# DSOS and SDSOS Optimization: LP and SOCP-Based Alternatives to Sum of Squares Optimization

Amir Ali Ahmadi and Anirudha Majumdar

*Abstract*— **Sum of squares (SOS) optimization has been a powerful and influential addition to the theory of optimization in the past decade. Its reliance on relatively large-scale semidefinite programming, however, has seriously challenged its ability to scale in many practical applications. In this paper, we introduce DSOS and SDSOS optimization[1] as more tractable alternatives to sum of squares optimization that rely instead on linear programming and second order cone programming. These are optimization problems over certain subsets of sum of squares polynomials and positive semidefinite matrices and can be of potential interest in general applications of semidefinite programming where scalability is a limitation.**

*Note: This is an invited paper for the session on "Optimization in the Information Sciences" of the 48th Annual Conference on Information Sciences and Systems (CISS 2014). It is meant to serve as an extended abstract for a longer upcoming paper [1] by the authors on the same topic. Most details are omitted.*

## I. Introduction

A fundamental problem in applied and computational mathematics is that of optimizing over the set of *nonnegative polynomials*. In such an optimization problem, one would like to impose constraints on the coefficients of some unknown (multivariate) polynomials in a way that they become nonnegative, either globally in $\mathbb{R}^n$, or on certain *basic semialgebraic sets*, i.e., subsets of $\mathbb{R}^n$ defined by a finite number of polynomial inequalities. We give two examples of fundamental domains where such optimization problems arise and refer the reader to [1] for several others.

**Polynomial optimization:** The polynomial optimization problem (POP) in its general form is represented as

$$\begin{array}{ll} \text{minimize} & p(x) \\ \text{subject to} & x \in K := \{x \in \mathbb{R}^n \mid g_i(x) \geq 0, h_i(x) = 0\}, \end{array} \tag{1}$$

where $p$, $g_i$, and $h_i$ are multivariate polynomials. The special case of problem (1) where the polynomials $p, g_i, h_i$ all have degree one is of course *linear programming*, which can be solved very efficiently. For higher degrees, POP contains as special case many important problems in operations research; e.g., the optimal power flow problem in power engineering [2], computation of Nash equilibria in game theory [3], and a host of problems in combinatorial optimization. By a straightforward reformulation of problem (1), we observe that if we could optimize over the set of polynomials that are *nonnegative on a basic semialgebraic set*, then we could solve the POP problem to global optimality. To see this, note that the optimal value of problem (1) is equal to the optimal value of the following problem:

$$\begin{array}{ll} \text{maximize} & \gamma \\ \text{subject to} & p(x) - \gamma \geq 0, \ \forall x \in K. \end{array} \tag{2}$$

Here, we are trying to find the largest constant $\gamma$ such that the polynomial $p(x) - \gamma$ is nonnegative on the set $K$; i.e., the largest lower bound on problem (1).

**Control systems and robotics:** Numerous fundamental problems in nonlinear dynamics and control, such as stability, robustness, collision avoidance, controller design, etc., can be turned into problems about finding special functions—the so-called *Lyapunov functions* (see, e.g., [4])—that satisfy certain sign conditions. For example, given a differential equation $\dot{x} = f(x)$, with $f : \mathbb{R}^n \to \mathbb{R}^n$, and origin as an equilibrium point (i.e., satisfying $f(0) = 0$), consider the "region of attraction problem": For what set of initial conditions in $\mathbb{R}^n$ do trajectories flow to the origin? Lyapunov's stability theorem (see, e.g., [4, Chap. 4]) tells us that if we can find a (Lyapunov) function $V : \mathbb{R}^n \to \mathbb{R}$, which together with its gradient $\nabla V$ satisfies

$$\begin{array}{rl} V(x) & > 0, \forall x \neq 0, \\ -\langle \nabla V(x), f(x) \rangle & > 0, \forall x \in \{x \mid V(x) \leq \beta, x \neq 0\}, \end{array} \tag{3}$$

then the sublevel set $\{x \mid V(x) \leq \beta\}$ is part of the region of attraction. If $f$ is a polynomial function (an immensely important case in applications [5]), and if we parameterize $V$ as a polynomial function, then the search for the coefficients of $V$ satisfying the conditions in (3) is an optimization problem over the set of nonnegative (or in this case positive) polynomials. Designing stable equilibrium points with large regions of attraction is a fundamental problem in control engineering and robotics [6].

Closely related to nonnegative polynomials are polynomials that are sums of squares. We say that a polynomial $p$ is a *sum of squares* (sos) if it can be written as $p = \sum_i q_i^2$ for some (finite number of) polynomials $q_i$. The study of the relationship between nonnegative and sum of squares

AAA is a Goldstine Fellow at the IBM Watson Research Center and a member of the Mathematical Programming Group at IBM's Department of Business Analytics and Mathematical Sciences.
aaa@us.ibm.com, `http://aaa.lids.mit.edu/`
AM is with the Department of Electrical Engineering and Computer Science, CSAIL, MIT.
anirudha@mit.edu, `http://www.mit.edu/~anirudha`

[1]We use and recommend the pronunciation *"d-sauce"* and *"s-d-sauce"*.

polynomials is a classical problem in real algebraic geometry. In the general, while sum of squares polynomials are clearly always (globally) nonnegative, the exist nonnegative polynomials that cannot be written as a sum of squares. This fact was first shown with non-constructive arguments in a celebrated paper of Hilbert [7], and later with concrete examples starting with the work of Motzkin [8].

The classical questions around nonnegative and sum of squares polynomials have been revisited quite extensively in the past 10-15 years in different communities among applied and computational mathematicians. The reason for this renewed interest is twofold: (i) the discovery that many problems of modern practical interest can be cast as optimization problems over nonnegative polynomials, and (ii) the observation that while optimizing over nonnegative polynomials is generally NP-hard, optimization over the set of sum of squares polynomials can be done via *semidefinite programming* (SDP). The latter development, originally explored in the pioneering works of Shor [9], Nesterov [10], Parrilo [11], [12], and Lasserre [13], has led to the so-called area of *sum of squares optimization*—a computational framework, with semidefinite programming as its underlying engine, that can tackle many fundamental problems of real algebra and polynomial optimization.

## A. Motivation and contributions

The dependence of sum of squares approaches on semidefinite programming is a strength and at the same time a weakness, depending on point of view. From a computational complexity perspective, semidefinite programs can be solved with arbitrary accuracy in polynomial time using interior point methods (see [14] for a comprehensive survey). As a result, sum of squares techniques offer polynomial time algorithms that approximate a very broad class of NP-hard problems of interest. From a more practical viewpoint, however, SDPs are among the most expensive convex relaxations and the speed and reliability with which they can be solved currently lag by a wide margin those of some more restricted classes of convex programs, such as linear programs or second order cone programs. This, combined with the fact that sos problems generate large-scale semidefinite programs to begin with (see [1] for details), has made scalability the single most outstanding challenge for sum of squares optimization.

In [1], we take the latter, more practical viewpoint on this issue and offer alternatives to sum of squares optimization that while more conservative in general, are significantly more scalable. Our hope is that by doing so, we expand the use and applicability of algebraic techniques in optimization to new areas and share its appeal with a broader audience. We call our new computational frameworks, which rely on linear and second order cone programming, *DSOS and SDSOS optimization* (see Section II for precise definitions). While these tools are primarily designed for sum of squares optimization, they are also applicable to general applications of semidefinite programming where tradeoffs between scalability and performance may be desirable.

A particularly nice feature of DSOS and SDSOS optimization is that they enjoy many of the same theoretical guarantees that underly SOS optimization. In [1], we show how several basic results from SOS optimization, which often rely on theorems of real algebraic geometry, carry over in a straightforward manner to our new optimization problems. We also show in [1] with numerical experiments from diverse application areas—polynomial optimization, statistics and machine learning, derivative pricing, and control theory—that we can handle problems at scales that are currently far beyond reach for sum of squares approaches. The remainder of this note gives a brief presentation of a subset of the results in [1] with proofs and details omitted.

## II. DSOS and SDSOS Optimization

Let $PSD_{n,d}$ and $SOS_{n,d}$ respectively denote the cone of nonnegative and sum of squares polynomials in $n$ variables and degree $d$, with the obvious inclusion relation $SOS_{n,d} \subseteq PSD_{n,d}$. The basic idea is to approximate the cone $SOS_{n,d}$ from the inside with new cones that are more tractable for optimization. Towards this goal, one may think of several natural sufficient conditions for a polynomial to be a sum of squares. For example, consider the following cones:

- The cone of polynomials that are sums of 4-th powers of polynomials: $\{p \mid p = \sum q_i^4\}$,
- The cone of polynomials that are a sum of three squares of polynomials: $\{p \mid p = q_1^2 + q_2^2 + q_3^2\}$.

Even though both of these cones clearly reside inside the sos cone, they are not any easier to optimize over. In fact, they are harder! Testing membership to either of these two cones is NP-hard [15]. This shows that we have to take some care in deciding what subset of $SOS_{n,d}$ we exactly choose to work with: on one hand, it should be computationally simpler for optimization; on the other hand, it has to comprise a "big enough" subset.

## A. The cone of r-dsos and r-sdsos polynomials

We now describe cones inside $P_{n,d}$ that are naturally motivated and that lend themselves to linear and second order cone programming. There are also useful generalizations of these cones that result in "small" semidefinite programs. These generalizations are presented in [1] and are omitted from here.

*Definition 1 (Ahmadi, Majumdar,'13):*

- A polynomial $p$ is *diagonally-dominant-sum-of-squares* (dsos) if it can be written as
$$p = \sum_i \alpha_i m_i^2 + \sum_{i,j} \beta_{ij}(m_i \pm m_j)^2,$$
for some monomials $m_i, m_j$ and some nonnegative constants $\alpha_i, \beta_{i,j}$.

- A polynomial $p$ is *scaled-diagonally-dominant-sum-of-squares* (sdsos) if it can be written as
$$p = \sum_i \alpha_i m_i^2 + \sum_{i,j}(\beta_i m_i \pm \gamma_j m_j)^2,$$

for some monomials $m_i, m_j$ and some constants $\alpha_i \geq 0, \beta_i, \gamma_i$.

- For a positive integer $r$, a polynomial $p$ is *r-diagonally-dominant-sum-of-squares* (r-dsos) if $p \cdot \left(\sum_i x_i^2\right)^r$ is dsos.
- For a positive integer $r$, a polynomial $p$ is *r-scaled-diagonally-dominant-sum-of-squares* (r-sdsos) if $p \cdot \left(\sum_i x_i^2\right)^r$ is sdsos.

We denote the set of polynomials in $n$ variables and degree $d$ that are dsos, sdos, r-dsos, and r-sdsos by $DSOS_{n,d}, SDSOS_{n,d}, rDSOS_{n,d}, rSDSOS_{n,d}$, respectively.

The following inclusion relations are straightforward:

$$DSOS_{n,d} \subseteq SDSOS_{n,d} \subseteq SOS_{n,d} \subseteq POS_{n,d},$$

$$rDSOS_{n,d} \subseteq rSDSOS_{n,d} \subseteq POS_{n,d}, \forall r.$$

Our terminology in Definition 1 comes from the following concepts in linear algebra.

*Definition 2:* A symmetric matrix $A$ is *diagonally dominant* (dd) if $a_{ii} \geq \sum_{j \neq i} |a_{ij}|$ for all $i$. A symmetric matrix $A$ is *scaled diagonally dominant* (sdd) if there exists an element-wise positive vector $y$ such that:

$$a_{ii} y_i \geq \sum_{j \neq i} |a_{ij}| y_j, \forall i.$$

Equivalently, $A$ is sdd if there exists a positive diagonal matrix $D$ such that $AD$ (or equivalently, $DAD$) is dd. We denote the set of $n \times n$ dd and sdd matrices with $DD_n$ and $SDD_n$ respectively.

*Theorem 2.1 (Ahmadi, Majumdar,'13):*

- A polynomial $p$ of degree $2d$ is dsos if and only if it admits a representation as $p(x) = z^T(x)Qz(x)$, where $z(x)$ is the standard monomial vector of degree $d$, and $Q$ is a dd matrix.
- A polynomial $p$ of degree $2d$ is sdsos if and only if it admits a representation as $p(x) = z^T(x)Qz(x)$, where $z(x)$ is the standard monomial vector of degree $d$, and $Q$ is a sdd matrix.

*Theorem 2.2 (Ahmadi, Majumdar,'13):* For any nonnegative integer $r$, the set $rDSOS_{n,d}$ is polyhedral and the set $rSDSOS_{n,d}$ has a second order cone representation. For any fixed $r$ and $d$, optimization over $rDSOS_{n,d}$ (resp. $rSDSOS_{n,d}$) can be done with linear programming (resp. second order cone programming), of size polynomial in $n$.

## B. Asymptotic guarantees and comparison to sos techniques

The purpose of the parameter $r$ is to have a knob for trading off speed with accuracy of approximation. By increasing $r$, we obtain increasingly accurate inner approximations to the set of nonnegative polynomials. The following two examples show that the linear programs obtained from even small $r$ can outperform the semidefinite programs resulting from sum of squares.

*Example 2.1:* Consider the Motzkin polynomial [8], $M(x) = x_1^4x_2^2 + x_1^2x_2^4 - 3x_1^2x_2^2x_3^2 + x_3^6$, which is famously known to be nonnegative but *not* a sum of squares. We can give an LP-based nonnegativity certificate of this polynomial by showing that $M \in 2DSOS$. Hence, $2DSOS \nsubseteq SOS$.

*Example 2.2:* Consider the polynomial, $p(x) = x_1^4x_2^2 + x_2^4x_3^2 + x_3^4x_1^2 - 3x_1^2x_2^2x_3^2$. Once again this polynomial is nonnegative but *not* a sum of squares [16]. However, there is an LP-based nonnegativity certificate since $M \in 1DSOS$. Hence, $1DSOS \nsubseteq SOS$.

The following two theorems provide asymptotic guarantees on r-dsos (and hence r-sdsos) hierarchies. Their proofs rely on powerful Positivstellensatz results from real algebraic geometry.

*Theorem 2.3 (Ahmadi, Majumdar,'13):* Let $p$ be an even form, with $p(x) > 0$ for all $x \neq 0$, then there exists an integer $r$ such that $p \in rDSOS$.

The theorem states that at least for even forms (forms where individual variables are only raised to even degrees), a certificate of nonnegativity can *always* be found by one of the LPs in our hierarchy. Even forms already form a very interesting class of polynomials since they include, e.g., all polynomials coming from *copositive programming* [12], and many important $0/1$ integer programs and combinatorial optimization problems. For example, when the hierarchy is applied to the *maximum stable set problem*, by using a result from [17] we can show that $\alpha^2(G)$ levels of the $rDSOS$ hierarchy are enough to give the exact value $\alpha(G)$ of the maximum stable set of any graph $G$.

The next theorem removes the assumption of evenness from the previous theorem by using a general multiplier (as opposed to $\sum x_i^2$).

*Theorem 2.4 (Ahmadi, Majumdar,'13):* For any positive definite form $p$, there exists a form $q$ such that $q$ and $pq$ are both dsos.

Note that given $p$, the search for such a $q$ (of a given degree) is a linear program. Moreover, a feasible solution to this linear program certifies nonnegativity of $p$. This theorem can be used to prove the following general result, which we state here informally.

*Theorem 2.5 (Ahmadi, Majumdar,'13):* Consider the general polynomial optimization problem (POP) in 1. There is a hierarchy of linear programs based on optimization over dsos polynomials that can solve POP to arbitrary accuracy.

This theorem is similar to the Parrilo and Lasserre hierarchies for polynomial optimization that are instead based on a search over sum of squares polynomials via semidefinite programming [11], [13].

## III. Numerical examples

In this section, we focus on two problems from statistics and control theory to show the magnitude of improvements in scalability that we can achieve by using dsos and sdsos conditions over sos. See [1] for several other examples. The machine used for our numerical examples is a PC with 4 processors, a clock speed of 3.4 GHz, and 16GB of RAM.

## A. Convex regression

We consider the problem of fitting a function to data subject to a constraint on convexity of the function. This is an important problem in statistics with applications such as value function estimation in reinforcement learning, utility function estimation in economics, among many others. Formally, we are given pairs of data points $(x_i, y_i)$ with $x_i \in \mathbb{R}^N$, $y_i \in \mathbb{R}$ and the task is to find a *convex* function $f$ from a family $\mathcal{F}$ that minimizes an appropriate notion of fitting error (e.g., $L_\infty$ error):

$$\begin{array}{ll} \min\limits_{f \in \mathcal{F}} & \max\limits_{i} |f(x_i) - y_i| \\ \text{s.t.} & f(x) \text{ is convex.} \end{array}$$

The convexity constraint here can be imposed by requiring that the function $y^T H(x) y$ associated with the Hessian $H(x)$ of $f(x)$ be nonnegative for all $x, y$. Restricting ourselves to polynomial functions $f$ of bounded degree, we obtain the following optimization problem:

$$\begin{array}{ll} \min\limits_{f \in \mathbb{R}^{d}[x]} & \max\limits_{i} |f(x_i) - y_i| \\ \text{s.t.} & y^T H(x) y \geq 0, \end{array}$$

where $H(x)$ is again the Hessian of $f(x)$. Unfortunately, optimizing over the set of convex polynomials or even testing convexity of a given polynomial function is NP-hard in the strong sense [18]. As an algebraic relaxation, we can replace the latter nonnegativity constraint with a dsos/sdsos/sos constraint and obtain linear, second order cone, or semidefinite programs. For our numerical experiment, we generated 300 random vectors $x_i$ in $\mathbb{R}^{20}$ drawn i.i.d. from the standard normal distribution. The function values $y_i$ are computed as follows:

$$y_i = \exp(\|x_i\|_2) + w_i,$$

where $w_i$ is chosen i.i.d. from the standard normal distribution. Note here that we are trying to approximate a non-polynomial convex function with a convex polynomial function (in the presence of noise), and that the dsos/sdsos/sos constraints are on a polynomial with 40 variables. Tables III-A and III-A present the fitting errors and running times for the DSOS/SDSOS/SOS programs resulting from restricting the class of functions $\mathcal{F}$ to polynomials of degree $d = 2$ and $d = 4$. As the results illustrate, we are able to obtain significantly smaller errors with polynomials of degree 4 using DSOS and SDSOS (compared to SOS with $d = 2$), while the SOS program for $d = 4$ simply does not get past the first iteration of the interior point solver even after several hours have elapsed.

## B. Region of attraction computation

Most (but not all) applications of SOS programming in control theory rely on checking some type of *Lyapunov inequalities* for trajectories of nonlinear systems. Here, we consider the representative problem of estimating a region of attraction, which has been studied in the literature extensively [19], [20].

| | DSOS | SDSOS | SOS |
|---|---|---|---|
| $d = 2$ | 35.11 | 33.92 | 21.28 |
| $d = 4$ | 14.86 | 12.94 | NA |

TABLE I

COMPARISON OF FITTING ERRORS FOR OUR CONVEX REGRESSION PROBLEM.

| | DSOS | SDSOS | SOS |
|---|---|---|---|
| $d = 2$ | <1 s | <1 s | <1 s |
| $d = 4$ | 145.85 s | 240.18 s | $\infty$ |

TABLE II

COMPARISON OF RUNNING TIMES ERRORS FOR OUR CONVEX REGRESSION PROBLEM.

As discussed in Section I, one can compute (inner approximations of) the region of attraction (ROA) of a dynamical system $\dot{x} = f(x)$ by finding a function $V : \mathbb{R}^n \to \mathbb{R}$ that satisfies the following conditions:

$$\begin{array}{rl} V(x) & > 0, \forall x \neq 0, \\ -\langle \nabla V(x), f(x) \rangle & > 0, \forall x \in \{x | \ V(x) \leq \beta, x \neq 0\}. \end{array} \tag{4}$$

This guarantees that the sublevel set $\{x | \ V(x) \leq \beta\}$ is a subset of the ROA of the system; i.e., initial conditions that begin in this sublevel set converge to the origin. However, the task finding a polynomial Lyapunov function satisfying the inequalities in (4) is intractable. For example, even the problem of testing if a cubic vector field admits a local quadratic Lyapunov function is strongly NP-hard [21].

Nevertheless, there are standard ways of replacing the inequalities in (4) with (more conservative) sum of squares conditions to attack the problem with semidefinite programming; see, e.g., [12], [19], [20]. By replacing the sos constraints with dsos/sdsos constraints, one can compute inner approximations to the ROA more efficiently. While the approximations will be more conservative in general, the ability to tradeoff conservatism with computation time afforded to us by the approach presented in this paper is an important one.

Figures 2(a) and 2(b) present preliminary evidence that the sacrifice in terms of conservatism can be relatively minor. We consider an underactuated 5-link pendulum (depicted in Figure 1), which has four actuators (one at every joint except the base joint) and a $n = 10$ dimensional state space. The pendulum is balanced about the upright position using a Linear Quadratic Regulator (LQR) controller. The figure compares projections of the computed ROAs onto two $2-$dimensional subspaces of the state space. As the plot suggests, the ROA computed using SDSOS programming is only slightly more conservative than the SOS ROA. In fact, comparing the volumes of the two ROAs raised to the reciprocal of the ambient dimension, we find:

$$\frac{Vol^{1/n}_{\text{ROA-sdsos}}}{Vol^{1/n}_{\text{ROA-sos}}} = 0.79. \tag{5}$$

The ROA computed using DSOS is much smaller (but still potentially useful in practice). The running time of the solvers for DSOS,SDSOS, and SOS (with the standard solver SeDuMi [22]) are respectively 37 seconds, 40 seconds, and 1.6 hours on a standard PC. The speed-up is about a factor of 150.

## References

[1] A. A. Ahmadi and A. Majumdar. DSOS and SDSOS: more tractable alternatives to sum of squares and semidefinite optimization. In preparation, 2014.

[2] M Huneault and FD Galiana. A survey of the optimal power flow literature. *IEEE Transactions on Power Systems*, 6(2):762–770, 1991.

[3] Bernd Sturmfels. *Solving systems of polynomial equations*. Number 97. American Mathematical Soc., 2002.

[4] H. Khalil. *Nonlinear systems*. Prentice Hall, 2002. Third edition.

[5] A. A. Ahmadi and P. A. Parrilo. Stability of polynomial differential equations: complexity and converse Lyapunov questions. arXiv:1308.6833, 2013.

[6] A. Majumdar, A. A. Ahmadi, and R. Tedrake. Control design along trajectories with sums of squares programming. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013.

[7] D. Hilbert. Über die Darstellung Definiter Formen als Summe von Formenquadraten. *Math. Ann.*, 32, 1888.

[8] T. S. Motzkin. The arithmetic-geometric inequality. In *Inequalities (Proc. Sympos. Wright-Patterson Air Force Base, Ohio, 1965)*, pages 205–224. Academic Press, New York, 1967.

[9] N. Z. Shor. Class of global minimum bounds of polynomial functions. *Cybernetics*, 23(6):731–734, 1987. (Russian orig.: Kibernetika, No. 6, (1987), 9–11).

[10] Yu. Nesterov. Squared functional systems and optimization problems. In *High performance optimization*, volume 33 of *Appl. Optim.*, pages 405–440. Kluwer Acad. Publ., Dordrecht, 2000.

[11] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2, Ser. B):293–320, 2003.

[12] P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, May 2000.

[13] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[14] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, March 1996.

[15] Christopher Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *arXiv preprint arXiv:0911.1393*, 2009.

[16] B. Reznick. Some concrete aspects of Hilbert's 17th problem. In *Contemporary Mathematics*, volume 253, pages 251–272. American Mathematical Society, 2000.

[17] E. de Klerk and D. V. Pasechnik. Approximation of the stability number of a graph via copositive programming. *SIAM Journal on Optimization*, 12(4):875–892, 2002.

[18] A. A. Ahmadi, A. Olshevsky, P. A. Parrilo, and J. N. Tsitsiklis. NP-hardness of deciding convexity of quartic polynomials and related problems. *Mathematical Programming*, 137(1-2):453–476, 2013.

[19] Z. Jarvis-Wloszek, R. Feeley, W. Tan, K. Sun, and A. Packard. Some controls applications of sum of squares programming. In *Proceedings of the $42^{th}$ IEEE Conference on Decision and Control*, pages 4676–4681, 2003.

[20] Ufuk Topcu, Andrew Packard, and Peter Seiler. Local stability analysis using simulations and sum-of-squares programming. *Automatica*, 44(10):2669–2675, 2008.

[21] A. A. Ahmadi, A. Majumdar, and R. Tedrake. Complexity of ten decision problems in continuous time dynamical systems. In *Proceedings of the American Control Conference*, 2013.

[22] J. Sturm. *SeDuMi version 1.05*, October 2001. Latest version available at `http://sedumi.ie.lehigh.edu/`.

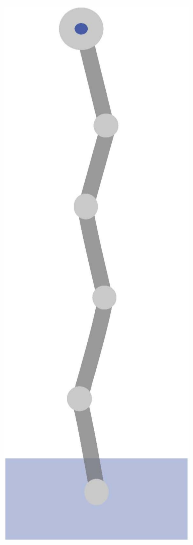
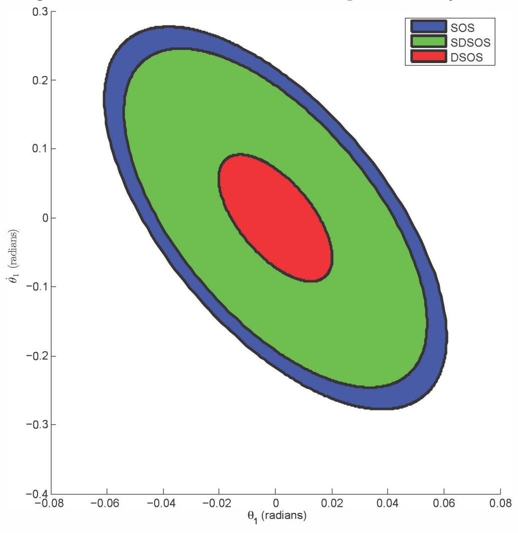
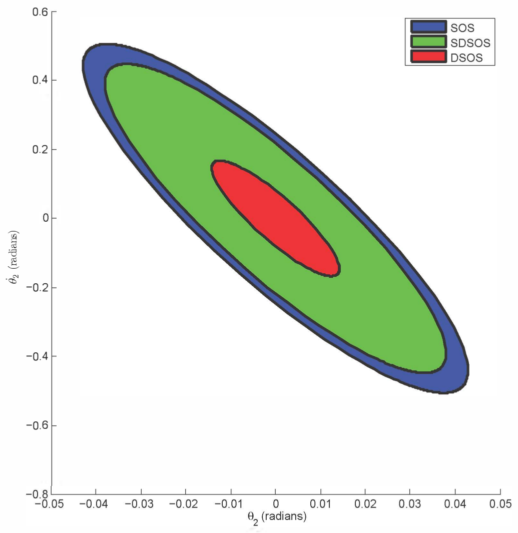


Fig. 1. An illustration of the 5-link pendulum system.



(a) Projection of the ROAs onto the $\theta_1$-$\dot{\theta}_1$ subspace.



(b) Projection of the ROAs onto the $\theta_2$-$\dot{\theta}_2$ subspace.

Fig. 2. Comparisons of the ROAs computed for the 5-link pendulum system using DSOS, SDSOS and SOS.