A Matlab Tool for Regions of Attraction Estimation via Numerical Algebraic Geometry

Max Demenkov
Institute of Control Sciences
Profsoyuznaya str. 65, Moscow 117997, Russia
Email: max.demenkov@gmail.com

***Abstract*—For a locally stable polynomial dynamical system its region of attraction can be estimated by a polynomial Lyapunov function level set. We formulate this problem in terms of global minimization of a polynomial function over single polynomial constraint. We describe a simple Matlab tool, which employs numerical algebraic geometry methods for computing all local solutions of the optimization problem and therefore its global solution.**

I. Introduction

We consider local stability of the following polynomial dynamical system:

$$\dot{x}(t) = f(x(t)), \tag{1}$$

where $x \in \mathbb{R}^n, n \geq 2$ and $f(0) = 0$. We suppose that the matrix of its linearization $A = \frac{\partial f(x)}{\partial x}|_{x=0}$ is Hurwitz. This choice may seem restrictive, but in fact many mechanical systems (e.g. aircraft [1], [2], [3], walking robots [4]), can be approximated by polynomials quite well. Stability conditions for systems with impact and friction can be formulated in terms of polynomial inequalities [5].

An important characteristic of a stable attractor $\mathcal{A}$ (e.g. an equilibrium $x_0 = 0$) is the size of its *region of attraction* (ROA), which is a set of initial points $x(0)$ with the property $x(t) \to \mathcal{A}$ as $t \to \infty$. Basically, in mechanical systems this stable attractor is usually a preferable mode of operation and the size of its ROA is directly connected with the system ability to maintain the mode under external disturbances (e.g. to maintain stable walking, vertical position of Segway-like vehicles [6] or cruise aircraft flight). The size of ROA of a controlled mechanical system can be used in controller assessment (see e.g. [1], [2], [7], [8], [9] for a comparison of flight control laws).

Let us consider a positive definite continuously twice differentiable function $V(x) : \mathbb{R}^n \to \mathbb{R}$ so that its sublevel sets

$$\Omega(\gamma) = \{x : V(x) \leq \gamma\} \tag{2}$$

are bounded for $\gamma > 0$ and $V(x) = 0, ||\nabla V(x)|| = 0$ if and only if $x = 0$. For such function we expect $V(x) \to \infty$ as $x \to \infty$ and

$$\Omega(\gamma_1) \subseteq \Omega(\gamma_2) \text{ if } \gamma_1 \leq \gamma_2. \tag{3}$$

We say that $V(x)$ is a local Lyapunov function (LLF) for the equilibrium $x_0 = 0$ if it is decreasing along all trajectories $x(t)$ of the system (1) inside some nonzero invariant sublevel set:

$$\exists \gamma > 0 : \dot{V}(x(t)) < 0, \;\; x(t) \neq 0, x(t) \in \Omega(\gamma), t \geq 0, \tag{4}$$

where $\dot{V}(x) =< \nabla V(x), f(x) >$. We can estimate ROA through the sublevel sets of LLF (see e.g. [7], [10], [11]):

$$\Omega(\gamma) \subseteq \{x : \dot{V}(x) < 0\} \cup 0. \tag{5}$$

A Lyapunov function which is valid in some region of state-space can be computed e.g. from the Lyapunov equation using the linearization of our system. Another possibility is, for example, to tune parameters of a candidate LLF using system trajectories attracted to a stable equilibrium [12] or an interval analysis based branch-and-bound method [13]. For a mechanical system one can often use its kinetic and potential energy as the components of a candidate LLF. There is generally a lot of approaches for LLF determination, which we do not review here and we suppose that both dynamical system (1) and LLF have been provided.

Note that in [14] it has been proved that locally exponentially stable polynomial systems admit polynomial Lyapunov functions on compact sets. For polynomial systems and polynomial LLF one can optimize the size of the ROA estimation (5) using sum-of-squares (SOS) approach [12], [15], [16]. The approach relies on polynomial SOS representability via linear matrix inequalities (LMI) formulation and semidefinite programming. It is the most popular and well-developed method for ROA estimation with corresponding Matlab software available (see e.g. SOSTOOLS and SOSAnalysis as well as SMRSOFT). But, using the SOS approach we usually introduce some artificial functions, which (together with SOS characterization itself) impose additional constraints on the size of the computed estimation. Note that SOS polynomials represent only a fraction of all positive definite polynomials that tends to zero as we increase the state space dimension [17], although a positive polynomial can always be expressed as a ratio of two SOS polynomials. Another global optimization method that relies on LMIs is based on occupation measures and generalized moments [18]. Being an extension of Lasserre's hierarchy of LMI relaxations [19] for the solution of a global polynomial optimization problem, it is more general in nature than the SOS method. Moreover, the Lyapunov function in [18] is not fixed as in our case. We can, in fact, formulate our optimization problem directly in terms of generalized problem of moments [20] and solve it using Matlab toolbox GloptiPoly. But the size of LMIs required for realistic engineering applications can be quite large and this can be a (numerical) problem. Also, both approaches are sensitive for the choice of monomial basis in LMI formulation (simply, wrong choice can lead to poor numerical results). Therefore, despite all LMI advantages an alternative approach is worth to investigate.

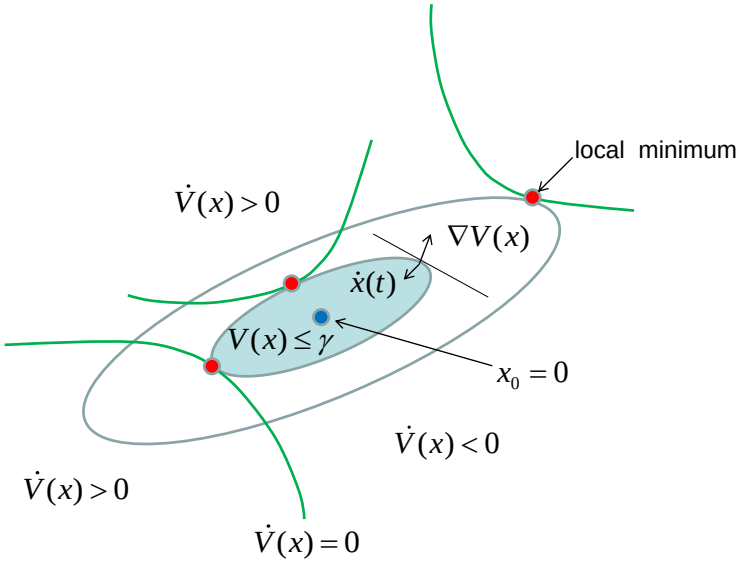

Fig. 1: ROA estimation by sublevel sets

II. Global optimization via local minima enumeration

To determine $\Omega(\gamma)$ of maximum size in terms of inclusion we compute the following open upper bound for γ (see e.g. [21], [22], [23], [24], [26]):

$$\gamma^* = \min_{x \in \mathcal{M}} V(x), \quad \mathcal{M} = \{x : \dot{V}(x) = 0, x \neq x_0\}. \tag{6}$$

In fact, one can consider (6) as a constraint in higher-level optimization of LLF parameters in order to enlarge maximal sublevel set. In [27], [11], [28] predictor-corrector methods were proposed for enlarging ROA estimation based on Clarke's generalized subgradient (predictor) and numerical solution of (6) (corrector). In the former paper modified LLF is of the quadratic form and optimization algorithm from [29] is used while in the latter papers general form of differentiable LLF is considered and (6) is modified to include additional constraints within an original framework.

Our global optimization problem can be solved using subdivision methods based on e.g. convex relaxations such as BARON [24] or interval analysis [25], [26]. Nevertheless, the solution is not always possible due to the computer memory and time demands of subdivision-based methods (solutions are approximated by boxes, the number of which is growing with each iteration).

One can find global minimum in (6) by enumerating all its local minima [21] via Karush-Kuhn-Tucker (KKT) conditions. The KKT conditions are usually not solved directly. Nevertheless, we propose to do exactly this task using new numerical methods from algebraic geometry.

In our case, the KKT approach is equal to the method of Lagrange multipliers [30] and leads to the following square system of polynomial equations:

$$F(x, \lambda) = \begin{bmatrix} \nabla V(x) - \lambda \nabla \dot{V}(x) \\ \dot{V}(x) \end{bmatrix} = 0, \tag{7}$$

where λ is the Lagrange multiplier. If we have calculated all its real isolated solutions (x, λ), we can check them for additional conditions to determine those corresponding to the global minimum (see Fig. 1). Providing that $x_0 = 0$ and the manifold $\mathcal{M}$ are separated in $\mathbb{R}^n$, the global minimum of $V(x)$ on local minima gives us the required estimation:

$$\gamma^* = \min_{i=1,N} V(x_i), \tag{8}$$

where $x_i, i = \overline{1, N}$ - x-components of the solutions of (7). Suppose we have a set I of indices i for which $V(x_i) = \gamma^*$. We need to check that at the corresponding minimum points $(x_k, \lambda_k), k \in I$ gradients of $V(x)$ and $\dot{V}(x)$ are pointing in the same direction — in this case $\lambda_k > 0$.

Finding all local minima not necessarily guarantees the correct estimation of ROA in the general case (see e.g. [11], [31]). Our method might fail by several possible reasons — e.g. we are unable to numerically compute all solutions or our global minimum is not an isolated point. An interesting case arises when we have no real solutions of (7) and our LLF candidate is actually a Lyapunov function ensuring global stability of (1). Let us consider three other special situations, which might affect correctness of our computations.

A. *The origin $x_0 = 0$ is not separated from $\mathcal{M}$.*

The minimum in (6) is 0, but one cannot detect if $x_0 = 0$ is isolated or not from considering other local minima. Nevertheless, we can exploit the fact that $V(x)$ is not LLF in this case. Let us consider local quadratic approximation of $V(x)$

using its Hessian H:

$$V(x) \approx x^T Q x^T,\ Q = \frac{1}{2}H,$$

this approximation is LLF in small neighbourhood of the origin if the following Lyapunov inequality is satisfied [10]:

$$P = A^T Q + HQ < 0.$$

This is true if all eigenvalues of P have negative real parts.

B. Isolated points $x \neq 0$ where $\dot{V}(x) = 0$.

The obvious reason for that is the existence of nonzero equilibria $x_i \neq 0, f(x_i) = 0$. We can simply add them to the collection of points in (8) (we suppose to be able to solve any polynomial square system). In fact, such points might exist only outside or on the boundary of ROA.

C. Bifurcation points of $\mathcal{M}$ where $F(x, \lambda) = 0$.

Suppose that the Jacobian of the system $F(x, \lambda)$ is not of full rank at some points of $\mathcal{M}$. According to the implicit function theorem, the system is on the brink of $bifurcation$ [32] - with infinitesimal change of its parameters, these points (or it might be connected sets) disappear or "grow" into the boundaries of regions in $\mathbb{R}^n$ characterized by a constant sign of $\dot{V}(x)$. Therefore, we can simply disregard these sets in our optimization problem. Note that an interval-based approach [25], [26] might potentially detect sets of this type.

III. Polynomial continuation

All solutions of a polynomial system of equations can be obtained with the help of Gröbner basis [23], but in practice symbolical computations associated with it are quite hard and in many cases it is impossible to get a solution at all due to limited computational resources. In this paper we use the polynomial continuation methods from *numerical algebraic geometry* [33] — a numerical approach to manipulations with polynomial systems. Apart from symbolic manipulations (such as in computing Gröbner basis) these new methods deal with numerical ways of representing solution sets of polynomial equations. To obtain all isolated complex solutions, we perform the following artificial-parameter homotopy:

$$\alpha(1-\beta)G(x,\lambda) + \beta F(x,\lambda) = 0,\ \beta \in [0,1), \qquad (9)$$

where $G(x, \lambda)$ is a start system, α is a random constant number and β is the continuation parameter. The idea here is that $G(x, \lambda)$ is much simpler and all its solutions can be easily found. Then the theory guarantees that if we trace all solutions of $G(x, \lambda)$ along some path in (x, λ, β)-space, we (almost surely) will arrive to all solutions of $F(x, \lambda)$. This is usually done using Davidenko predictor and Newton corrector [32]. There is no singularities along a solution path with a possible exception of the endpoints. In the rare case of singularity, we need to simply change α and try again. Note that for our problem we need only real solutions of (9) (in practice, one can pick up real solutions by introducing some nonzero tolerance on their complex parts).

Applications from mechanical engineering, especially robot and mechanism kinematics, greatly motivated the development of numerical algebraic geometry (some of its founders started this field while working at General Motors R&D on a number of mechanical problems [34]).

Continuation methods for the solution of polynomial systems have been studied for around 40 years. The Bézout's theorem states that the number of complex isolated solutions of a polynomial system is bounded above by the product of the total degrees of the polynomials in the system. The start system in this case can be formed from single variable equations like $x_i^k = c_i$, solution of which is straightforward. But the applicability of polynomial continuation were constrained by the enormous number of paths which we need to investigate and limited computational resources available at that time [34]. The majority of these computations were actually unnecessary. Around 20 years ago new, polyhedral methods were introduced for constructing of start systems with the help of Newton polytopes of the original system [35], [36] (Newton polytope of a polynomial is the convex hull of exponent vectors of its monomials). The corresponding theory are based on the earlier research of Soviet mathematicians Bernshtein [37], Kushnirenko [38] and Khovanskii [39]. The number of paths in this case can be reduced significantly if the polynomial system has sparse structure. At the present time, several software packages are publicly available to perform polynomial continuation, the most known are:

- Bertini [40],
- PHCpack [41],
- HOM4PS [42].

A big advantage of the polynomial continuation methods is their parallel nature [42]. Each path can be traced independently, which means we have significant reduction in solution time if we perform parallel computations using multicore processors or GPUs. Algorithms based on LMIs and Gröbner basis are essentially sequential (although LMI-based methods can benefit from parallelization of matrix operations). The scalability of numerical algebraic geometry methods is also definitely much better than subdivision-type methods (e.g. based on interval analysis). In the same time, polynomial continuation has its drawback. The wrong behaviour can arise from the so called *path-jumping* [43], when two paths are too close to each other and, while tracing one of the paths, we actually arrive on another. Then one of paths (and the corresponding solution) is lost. To avoid this, in [43], [44] robust techniques are proposed for path tracing. But, these methods are simply not available as mature public codes yet. Nevertheless, numerical algebraic geometry is a hot research field and we expect that more robust methods and corresponding software will be available in near future.

In [45], [46] these new methods were first applied to ROA estimation. Note that in [46] the precautions are taken using Bertini [40] so that local minima can be positive dimensional algebraic sets. Nevertheless, both approaches (ours and in [46]) rely on numerical solvers which are susceptible to path jumping. Therefore, for now we can theoretically guarantee only estimation from above. In the worst case, we will have an unknown gap between tight ROA estimation by a level set of $V(x)$ and our ROA estimation, which includes the former.

IV. MATLAB IMPLEMENTATION

Our Matlab tool AGRA (Algebraic Geometry for Regions of Attraction) is freely available at

https://github.com/mdemenkov/agra

For the computation of local minima in (6),(7) AGRA uses PHCpack written by Prof. Jan Verschelde and available in both Windows and Linux environments. We need to copy the executable file **phc** and its Matlab interface PHClab to the directory with AGRA files.

First we have to define our system and Lyapunov function. AGRA has no dependency on Matlab Symbolic Toolbox; instead, we use our own multivariable polynomial class **mpoly** for polynomial manipulations. Let us consider an example with reversed Van der Pol dynamics:

$$\begin{aligned}\dot{x}_1 &= -x_2\\ \dot{x}_2 &= x_1 + x_1^2 x_2 - x_2\end{aligned}$$

This system can be represented as an **mpoly** array **f** (each element corresponds to one equation):

```
x=mpolyfun.singles(2);
f(1)=-x(2);
f(2)=x(1)+x(1)^2*x(2)-x(2);
```

The LLF which we took from [13] is as follows:

$$V(x) = x_1^2 - 0.3449172 x_1 x_2 + 0.8589766 x_2^2$$

and its corresponding Matlab representation is

```
a=1;b=-0.3449172;c=0.8589766;
V=mpoly([a;b;c],[2 0;1 1;0 2]);
```

Note that we've used different (but interchangeable) representation for polynomials here - we can either add individual monomials or define whole polynomial via its vector of coefficients and a matrix with rows corresponding to the vector of powers for each term.

To compute ROA estimation, we have to create an instance of another class **agrasys**, attach to it our LLF and calculate γ^* as follows (the last function invokes PHCpack):

```
sys=agrasys(f,V);
gamma_star=sys.max_level()
```

The result is approximately 1.6, in Fig. 2 one can see the corresponding level curve of the Lyapunov function (magenta) and the manifold $\mathcal{M}$ (red).

Let us now consider our Matlab functions in more detail.

A. Working with polynomials

Using overloaded operators, it is possible to add, subtract and multiply polynomials in the usual way. If we want to construct a monomial

$$p(x) = c x_1^{p1} x_2^{p2} ... x_n^{pn},$$

with the help of **mpoly**, we can type the following code:

```
p=mpoly(c,[p1 p2 ... pn])
```

The zero power corresponds to the absence of a variable in the monomial. Negative powers are not allowed. It is important to note that the number of variables in each monomial of a polynomial should be equal. The following example trying to build $ax_1 - bx_1x_2^2$ produces an error:

```
mpoly(a,1)-b*mpoly(1,[1 2])
```

while

```
mpoly(a,[1 0])-b*mpoly(1,[1 2])
```

is the correct code. In the last example, one can replace its first term with

```
extend(mpoly(a,1),1)
```

— this function adds dimensionality to the given polynomial (the number of additional variables is defined by its second argument). The value of polynomial p at the point x is given by **fval**(p,x) and its **derivative**(p,i) w.r.t. i-th variable is provided.

Another package class **mpolyfun** contains some auxiliary functions for **mpoly** polynomials. For example, **get_A(f)** returns matrix A of linearized system dynamics, while **get_quadratic(Q)** translates positive definite matrix Q into **mpoly** polynomial x^TQx. The complete list of available functions can be found in AGRA help files.

B. Visualization

We have provided a simple GUI version of our tool in the form of the class **agragui**, for its construction we need an instance of **agrasys**:

```
mygui=agragui(sys);  mygui.window();
```

A number of functions for producing graphical output in a standalone window are available in this class as well. If the system (1) has more than two dimensions, we need to set the desired visualization plane for cross-sections of $\dot{V}(x) = 0$ and $V(x) = \gamma^*$. We can also compute ROA numerically by the Runge-Kutta integration method **ode45** to compare with the Lyapunov estimation. Note that comparison between $\dot{V}(x)$ and $V(x)$ makes sense only for 2D/3D examples.

The example in Fig. 3 corresponds to the following system and LLF, again taken from [13]:

$$\begin{aligned}\dot{x}_1 &= -x_1 + x_2\\ \dot{x}_2 &= 0.1x_1 - 2x_2 - x_1^2 - 0.1x_1^3\\ V(x) &= x_1^2 + 1.6513 x_2^2\end{aligned}$$

ACKNOWLEDGMENT

The author would like to thank Dr. V. A. Kamenetskiy (Institute of Control Sciences, Moscow) for his valuable comments on the Lyapunov stability conditions in this paper as well as his colleague Dr. A. V. Gorbunov for numerous discussions on the subject. Prof. M. G. Goman takes credit for the motivation for AGRA development during the author's visit at De Montfort University, UK.

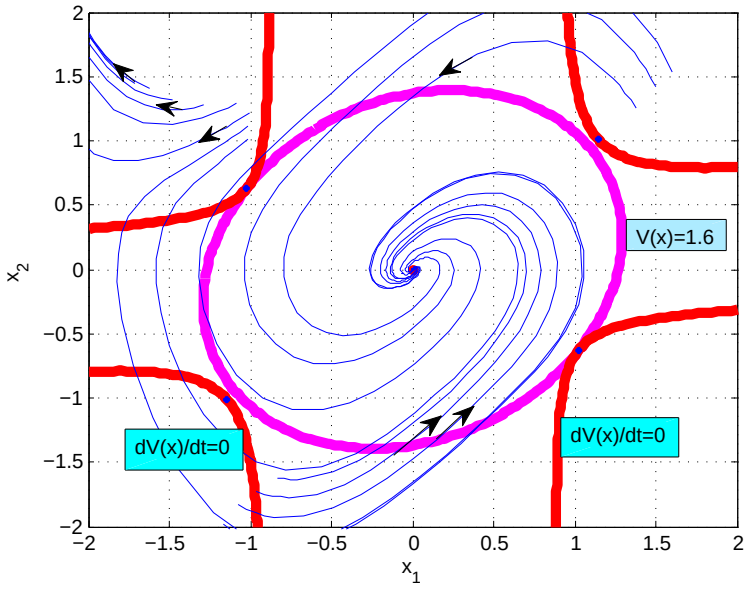

Fig. 2: Van der Pol example

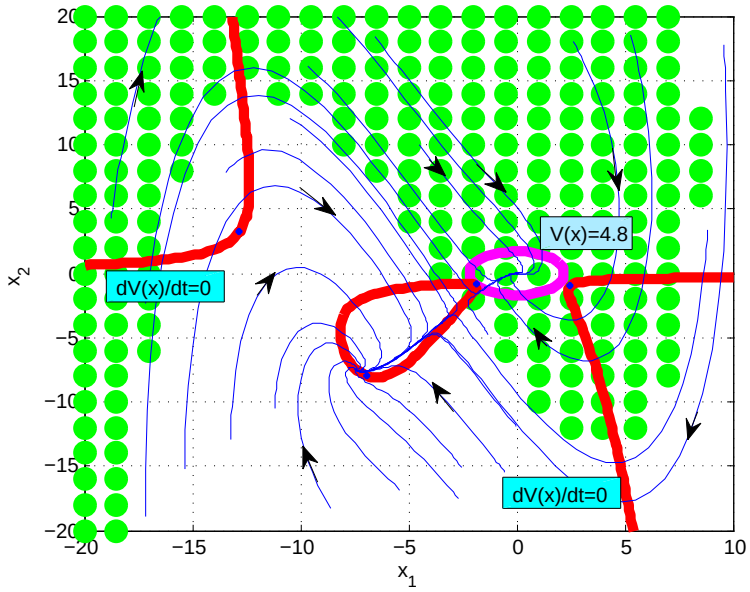

Fig. 3: System with multiple equilibria (green circles cover ROA)

REFERENCES

[1] A. Chakraborty, P. Seiler, G. J. Balas, Nonlinear region of attraction analysis for flight control verification and validation. *Control Engineering Practice*, Vol. 19, No. 4, pp. 335-345, 2011.

[2] A. Chakraborty, P. Seiler, G. Balas, Susceptibility of F/A-18 flight controllers to the falling-leaf mode: nonlinear analysis. *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 1, pp. 73-85, 2011.

[3] E. Glassman, A. L. Desbiens, M. Tobenkin, M. Cutkosky, R. Tedrake, Region of attraction estimation for a perching aircraft: a Lyapunov method exploiting barrier certificates. *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 2235-2242, Saint Paul, Minnesota, 2012.

[4] I. Manchester, M. Tobenkin, M. Levashov, R. Tedrake, Regions of attraction for hybrid limit cycles of walking robots. *Proc. of the 18th World Congress of the International Federation of Automatic Control*, pp. 5801-5806, Milano, 2011.

[5] M. Posa, M. Tobenkin, R. Tedrake, Lyapunov analysis of rigid body systems with impacts and friction via sums-of-squares. *Proc. of the 16th ACM Conference on Hybrid Systems: Computation and Control*, pp. 63-72, Philadelphia, 2013.

[6] Yu. G. Martynenko, A. M. Formal'skii, The theory of the control of a monocycle. *Journal of Applied Mathematics and Mechanics*, Vol. 69, No. 4, pp. 516-528, 2005.

[7] P. Seiler, G. J. Balas, A. K. Packard, Assessment of aircraft flight controllers using nonlinear robustness analysis techniques. In: *Optimization Based Clearance of Flight Control Laws*, Lecture Notes in Control and Information Sciences, Vol. 416, pp. 369-397, Springer-Verlag, Berlin, 2012.

[8] M. Goman, Y. Patel, M. Sidoryuk, Regions of attraction for robustness assessment. *Proc. of the AIAA GNC Conference*, Austin, 2003.

[9] M. Demenkov, M. Goman, Suppressing aeroelastic vibrations via stability region maximization and numerical continuation techniques. *Proc. of the UKACC Control Conference*, Manchester, UK, 2008.

[10] H. K. Khalil, *Nonlinear systems*. 3rd edition. Prentice Hall, Upper Saddle River, New Jersey, 2002.

[11] V. A. Kamenetskii, Construction of regions of attraction by the Lyapunov function method. *Automation and Remote Control*, Vol. 55, No. 6, pp. 778-791, 1994.

[12] U. Topcu, A. Packard, P. Seiler, Local stability analysis using simulations and sum-of- squares programming. *Automatica*, Vol. 44, pp. 2669-2675, 2008.

[13] S. Ratschan, Z. She, Providing a basin of attraction to a target region of polynomial systems by computation of Lyapunov-like functions. *SIAM Journal of Control and Optimization*, Vol. 48, No. 7, pp. 4377-4394, 2010.

[14] M. M. Peet, Exponentially stable nonlinear systems have polynomial Lyapunov functions on bounded regions. *IEEE Transactions on Automatic Contro*l, Vol. 54, No. 5, pp. 979-987, 2009.

[15] G. Chesi, *Domain of attraction. Analysis and control via SOS programming.* Springer-Verlag, London, 2011.

[16] U. Topcu, A. Packard, P. Seiler and G. Balas, Ask the experts: help on SOS. *IEEE Control Systems Magazine*, Vol. 30, No.4, pp. 18-23, 2010.

[17] G. Blekherman, There are significantly more nonnegative polynomials than sums of squares, *Israel Journal of Mathematics*,Vol. 153, No. 1, pp. 355-380, 2006.

[18] D. Henrion, M. Korda, Convex computation of the region of attraction of polynomial control systems. *IEEE Transactions on Automatic Control*, Vol. 59, No. 2, pp. 297-312, 2014.

[19] J. B. Lasserre, Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, Vol. 11, No. 3, pp. 796-817.

[20] J. B. Lasserre, *Moments, positive polynomials and their applications*. Imperial College Press, London, 2009.

[21] J. J. Rodden, *Applications of Lyapunov stability theory*. Doctoral dissertation, Division of Engineering Mechanics, Stanford University, Stanford, 1964.

[22] D. Shields, C. Storey, The behaviour of optimal Lyapunov functions. *International Journal of Control*, Vol. 21, No. 4, pp. 561-573, 1975.

[23] K. Forsman, Construction of Lyapunov functions using Gröbner basis. *Proc. of the 30th CDC*, pp. 798-799, Brighton, England, 1991.

[24] L. G. Matallana, A. M. Blanco, J. A. Bandoni, Estimation of domains of attraction: a global optimization approach. *Mathematical and Computer Modelling*, Vol. 52, No. 34, pp. 574-585, 2010.

[25] H. Burchardt, S. Ratschan, Estimating the region of attraction of ordinary differential equations by quantified constraint solving. *Proc. of the CONTROL'07*, WSEAS Press, pp. 241-246, Arcachon, 2007.

[26] S. Warthenpfuhl, B. Tibken, S. Mayer, An interval arithmetic approach for the estimation of the domain of attraction. *Proc. of IEEE Multi-Conference on Systems and Control*, pp. 1999-2004, Yokohama, Japan, 2010.

[27] J. Hauser, M. C. Lai, Estimating quadratic stability domains by nonsmooth optimization. *Proc. of the ACC*, pp. 571-576, Chicago, 1992.

[28] V. A. Kamenetskiy, A method for construction of stability regions by Lyapunov functions. *Systems & Control Letters*, Vol. 26, No. 2, pp. 147-151, 1995.

[29] E. Polak. On the mathematical foundations of nondifferentiable optimization in engineering design. *SIAM Review*, Vol. 29, pp. 21-89, 1987.

[30] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*. Academic Press, New York, 1982.

[31] A. V. Gorbunov, On critical level evaluation for basin of attraction estimation by Lyapunov functions technique, *Proc. of the XI International Conference "Stability and oscillations of nonlinear control systems"*, pp. 96-98, Moscow, 2010.

[32] E. L. Allgower, K. Georg, *Numerical continuation methods: an introduction*. Springer-Verlag, New York, 1990.

[33] A. J. Sommese, J. Verschelde, C. W. Wampler, Introduction to numerical algebraic geometry. *Solving polynomial equations: foundations, algorithms, and applications. Algorithms and Computation in Mathematics*, Vol. 14, pp. 301-337, Springer-Verlag, Berlin, 2005.

[34] A. Morgan, *Solving polynomial systems using continuation for engineering and scientific problems*. Prentice-Hall, Upper Saddle River, New Jersey, 1987.

[35] J. Verschelde, P. Verlinden and R. Cools, Homotopies exploiting Newton polytopes for solving sparse polynomial systems. *SIAM Journal of Numerical Analysis*, Vol. 31, pp. 915-930, 1994.

[36] B. Huber, B. Sturmfels, A polyhedral method for solving sparse polynomial systems. *Mathematics of Computation*, Vol. 64, pp. 1541-1555, 1995.

[37] D. N. Bernshtein, The number of roots of a system of equations. *Functional Analysis and Applications*, Vol. 9, No. 3, pp.183-185, 1975.

[38] A. G. Kushnirenko, Newton polytopes and the Bezout theorem. *Functional Analysis and Applications*, Vol. 10, No. 3, pp. 233-235, 1976.

[39] A. G. Khovanskii, Newton polyhedra and the genus of complete intersections. *Functional Analysis and Applications*, Vol. 12, No. 1, pp. 38-46, 1978.

[40] D. J. Bates, J. D. Hauenstein, A. J. Sommese, C. W. Wampler, *Numerically solving polynomial systems with Bertini*. SIAM, Philadelphia, 2013.

[41] J. Verschelde, Algorithm 795. PHCpack: a general-purpose solver for polynomial systems by homotopy continuation. *ACM Transaction on Mathematical Software*, Vol. 25, No. 2, pp. 251-276, 1999.

[42] T. Chen, T. L. Lee, T. Y. Li, Hom4PS-3: a parallel numerical solver for systems of polynomial equations based on polyhedral homotopy continuation methods. In: *Mathematical Software - ICMS 2014*, LNCS Vol. 8592, pp. 183-190, Springer-Verlag, Berlin, 2014.

[43] C. Beltran, A. Leykin, Robust certified numerical homotopy tracking. *Foundations of Computational Mathematics*, Vol. 13, No. 2, pp. 253-295, 2013.

[44] B. Martin, A. Goldsztejn, L. Granvilliers, C. Jermann, Certified parallelotope continuation for one-manifolds. *SIAM Journal of Numerical Analysis*, Vol. 51, No. 6, pp. 3373-3401, 2013.

[45] M. Demenkov, Estimating region of attraction for polynomial vector fields by homotopy methods. *ACM Communications in Computer Algebra*, Vol. 46, No. 3, pp. 84-85, 2012.

[46] F. Permenter, C. Wampler, R. Tedrake, A numerical algebraic geometry approach to regional stability analysis of polynomial systems. *Proc. of the ACC*, pp. 2127-2132, Washington DC, 2013.