conference, Bruxelles, Belgium, 1997

31. GRAF, B.: 'Computation of stability regions', Technical report 9801, Fachhochschule Konstanz, Fachbereich Informatik, Konstanz, Germany, 1998

32. WALTER, E., and PRONZATO, L.: 'Identification of parametric models from experimental data', (Springer, London, 1997)

33. KNUTH, D.E.: 'The art of computer programming, vol. 1' (Addison-Wesley, Reading, MA, 1973, 2nd edn.)

34. SEDGEWICK, R.: 'Algorithms' (Addison-Wesley, Reading, MA, 1988)

35. BARNETT, S.: 'Polynomials and linear control systems' (Marcel Dekker, New York, 1983)